Broadside Tests for Transition and Stuck-at Faults

Irith Pomeranz

***Abstract*— A recent work showed that it is possible to transform a single-cycle test for stuck-at faults into a skewed-load test that detects the same stuck-at faults without performing logic or fault simulation. By using this transformation, it is possible to generate a compact skewed-load test set for stuck-at and transition faults. The advantage for test compaction is related to the fact that the test set contains a single test type. For cases where broadside tests are preferred over skewed-load tests, this paper studies the possibility of transforming a single-cycle test into a broadside test, and generating a compact broadside test set for stuck-at and transition faults. The paper addresses several challenges in order to achieve this goal without resorting to sequential test generation or state justification that have a high computational complexity. Experimental results for benchmark circuits demonstrate the levels of test compaction that can be achieved using small numbers of observation points.**

***Index Terms*— broadside tests, stuck-at faults, test compaction, transition faults.**

I. Introduction

Stuck-at and transition faults are commonly considered as necessary to detect. Additional fault models may be targeted to further improve the quality of a test set [1]-[6]. The discussion in this paper is applicable to any combination of fault models where some require single-cycle tests, and others, two-cycle tests.

In general, stuck-at faults are detected by single-cycle tests, while transition faults require two-cycle tests. In a standard scan circuit, broadside (launch-on-capture, or LOC) tests [7], or skewed-load (launch-on-shift, or LOS) tests [8] may be used for transition faults. Accordingly, test generation procedures typically produce single-cycle tests for stuck-at faults, and two-cycle tests for transition faults. The two-cycle tests detect both types of faults, while the single-cycle tests detect only single stuck-at faults. Therefore, the two-cycle tests are preferred for test compaction.

A recent work [9] showed that a single-cycle test for a stuck-at fault f can be transformed into a skewed-load test that detects f without performing logic or fault simulation. This is achieved by duplicating the single-cycle test during the second clock cycle of a skewed-load test. Consequently, instead of using a single-cycle test set for stuck-at faults, and a skewed-load test set for transition faults, it is possible to transform single-cycle tests into skewed-load tests, and produce a skewed-load test set for both fault models. The advantage is in the ability to compact the test set. While single-cycle tests do not detect transition faults, skewed-load tests detect both stuck-at and transition faults. Moreover, a skewed-load test that was transformed from a single-cycle test may detect more stuck-at and transition faults than a skewed-load test that was generated for transition faults. These properties are used for producing a compact skewed-load test set for both stuck-at and transition faults.

Skewed-load tests require the scan enable input to change at-speed between the first and second cycle of the test [10]-[11]. Broadside tests avoid this requirement. Consequently, broadside tests are sometimes preferred over skewed-load tests.

This paper studies the possibility of generating a broadside test set for stuck-at and transition faults in order to support test compaction when broadside tests are preferred over skewed-load tests. The paper achieves this goal without performing sequential test generation for stuck-at faults over the two clock cycles of a broadside test. This would have been required for generating broadside tests for stuck-at faults directly. Instead, the paper relies on a transformation of single-cycle tests for stuck-at faults into broadside tests that does not require complex processing. To make this possible, the paper needs to address several challenges as discussed next.

Transforming a single-cycle test for a stuck-at fault f into a broadside test by duplicating the single-cycle test during the second functional capture cycle of a broadside test requires a state justification procedure that has a high computational complexity. In addition, state justification is not always possible. Moreover, the broadside test may not detect f even if the transformation is possible. To avoid these complications, the paper suggests a transformation that duplicates the single-cycle test during the first functional capture cycle of a broadside test without any logic or fault simulation.

The transformation suggested in this paper does not guarantee that the broadside test will detect the same stuck-at faults as the single-cycle test. Therefore, stuck-at faults need to be simulated under the two clock cycles of a broadside test in order to determine accurately whether a fault is detected. Fault simulation may reveal that additional stuck-at faults are detected accidentally by the broadside test. This will contribute to test compaction. However, the main contribution to test compaction comes from using tests for stuck-at faults to detect transition faults.

To increase the possibility that stuck-at faults will be detected by a broadside test whose first functional capture cycle duplicates a single-cycle test, the paper observes that fault effects of stuck-at faults are propagated to the primary outputs or next-state variables during the first clock cycle of such a broadside test. Accordingly, the paper uses two options. The first option is related to the observation of primary output values. For transition faults, the primary output values are observed only during the second clock cycle of a broadside test. For stuck-at faults, it is sometimes important to observe the primary output values during the first clock cycle of a broadside test. The paper considers this option, and uses it when it increases the stuck-at fault coverage achieved by broadside tests.

The second option is based on the insertion of observation points on selected state variables that receive fault effects of stuck-at faults. The insertion of observation points on state variables is used in [12] for increasing the fault coverage of a built-in test generation scheme that produces multicycle broadside tests. It is also used in [13] for compacting a multicycle broadside test set. In [14]-[15] it is used for observation of fault effects during scan shift cycles to support test compaction. In this paper, this design-for-testability (DFT) approach makes it possible to obtain a test set for stuck-at and transition faults that consists only of broadside tests without performing sequential test generation for stuck-at faults. Experimental results for benchmark circuits show that at most a small number of observation points is needed. In addition, the same observation points can be used for other purposes. It is also possible to insert observation points into the combinational logic of the circuit [16]. However, when such observation points are used for transition faults, they allow transitions to be propagated through shorter paths, preventing small delay defects from being detected. Considering only stuck-at faults, the effects of observation points inside the combinational logic on the ability to compact the test set are expected to be small. Thus, it is sufficient to use observation points on state variables to allow broadside tests to detect stuck-at faults, and avoid the routing of observation points that are internal to the combinational logic.

After a broadside test set is obtained that detects all the detectable stuck-at and transition faults, static test compaction is applied to the broadside test set to reduce the number of tests. The ability to compact the test set is illustrated by Figure 1. The left part of Figure 1 shows the conventional case where a two-cycle broadside test set for

Irith Pomeranz is with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907, U.S.A. (e-mail: pomeranz@ecn.purdue.edu).

This work was supported in part by NSF grant CCF-1714147

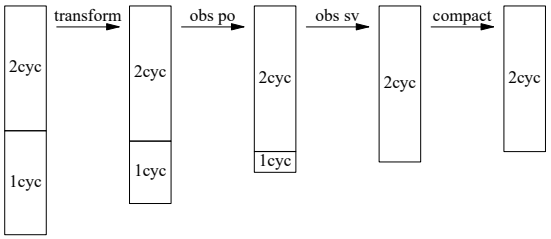

Fig. 1. Test compaction

TABLE I
NOTATION

symbol	meaning	symbol	meaning
sv	state variables	w_i^1	single-cycle test
po	primary outputs	p_i	scan-in state of w_i^1
f.c.	fault coverage	u_i	primary input vector of w_i^1
tr	transition faults	w_i^2	two-cycle test based on w_i^1
sa	stuck-at faults		
t_i	two-cycle test	T	broadside transition fault test set
s_i	scan-in state of t_i	W^1	single-cycle stuck-at test set
v_i	primary input vector of t_i	W^2	broadside test set based on W^1

symbol	meaning
T_{base}, T_{po}, T_{sv}, T_{comp}	test sets obtained during test compaction
n_{base}^1, n_{po}^1	numbers of faults detected by single-cycle tests

transition faults is complemented with single-cycle tests for stuck-at faults. In this paper, single-cycle tests are translated into broadside tests to obtain a broadside test set with the same or lower number of tests compared with the conventional case. This is achieved by transforming single-cycle tests into two-cycle tests, and observing primary outputs and state variables. The test set is then compacted to reduce the number of tests below that of the conventional case. Figure 1 illustrates these steps that yield a compact broadside test set.

The paper is organized as follows. The transformation of a single-cycle test into a broadside test is discussed in Section II. The procedure for deriving a compact broadside test set for stuck-at and transition faults is described in Section III. Experimental results are presented in Section IV. Table I summarizes the notation used, and provides the short forms in the header of Table II.

II. Transforming a Single-Cycle into a Broadside Test

A single-cycle test is denoted by $w_i^1 = \langle p_i, u_i, 1\rangle$, where p_i is the scan-in state, and u_i is the primary input vector that is applied during a functional capture cycle. The test ends with a scan-out operation. The primary output values are observed during the functional capture cycle of the test.

A broadside test is denoted by $t_i = \langle s_i, v_i, 2\rangle$, where s_i is the scan-in state, and v_i is a primary input vector that is applied during two consecutive functional capture cycles. Broadside tests with equal primary input vectors are commonly used to avoid the need to change the primary input vector at-speed during a test. The test ends with a scan-out operation. The primary output values are observed during the second functional capture cycle of the test.

A broadside test is shown in Figure 2. At the end of the first functional capture cycle the fault-free circuit is in a state that is denoted by $s_{i,1}$. At the end of the second functional capture cycle the fault-free circuit is in a state that is denoted by $s_{i,2}$.

The stuck-at fault where line g is stuck-at the value a is denoted by g/a. The transition fault that delays the $a \rightarrow a'$ transition on line g is denoted by $g : a \rightarrow a'$. Figure 2 shows the conditions required for detecting the transition fault $g : a \rightarrow a'$. The value $g = a$ is required

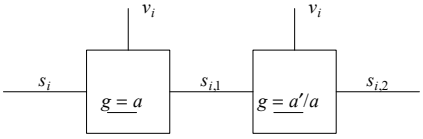

Fig. 2. Broadside test

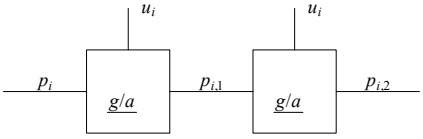

Fig. 3. Transforming a single-cycle into a broadside test

during the first functional capture cycle. The fault g/a needs to be detected during the second functional capture cycle.

Let $w_i^1 = \langle p_i, u_i, 1\rangle$ be a single-cycle test that detects the fault g/a. Referring to Figure 2, a transformation of w_i^1 into a broadside test $t_i = \langle s_i, v_i, 2\rangle$ can be performed by requiring that $s_{i,1} = p_i$ and $v_i = u_i$. In this case, the second functional capture cycle of the broadside test duplicates the single-cycle test, and the expectation is that f will be detected during this clock cycle of the broadside test. However, this transformation suffers from three issues. (1) To obtain $s_{i,1} = p_i$ under a broadside test it is necessary to compute a scan-in state s_i such that p_i is the next-state obtained for s_i and u_i. This computation requires a state justification procedure, with a high computational complexity. (2) Given p_i and u_i, a scan-in state s_i for a broadside test may not exist. (3) Even if s_i exists, the stuck-at fault g/a may be activated during the first functional capture cycle of the test, causing the faulty circuit to reach a state other than $s_{i,1} = p_i$ in the second cycle. In this case, f may not be detected.

Instead of using a transformation that requires a state justification procedure, the transformation suggested in this paper uses the single-cycle test $w_i^1 = \langle p_i, u_i, 1\rangle$ to define the broadside test $w_i^2 = \langle p_i, u_i, 2\rangle$. This test is shown in Figure 3. This transformation duplicates the single-cycle test during the first functional capture cycle of the broadside test.

If a stuck-at fault g/a is detected by the single-cycle test w_i^1, the broadside test activates the fault in the first functional capture cycle, and propagates fault effects to the primary outputs or next-state variables. Thus, the state $p_{i,1}$ may be obtained in the fault-free but not the faulty circuit. Fault simulation for a stuck-at fault g/a is carried out over the two functional capture cycles of the broadside test to determine whether the fault is detected.

III. Generating a Compact Broadside Test Set

Several test sets are computed in this section in order to eventually compute a compact test set that consists only of broadside tests, and detects all the detectable stuck-at and transition faults.

The input to the procedures described in this section consists of two test sets. A test set T consists of broadside tests for transition faults. A test set W^1 consists of single-cycle tests for stuck-at faults. Both test sets are compact. The set of target faults F consists of stuck-at and transition faults.

The test set W^1 is transformed into a broadside test set W^2 by transforming every single-cycle test $w_i^1 = \langle p_i, u_i, 1\rangle \in W^1$ into the broadside test $w_i^2 = \langle p_i, u_i, 2\rangle \in W^2$.

The test sets T and W^1 can be generated under the constraints of a test data compression method. The first steps of the procedure (including the transformation of W^1 into W^2) do not change scan-in states or primary input vectors. Therefore, the test sets remain

applicable under the same test data compression method. The last step of test compaction can be applied under the constraints of the same method.

A. Test Set T_{base}

The test set T_{base} is constructed by performing fault simulation with fault dropping of T, W^2 and W^1, in this order.

Initially, $T_{base} = \emptyset$, and F contains all the target faults. For every test $t_i \in T$, fault simulation with fault dropping is carried out for F under t_i. If any fault from F is detected, t_i is added to T_{base}. After T is considered, all the detectable transition faults, and some of the stuck-at faults, are detected (accurate fault simulation is carried out to account for the fact that a stuck-at fault may be activated in the first or second clock cycle of a broadside test).

Next, for every test $w_i^2 \in W^2$, fault simulation with fault dropping is carried out for F under w_i^2. If any fault from F is detected, w_i^2 is added to T_{base}. After W^2 is considered, additional stuck-at faults are detected.

Finally, for every test $w_i^1 \in W^1$, fault simulation with fault dropping is carried out for F under w_i^1. If any fault from F is detected, w_i^1 is added to T_{base}. After W^1 is considered, all the detectable stuck-at faults are detected.

With this order, the procedure prefers to include in T_{base} broadside tests for both stuck-at and transition faults. It adds single-cycle tests for stuck-at faults that are not detected by T and W^2.

The number of faults detected by single-cycle tests in T_{base} is denoted by n_{base}^1. If $n_{base}^1 = 0$, the test sets T_{po} and T_{sv} described in the next subsections are not needed, and $T_{po} = T_{sv} = T_{base}$.

B. Test Set T_{po}

For the test set T_{po}, it is noted that a single-cycle test $w_i^1 = \langle p_i, u_i, 1\rangle \in W^1$ may detect a stuck-at fault f on a primary output. In this case, the broadside test $w_i^2 = \langle p_i, u_i, 2\rangle \in W^2$ will propagate a fault effect of f to a primary output during the first functional capture cycle. The fault is not considered to be detected because the primary outputs are not observed during the first functional capture cycle of a broadside test. This decision is based on the fact that transition faults are not detected during the first clock cycle after the scan-in operation. However, stuck-at faults may be detected.

For the test set T_{po}, the primary outputs are observed during both functional capture cycles of a broadside test. Similar to T_{base}, the test set T_{po} is constructed by performing fault simulation with fault dropping of T, W^2 and W^1, in this order.

The number of faults that are detected by single-cycle tests in T_{po} is denoted by n_{po}^1. Observation of the primary outputs during both functional capture cycles of a broadside test is important if $n_{po}^1 < n_{base}^1$. Otherwise, if $n_{po}^1 = n_{base}^1$, primary output values are not observed during the first functional capture cycle, and $T_{po} = T_{base}$.

If $n_{po}^1 = 0$, the test set T_{sv} described in the next subsection is not computed, and $T_{sv} = T_{po}$.

C. Test Set T_{sv}

For the test set T_{sv}, the goal is to ensure that single-cycle tests are not needed for the detection of stuck-at faults. This can be achieved by inserting observation points as discussed next.

Let X^1 be the subset of single-cycle tests from W^1 that are included in T_{po}. Let F^1 include every stuck-at fault $f \in F$ such that f is detected by a test $w_i^1 \in X^1$.

Detections on the primary outputs are already considered for the computation of T_{po}. Therefore, a fault $f \in F^1$ requires a single-cycle test $w_i^1 \in W^1$ to be included in T_{po} because its fault effects are propagated to the next-state variables under w_i^1. As in Figure 3, let the next-state of the fault-free circuit under p_i and u_i be $p_{i,1}$. Let the next-state of the faulty circuit in the presence of f be $q_{i,1}$. If $p_{i,1}$ and $q_{i,1}$ differ in the value of state variable j, an observation point on state variable j will allow f to be detected.

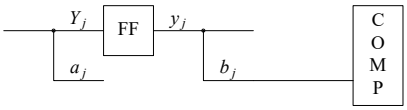

Fig. 4. Observation point on state variable

Figure 4 shows a flip-flop with a next-state variable Y_j and a present-state variable y_j. An observation point may be added on the next-state variable as a fanout branch a_j, or the present-state variable as a fanout branch b_j.

With the observation point a_j, a stuck-at fault on Y_j will not benefit from a_j. Therefore, in this paper, an observation point is placed on a present-state variable. With the observation point b_j, additional stuck-at faults will be detected during the second functional capture cycle of the test. Observation points are used only for the detection of stuck-at faults.

Figure 4 also illustrates the possibility of using output compaction logic to capture the values of the observation points. The output compaction logic that captures primary output values and scan-out states can also capture the values of the observation points.

The selection of present-state variables for observation point insertion is described next.

The procedure associates with every fault $f \in F^1$ a set of possible observation points $J(f)$. Initially, $J(f) = \emptyset$. For every test $w_i^1 \in X^1$, the procedure simulates every fault $f \in F^1$ under w_i^1. If the fault is detected on a next-state variable Y_j, the index j is added to $J(f)$.

The procedure uses a greedy set covering procedure to select a subset J_{obs} such that, for every fault $f \in F^1$, there is an index $j \in J_{obs}$ of a state variable where f can be detected, or $j \in J(f)$.

After inserting observation points on the present-state variables in J_{obs}, the test set T_{sv} is constructed by performing fault simulation with fault dropping of T and W^2, in this order (it is not necessary to simulate W^1).

D. Test Set T_{comp}

The test set T_{sv} contains only broadside tests. The tests are from two sources, the transition fault test set T, and the single-cycle stuck-at test set W^1 that was transformed into the broadside test set W^2. Each one of the test sets was compacted separately, but the two test sets were not compacted together. Therefore, application of a static test compaction procedure can reduce the size of T_{sv}.

Initially, $T_{comp} = T_{sv}$. The test set T_{comp} is compacted by a static test compaction procedure that is based on the approach from [17]. The procedure has two subprocedures that are applied iteratively. Both subprocedures use fault simulation with fault dropping to associate a set of detected faults $D(t)$ with every test $t \in T_{comp}$.

The first subprocedure reorders the tests in T_{comp} such that the tests appear by decreasing number of detected faults. Reordering and fault simulation with fault dropping are repeated until no further changes in the order of the tests is obtained. If $|D(t)| = 0$ is obtained for a test t, the test is removed from T_{comp}.

The second subprocedure applies a basic step where a pair of tests $t_{mod} \in T_{comp}$ and $t_{rem} \in T_{comp}$ is considered. The procedure attempts to modify $t_{mod} \in T_{comp}$ so as to detect all the faults in $D(t_{mod}) \cup D(t_{rem})$. If the modification is successful, t_{rem} is removed from T_{comp}.

TABLE II
EXPERIMENTAL RESULTS

			observed			tests					f.c.		
circuit	sv	iter	po	sv	%sv	tot	T	W^2	W^1	n1	tr	sa	ntime
systemcaes	670	0	0	0	0.000	207	202	5	0	0	88.751	99.995	1.00
systemcaes	670	1	0	0	0.000	202	199	3	0	0	88.751	99.995	418.44
systemcaes	670	6	0	0	0.000	188	186	2	0	0	88.751	99.995	1582.07
systemcdes	190	0	0	0	0.000	92	91	1	0	0	96.165	100.000	1.00
systemcdes	190	1	0	0	0.000	91	91	0	0	0	96.165	100.000	51.36
systemcdes	190	2	0	0	0.000	88	88	0	0	0	96.165	100.000	83.81
s35932	1728	0	0	0	0.000	40	30	8	2	1244	71.800	89.809	1.00
s35932	1728	0	0	1244	3.346	38	30	8	0	0	71.800	89.809	3.76
s35932	1728	4	0	1244	3.346	33	27	6	0	0	71.800	89.809	42.23
b14	247	0	0	0	0.000	417	207	203	7	32	72.050	95.083	1.00
b14	247	0	0	31	0.360	410	207	203	0	0	72.050	95.083	3.98
b14	247	3	0	31	0.360	382	190	192	0	0	72.387	95.123	445.39
b20	494	0	0	0	0.000	495	306	187	2	33	79.655	94.315	1.00
b20	494	0	0	33	0.171	493	306	187	0	0	79.655	94.315	4.00
b20	494	15	0	33	0.171	401	241	160	0	0	79.953	94.532	2740.42
aes_core	530	0	0	0	0.000	312	311	0	1	1	96.187	100.000	1.00
aes_core	530	0	0	1	0.002	311	311	0	0	0	96.187	100.000	4.16
aes_core	530	3	0	1	0.002	307	307	0	0	0	96.187	100.000	79.22
spi	229	0	0	0	0.000	889	865	23	1	1	82.716	99.985	1.00
spi	229	0	0	1	0.017	889	865	24	0	0	82.716	99.985	4.00
spi	229	1	0	1	0.017	865	844	21	0	0	82.716	99.985	424.56
spi	229	22	0	1	0.017	756	743	13	0	0	82.923	99.985	19768.15
tv80	359	0	0	0	0.000	918	658	221	39	60	82.076	99.451	1.00
tv80	359	0	0	33	0.228	896	658	238	0	0	82.076	99.451	3.69
tv80	359	9	0	33	0.228	709	585	124	0	0	82.609	99.477	2242.18
s5378	179	0	0	0	0.000	288	180	19	89	262	77.863	99.152	1.00
s5378	179	0	1	0	0.000	245	180	55	10	11	77.863	99.152	1.63
s5378	179	0	1	3	0.056	237	180	57	0	0	77.863	99.152	2.24
s5378	179	8	1	3	0.056	180	155	25	0	0	77.863	99.152	579.73
s5378	179	9	1	3	0.056	177	154	23	0	0	77.863	99.152	639.15
s9234	228	0	0	0	0.000	426	355	32	39	73	76.594	93.626	1.00
s9234	228	0	1	0	0.000	425	355	32	38	72	76.594	93.626	1.99
s9234	228	0	1	7	0.075	400	355	45	0	0	76.594	93.626	2.94
s9234	228	2	1	7	0.075	355	334	21	0	0	76.594	93.626	239.85
s9234	228	4	1	7	0.075	340	323	17	0	0	76.594	93.626	414.18
s13207	669	0	0	0	0.000	420	349	56	15	42	80.317	98.542	1.00
s13207	669	0	1	0	0.000	411	349	60	2	2	80.317	98.542	1.93
s13207	669	0	1	2	0.015	409	349	60	0	0	80.317	98.542	2.86
s13207	669	2	1	2	0.015	349	332	17	0	0	80.317	98.542	727.77
s13207	669	7	1	2	0.015	317	312	5	0	0	80.325	98.542	2452.55
s38417	1636	0	0	0	0.000	664	648	11	5	6	97.164	99.503	1.00
s38417	1636	0	1	0	0.000	663	648	11	4	4	97.164	99.503	2.00
s38417	1636	0	1	4	0.010	660	648	12	0	0	97.164	99.503	2.99
s38417	1636	1	1	4	0.010	647	636	11	0	0	97.164	99.503	153.07
s38417	1636	9	1	4	0.010	459	457	2	0	0	97.188	99.503	12246.25
s38584	1452	0	0	0	0.000	629	536	58	35	80	71.239	95.819	1.00
s38584	1452	0	1	0	0.000	622	536	56	30	51	71.239	95.819	1.98
s38584	1452	0	1	32	0.083	591	536	55	0	0	71.239	95.857	2.94
s38584	1452	1	1	32	0.083	536	501	35	0	0	71.265	95.857	237.77
s38584	1452	12	1	32	0.083	437	423	14	0	0	71.291	95.857	2242.62
b15	447	0	0	0	0.000	603	488	102	13	115	81.118	98.640	1.00
b15	447	0	1	0	0.000	600	488	103	9	111	81.118	98.640	2.08
b15	447	0	1	110	0.618	590	488	102	0	0	81.118	98.640	3.27
b15	447	8	1	110	0.618	493	437	56	0	0	81.929	98.679	1503.12
wb_dma	523	0	0	0	0.000	230	175	48	7	10	75.634	100.000	1.00
wb_dma	523	0	1	0	0.000	228	175	47	6	9	75.634	100.000	1.97
wb_dma	523	0	1	8	0.095	223	175	48	0	0	75.634	100.000	2.90
wb_dma	523	1	1	8	0.095	175	138	37	0	0	75.687	100.000	535.09
wb_dma	523	8	1	8	0.095	147	116	31	0	0	75.699	100.000	2363.35

TABLE III
TEST COMPACTION RESULTS

circuit	T_{base}	T	circuit	T_{base}	T
systemcaes	-9.18	-6.93	s5378	-38.54	-1.67
systemcdes	-4.35	-3.30	s9234	-20.19	-4.23
s35932	-17.50	10.00	s13207	-24.52	-9.17
b14	-8.39	84.54	s38417	-30.87	-29.17
b20	-18.99	31.05	s38584	-30.52	-18.47
aes_core	-1.60	-1.29	b15	-18.24	1.02
spi	-14.96	-12.60	wb_dma	-36.09	-16.00
tv80	-22.77	7.75			

The procedure considers all the test pairs such that $|D_{rem}| = 1$, $|D_{mod}| > 1$, and t_{mod} is not in the first quarter of the test set. These conditions ensure that the tests detect moderate numbers of faults, and makes it likely that they can be modified or removed. In addition, a test that detects a single fault is not considered for modification in order to ensure that it remains a candidate for removal.

IV. EXPERIMENTAL RESULTS

This section presents the results of the computation of the test sets T_{base}, T_{po}, T_{sv} and T_{comp} for benchmark circuits.

The results are shown in Table II. Table III shows a summary of the test compaction levels reported in Table II. The circuits in Table II are partitioned into three groups according to their requirements for achieving a test set that consists only of broadside tests.

For every circuit, every one of the test sets T_{base}, T_{po}, T_{sv} and T_{comp} is shown on a separate row in Table II. A test set is omitted if it is equal to the previous test set. In the case of T_{comp}, intermediate test sets are sometimes reported as discussed later.

For every test set, column sv shows the number of state variables. For T_{comp}, column $iter$ shows the iteration of the test compaction procedure. For the other test sets the iteration is zero by default.

Column $observed$ subcolumn po shows whether or not primary output values are observed during the first functional capture cycle of a broadside test. Subcolumn sv shows the number of observation points on present-state variables. Subcolumn $\%sv$ shows the number of observation points as a percentage of the total number of lines.

Column $tests$ subcolumn tot shows the total number of tests in the test set. Subcolumns T, W^2 and W^1 show the numbers of tests from the corresponding test sets. When the test compaction procedure modifies a test, its source is preserved. Subcolumn $n1$ shows the number of stuck-at faults that are detected by single-cycle tests (this can be n^1_{base}, n^1_{po} or n^1_{sv}).

Column $f.c.$ shows the transition and stuck-at fault coverages. Column $ntime$ shows the cumulative run time, normalized to the run time required for producing T_{base}. The run time for T_{base} consists of fault simulation with fault dropping of T, W^2 and W^1.

The following points can be observed from Table II. There are several circuits for which the tests in T and W^2 are sufficient for detecting all the stuck-at faults. For other circuits, the additional observation of primary output values, and/or the insertion of observation points, eliminate single-cycle tests from the test set.

Even when the number of broadside tests is increased to allow single-cycle tests to be omitted, the overall number of tests in T_{po} and T_{sv} is reduced relative to T_{base}. This is related to the fact that broadside tests can detect more stuck-at faults than single-cycle tests. The importance of observing primary output values is related to the fact that the primary outputs capture the results of the computations that the circuit performs, and are, therefore, sensitive to faults.

The static test compaction procedure reduces the number of tests further. The effectiveness of the procedure can be seen in two ways. (1) The number of tests in T_{comp} is reduced significantly relative to T_{base}, T_{po} and T_{sv}. (2) Since T is a compact test set for transition faults, and T_{comp} detects both transition and stuck-at faults, it is expected that $|T_{comp}| \geq |T|$ will be obtained. Nevertheless, there are many cases where $|T_{comp}| \leq |T|$ is obtained. This is possible because more iterations of test compaction are applied to T_{comp}.

Table III further illustrates the reduction in the number of tests by showing the percentage reduction of T_{comp} relative to T_{base}, and relative to T. A negative number implies that the number of tests in T_{comp} is lower than in T_{base} or T.

The reduction in the number of tests has a computational cost. It is possible to terminate the static test compaction procedure earlier, with a larger test set. To illustrate this point, for circuits where $|T_{comp}| < |T|$ at termination, Table II includes the results of the static test compaction procedure when $|T_{comp}| = |T|$.

The percentage of observation points varies with the circuit. The higher percentages are obtained for circuits where the additional observation of primary output values is not useful for detecting stuck-at faults. When this approach is effective, the percentage of observation points is lower than 1% for all the circuits in the third part of Table II, and lower than 0.1% for most of these circuits.

V. Concluding Remarks

This paper described a procedure for generating a compact broadside test set for stuck-at and transition faults. The procedure is based on a transformation of a single-cycle test for stuck-at faults into a broadside test by duplicating the single-cycle test during the first functional capture cycle of the broadside test. This transformation does not require logic or fault simulation, and it avoids the need for sequential test generation or state justification. To compensate for the fact that stuck-at faults may not be detected by the transformed tests, the paper used three options: (1) accurate simulation of stuck-at faults under broadside tests, (2) observation of the primary output values during the first functional capture cycle of a broadside test, and (3) the insertion of observation points on selected present-state variables. Experimental results were presented for benchmark circuits to demonstrate the levels of test compaction that can be achieved, and numbers of observation points needed.

References

[1] L. N. Reddy, I. Pomeranz and S. M. Reddy, "COMPACTEST-II: A Method to Generate Compact Two-pattern Test Sets for Combinational Logic Circuits", in Proc. Intl. Conf. on Computer-Aided Design, 1992, pp. 568-574.

[2] R. Desineni, K. N. Dwarkanath and R. D. Blanton, "Universal Test Generation using Fault Tuples", in Proc. Intl. Test Conf., 2000, pp. 812-819.

[3] S. M. Reddy, G. Chen, J. Rajski, I. Pomeranz, P. Engelke and B. Becker, "A Unified Fault Model and Test Generation Procedure for Interconnect Opens and Bridges", in Proc. Europ. Test Symp., 2005, pp. 22-27.

[4] S. Goel and R. A. Parekhji, "Choosing the Right Mix of At-Speed Structural Test Patterns: Comparisons in Pattern Volume Reduction and Fault Detection Efficiency", in Proc. Asian Test Symp., 2005, pp. 330-336.

[5] S. Alampally, R. T. Venkatesh, P. Shanmugasundaram, R. A. Parekhji and V. D. Agrawal, "An Efficient Test Data Reduction Technique through Dynamic Pattern Mixing Across Multiple Fault Models", in Proc. VLSI Test Symp., 2011, pp. 285-290.

[6] C.-H. Wu and K.-J. Lee, "Transformation of Multiple Fault Models to a Unified Model for ATPG Efficiency Enhancement", in Proc. Int. Test Conf., 2016, pp. 1-10.

[7] J. Savir and S. Patil, "Broad-Side Delay Test", IEEE Trans. on Computer-Aided Design, Aug. 1994, pp. 1057-1064.

[8] J. Savir and S. Patil, "Scan-Based Transition Test", IEEE Trans. on Computer-Aided Design, Aug. 1993, pp. 1232-1241.

[9] I. Pomeranz, "Skewed-Load Tests for Transition and Stuck-at Faults", IEEE Trans. on Computer-Aided Design, 2019.

[10] N. Ahmed, M. Tehranipoor, C. P. Ravikumar and K. M. Butler, "Local At-Speed Scan Enable Generation for Transition Fault Testing Using Low-Cost Testers", IEEE Trans. on Computer-Aided Design, May 2007, pp. 896-905.

[11] G. Xu and A. D. Singh, "Scan Cell Design for Launch-on-Shift Delay Tests with Slow Scan Enable", IET Computers & Digital Techniques, May 2007, pp. 213-219.

[12] Y. Sato, H. Yamaguchi, M. Matsuzono and S. Kajihara, "Multi-Cycle Test with Partial Observation on Scan-Based BIST Structure", in Proc. Asian Test Symp., 2011, pp. 54-59.

[13] I. Pomeranz, "Observation Points on State Variables for the Compaction of Multicycle Tests", IEEE Trans. on VLSI Systems, 2018.

[14] F. Zhang, D. Hwong, Y. Sun, A. Garcia, S. Alhelaly, G. Shofner, L. Winemberg and J. Dworak, "Putting Wasted Clock Cycles to Use: Enhancing Fortuitous Cell-aware Fault Detection with Scan Shift Capture", in Proc. Intl. Test Conf., 2016 pp. 1-10.

[15] G. Mrugalski, J. Rajski, J. Solecki, J. Tyszer and C. Wang, "Trimodal Scan-Based Test Paradigm", IEEE Trans. on VLSI Systems, March 2017, Vol. 25, No. 3, pp. 1112-1125.

[16] C. Acero, D. Feltham, Y. Liu, E. Moghaddam, N. Mukherjee, M. Patyra, J. Rajski, S. M. Reddy, J. Tyszer and J. Zawada, "Embedded Deterministic Test Points", IEEE Trans. on VLSI Systems, 2017, vol. 25, no. 10, pp. 2949-2961.

[17] I. Pomeranz, "Static Test Compaction for Scan Circuits by Using Restoration to Modify and Remove Tests", IEEE Trans. on Computer-Aided Design, Dec. 2014, pp. 1955-1964.