Test Data Compression Based on Output Dependence

Irith Pomeranz[1]
School of Electrical & Computer Eng.
Purdue University
W. Lafayette, IN 47907

and

Sudhakar M. Reddy[2]
Electrical & Computer Eng. Dept.
University of Iowa
Iowa City, IA 52242

In a large circuit it is common to find that an output of the circuit depends structurally on a proper subset of the circuit inputs. We use this observation to provide test data compression. The proposed approach can be used in addition to test data compression techniques based on encoding.

Structural output dependence is utilized in this work to provide data compression as follows. Consider a test set T that needs to be applied to the circuit. Suppose that the circuit inputs can be divided into subsets $A_0, A_1, \cdots, A_{m-1}$ such that for every circuit output b_j there exists an input subset A_k satisfying the condition that b_j depends only on inputs in A_k. For simplicity, we also assume that all the subsets A_k contain the same number of inputs, denoted by N_A (later we will ensure that this condition is satisfied). The value of N_A is determined by the output that depends on the maximum number of inputs. Let the number of circuit inputs be N_C. If $N_A \ll N_C$, it is possible to use the input subsets in order to reduce the amount of input test data by loading into the circuit patterns of length N_A instead of loading patterns of length N_C.

There are two issues to consider in this scheme: (1) the computation of a set of patterns P of length N_A that will be loaded to the circuit, and (2) the computation of tests for the circuit based on the set P. We discuss these issues next.

In the worst case, to ensure complete fault coverage, we must include in P a set of patterns P_k based on T for every input subset A_k. For every test $t \in T$, P_k contains a pattern p_k that consists of the values assigned by t only to the inputs in A_k. For example, if $T = \{00011, 11100\}$, the circuit inputs are $a_0, \cdots, a_4$, and $A_0 = \{a_0, a_1, a_2\}$, we have $P_0 = \{000, 111\}$; and for $A_1 = \{a_0, a_2, a_4\}$ we have $P_1 = \{001, 110\}$. The set $P = \cup_{k=0}^{m-1} P_k$ is guaranteed to allow us to detect the same faults as T if each $p_k \in P$ is expanded and applied to the input subset A_k for which it was derived. For example, $000 \in P_0$ should be expanded into $000xx$, while $001 \in P_1$ should be expanded into $0x0x1$. In this case, compression results from the fact that input patterns of length N_A (instead of N_C) need to be loaded. However, the number of patterns in P may be larger than the number of tests in T since each test in T contributes multiple patterns to P. This may limit the level of compression that can be achieved.

To provide a higher level of compression, we observe that every pattern in $\cup_{k=0}^{m-1} P_k$ can be used to define m tests for the circuit, one for every subset of inputs A_k. For example, $p = 001 \in P_1$ above defines a pattern of the form $001xx$ with respect to A_0, and $0x0x1$ with respect to A_1. When every pattern in P is used for defining several tests, not all the patterns in $\cup_{k=0}^{m-1} P_k$ may be necessary in order to achieve complete fault coverage. We define a set of patterns $P \subseteq \cup_{k=0}^{m-1} P_k$, which is a minimal set of patterns necessary for achieving complete fault coverage. Thus, we take advantage of the ability to use $p \in P$ to define multiple tests, in order to reduce the size of P and improve the level of compression.

The level of compression is as follows. The size of P (in bits) is $N_A |P|$. The size of the given test set T is $N_C |T|$. Thus, the level of compression is $(N_A |P|)/(N_C |T|)$. If N_A is sufficiently smaller than N_C and P is sufficiently small compared to T, storage of P requires fewer bits than T. This can be used to provide input test data compression.

The use of P instead of T requires the inclusion on-chip of a *distribution block*. Depending on the subset A_k, the distribution block will apply the patterns in P to the appropriate circuit inputs. A circuit with a distribution block *DIST* is shown in Figure 1. The distribution block transforms an input pattern p of length N_A into input patterns of length N_C of the circuit-under-test (*CUT*) by applying p to the appropriate inputs and assigning arbitrary (random) values to the remaining inputs.

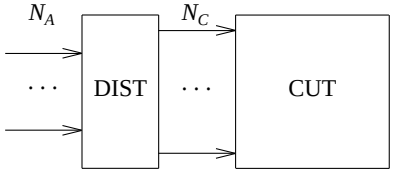

Figure 1: Output dependence based compression

It should be noted that the number of tests applied to the circuit under the proposed approach is $m|P|$, where m is the number of input subsets. Thus, the total number of tests applied to the circuit may be higher than the number of tests in T, and the test application time may be higher as well. The approach proposed here is therefore applicable in instances where test storage is a limiting factor, while an increase in test application time is acceptable. It should be pointed out that application of a larger number of tests achieves multiple detections of faults, which is known to lead to a higher defect coverage.

To partition the circuit inputs into subsets, we apply the following procedure. Initially, we define a set of input subsets α where for every output b_j we include the subset of inputs A_j that drives output b_j. We remove from α every subset $A_i \in \alpha$ which is contained in another subset $A_j \in \alpha$. We then merge as many subsets in α as possible without increasing the subset sizes beyond N_A. After merging subsets, we may again have a subset A_i which is contained in another subset A_j. We remove all subsets such as A_i from α. Finally, if any of the subsets in α contains fewer than N_A inputs, we add inputs to it arbitrarily until its size reaches N_A. This is done to simplify the construction of P, which will contain patterns of a uniform length N_A.

Given a set of input subsets $\alpha = \{A_0, A_1, \cdots, A_{m-1}\}$ with N_A inputs in each subset, an input pattern p of length N_A is used as follows. The pattern p is applied to every input subset A_i. When p is applied to A_i, the inputs in A_i are assigned values

1. Research supported in part by NSF Grant No. CCR-0098091 and in part by SRC Grant No. 2001-TJ-950.
2. Research supported in part by NSF Grant No. CCR-0097905 and in part by SRC Grant No. 2001-TJ-949.

according to p, while the remaining inputs are assigned arbitrary (random) values. The set of faults detected by p is determined by simulating p m times, once for every subset A_i. For a set of input patterns P of size $|P|$, we apply to the circuit $m|P|$ input patterns obtained by expanding every $p \in P$ with respect to every $A_i \in \alpha$.

To support this test application scheme, the distribution block consists of the following components. The inputs of the distribution block are $p_0, \cdots, p_{N_A}$ that form a pattern $p = p_0 \cdots p_{N_A} \in P$. The outputs of the distribution block are connected to the inputs of the CUT, $a_0, \cdots, a_{N_C-1}$. A counter that counts from 0 to $m-1$ determines the subset of CUT inputs to which p will be assigned. A register $r = r_0 r_1 \cdots r_{N_C-N_A-1}$ determines the values of the CUT inputs that are not included in the subset to which p is applied (an $LFSR$ producing random values is a good choice for r). For every CUT input a_i we have a multiplexer that determines, depending on the state of the counter, whether the input is assigned a bit from p or a bit from r.

Next, we describe the derivation of a set of patterns P based on a test set T using a set of input subsets α. Construction of P proceeds in two phases. In the first phase, we include in a set $\hat{P}$ candidate patterns extracted from the given test set T. Before adding a pattern p to $\hat{P}$, its expanded patterns with respect to the subsets in α are simulated. A pattern p that does not help in detecting new faults is not included in $\hat{P}$. The construction of $\hat{P}$ uses double-detection simulation. This is required for the selection of P, as explained later. Using the results of the first phase, in the second phase we select a minimal subset $P \subseteq \hat{P}$ such that P allows us to detect all the faults detected by the given test set T.

To construct $\hat{P}$, we consider the patterns assigned to $A_0, A_1, \cdots, A_{m-1}$ under the test set T. For A_i, we obtain a set of patterns P_i by restricting T to the inputs in A_i. Each pattern in every set P_i is expanded with respect to every input subset. Each expanded pattern is fault simulated as part of a double-detection fault simulation process. Under this process, a fault is dropped only after it has been detected twice, by two different patterns. During this process, we also store for every fault f the first and second pattern out of $\cup P_i$ that yielded expanded patterns which detected f. We denote these patterns by $p_{det1}(f)$ and $p_{det2}(f)$, respectively. If f is detected only once, $p_{det2}(f) = -$, and if f is not detected, $p_{det1}(f) = p_{det2}(f) = -$. Every pattern p with an expanded pattern that detects a fault for the first or second time is stored in $\hat{P}$.

Once $\hat{P}$ is defined, we select a minimal subset $P \subseteq \hat{P}$ to detect all the faults detected by T. Selection of P proceeds as follows. We denote by F the set of faults detected by T. We first consider the faults in F that are detected by single patterns out of $\hat{P}$. For such a fault $f_i \in F$, $p_{det1}(f_i)$ is the only pattern in $\hat{P}$ that can be used to detect f_i. We add $p_{det1}(f_i)$ to P. We then simulate the expanded patterns of $p_{det1}(f_i)$ with respect to A_k, for $k = 0,1, \cdots, m-1$, and we drop the detected faults from F. We repeat this process as long as there are faults $f_i \in F$ which are detected only once under $\hat{P}$.

To detect the remaining faults, we find for every pattern $p \in \hat{P}$ the number of faults $f_i \in F$ for which $p_{det1}(f_i) = p$ or $p_{det2}(f_i) = p$. We denote this number by $n_{det}(p)$. We note that expanded patterns of p may detect other faults out of F, that were dropped before p was considered in Phase 1. Therefore, expanded patterns of p may detect more than $n_{det}(p)$ faults out of F. However, $n_{det}(p)$ provides a good heuristic for selecting additional patterns out of p. We select $p \in \hat{P}$ for which $n_{det}(p)$ is the largest. We then simulate the expanded patterns of p with respect to A_k, for $k = 0,1, \cdots, m-1$, and we drop the detected faults from F. We repeat this process as long as F is not empty.

We applied the procedures described above to the combinational logic of benchmark circuits with large numbers of inputs. In our experiment, we expand a pattern p with respect to a subset of inputs A_i by assinging random values to the inputs not included in A_i.

In Table 1 we show the following information about the circuits we consider. After the circuit name we show the number of circuit inputs N_C, the number of tests $|T|$, and the number of bits required for storing the test set, $N_C|T|$.

Table 1: Circuit parameters

circuit	NC	|T|	test stor
s420	35	43	1505
s641	54	22	1188
s953	45	76	3420
s1423	91	26	2366
s5378	214	100	21400
s9234	247	111	27417
s13207	700	235	164500
s15850	611	97	59267
s38417	1664	87	144768
b04	78	72	5616
b11	38	76	2888
b14	280	194	54320
b20	527	335	176545

Table 2: Random filling of expanded patterns

circuit	NA	m	|P|	patt stor	ratio inp	ratio stor
s420	34	2	37	1258	0.97	0.84
s641	27	5	25	675	0.50	0.57
s953	18	6	57	1026	0.40	0.30
s1423	59	5	32	1888	0.65	0.80
s5378	61	9	129	7869	0.29	0.37
s9234	83	10	171	14193	0.34	0.52
s13207	212	6	273	57876	0.30	0.35
s15850	183	11	193	35319	0.30	0.60
s38417	99	42	321	31779	0.06	0.22
b04	40	7	34	1360	0.51	0.24
b11	25	4	50	1250	0.66	0.43
b14	226	2	140	31640	0.81	0.58
b20	294	4	222	65268	0.56	0.37

In Table 2 we show the results of the proposed procedure. After the circuit name we show the number of inputs in each subset N_A, the number of subsets m, the number of patterns in P, and the number of bits required for storing P, $N_A|P|$. In the last column we show the ratio N_A/N_C of the number of inputs, and the ratio $(N_A|P|)/(N_C|T|)$ of the storage requirements.

From Table 2 it can be seen that there are several circuits where the outputs depend on subsets of the inputs, with $N_A \ll N_C$. The number of patterns in P required to achieve the same fault coverage as the test set T is in many cases larger than the number of tests in T. Nevertheless, the storage requirements of P are lower than the storage requirements of T in all the cases. The number of different input subsets m is relatively small, implying that each pattern $p \in P$ results in a small number of test patterns.

The total number of tests applied to the circuit under the proposed scheme is equal to the number of input subsets m multiplied by the number of patterns in P, or $m|P|$. This number can be reduced by reducing the number of input subsets m.