Defect Aware Test Patterns

Huaxing Tang, Gang Chen, Sudhakar M. Reddy, Chen Wang, Janusz Rajski, Irith Pomeranz

Defect Aware Test Patterns

Huaxing Tang, Gang Chen, Sudhakar M. Reddy
ECE Department, University of Iowa, Iowa City, IA 52242

Chen Wang, Janusz Rajski
Mentor Graphic Corporation, Wilsonville, OR 97070

Irith Pomeranz
School of ECE, Purdue Univ., West Lafayette, IN 47907

Abstract

A method to generate test patterns referred to as defect aware test patterns is proposed. Defect aware test patterns have greater ability to detect un-modeled defects. The proposed method can be used with any test generation procedure to improve the effectiveness of the tests in detecting un-modeled defects. Experimental results on several industrial designs show the effectiveness of defect aware tests. We also propose a measure to estimate the effectiveness of given test sets in detecting un-modeled defects.

1. Introduction

Defects in deep submicron VLSI circuits are known to be predominantly opens and shorts or bridges. Two different approaches are currently being pursued to generate tests to achieve low value for defective parts per million shipped parts. One is to generate tests for more elaborate fault models that model defects more accurately [3,7,9,10,20, 21] and the other is to generate n-detect test sets using simpler fault models such as single stuck-at line (SSL) faults [4,8,11,22]. n-detect test sets contain tests to detect each modeled fault n times. The effectiveness of n-detect test sets is based on the observation that the possibility of accidentally activating an un-modeled defect and propagating its effect to observed outputs is enhanced by such test sets [11,22].

In this work we consider enhancement of the probability of detecting un-modeled defects using single pattern tests to detect static defects. Although not discussed in this work, the basic idea behind the proposed method can be extended for use with two-pattern tests to enhance the coverage of un-modeled dynamic defects.

As observed above, interconnect opens and bridges are known to be the most likely defects in deep submicron designs. We will present results on industrial circuits that show that tests for stuck-at faults detect essentially all detectable interconnect opens. We target improvement of coverage of un-modeled bridge defects when n-detect tests for SSL faults are generated.

The remainder of this paper is organized as follows. In Section 2 we review previous works related to generating tests to improve coverage of un-modeled defects and then present the basic idea behind the proposed method. In Section 3 we give details of the proposed method and of our implementation of generating defect aware tests using an ATPG that targets SSL fault detection. In Section 4 we propose a measure to estimate the coverage of un-modeled defects by given test sets. Experimental results are presented in Section 5 and Section 6 concludes the paper.

2. Preliminaries

Some earlier works have considered methods to generate n-detect tests for SSL faults with improved coverage of un-modeled defects. The method proposed in [15] generates n-detect tests such that every subset of faults of size m is either detected by a single test or each one of the faults in the subset is detected at least n times. The method proposed in [16] allows a test t that detects a fault f to be counted as an additional detection of f only if t satisfies a condition on its relationship with each one of the earlier generated tests that detects fault f. The methods discussed above do not directly consider increasing the probability of accidental detection of un-modeled bridges.

Several methods to model bridges have been proposed in the literature [1,6,9,12-14,20]. A model that is effective in modeling the bridges that can be used to efficiently generate tests to detect bridges is the so called *4-way bridging fault model* [20]. This model has been used to generate manufacturing tests for commercial processors [2,7,20]. We use 4-way bridging faults as surrogates for un-modeled defects in evaluating the method proposed in this work to generate defect aware tests. For the sake of completeness we describe below the 4-way bridging fault model.

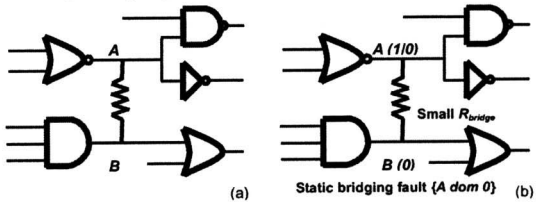

Figure 1. Bridging fault examples: (a) a bridge; (b) a DOM bridging fault

In the 4-way bridging fault model, each bridge defect between nodes <A, B>, shown in Figure 1(a), is modeled by four bridging faults: {*A dom 0, B dom 0, A*

Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'05)
1530-1591/05 $ 20.00 IEEE

dom 1, B dom 1}. The bridging fault *{A dom 0}* shown in Figure 1(b) is defined as node *A* is stuck-at-0 in the presence of the bridge if node *B* is 0. In this case, the signal value on node *A* is determined or dominated by the signal value on *B* in the presence of the bridge, thus leading to the fault name *{A dom 0}*. Node *A* is referred to as the *victim* node and node *B* as the *aggressor* node in the bridging fault *{A dom 0}*. The *{A dom 0}* fault is detected by a test that detects *A* stuck-at-0 and simultaneously sets *B* to 0. The other three components of a 4-way bridging fault and the tests to detect them are defined in a similar manner [20].

We next discuss the observation that is the basis for the method proposed to generate tests with enhanced coverage of un-modeled bridges.

Given a test set *T*, let the *1-probability* (*0-probabilty*) of a signal line *S* be the ratio of the number of times *S* is set to 1 (0) in the fault free circuit when *T* is applied and the number of tests in *T*. Let P_{S1} (P_{S0}) denote the 1-probability (0-probability) of *S*. Consider a bridge between nodes *A* and *B* and assume that nodes *A* and *B* are uncorrelated. The bridge may be detected by a test that detects *A* stuck-at-0 and sets *B* to 0. The probability of accidentally detecting the bridge by *n* different tests that detect *A* stuck-at-0 is $1-(1-P_{B0})^n$. In Figure 2 we plot the probability of detecting the bridge fault versus the number of times the *A* stuck-at-0 fault is detected for four different values of P_{B0}.

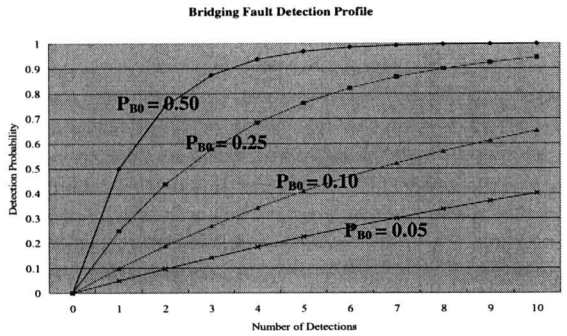

Figure 2. Probability of detecting a bridge by N-detect tests [18]

The plots in Figure 2 suggest that for a fixed value of *n* for *n*-detect test sets the probability of accidental detections of bridge defects can be increased significantly by increasing the probability of occurrence of the appropriate signal values on one of the pair of nodes involved in a bridge when the tests for modeled faults are applied.

Typically only a very small portion of the circuit inputs are specified in tests for modeled faults. Thus it is feasible to specify some of the unspecified circuit inputs to set circuit lines to desired values. This can be used to increase the probabilities of signal lines with low signal probabilities, to increase the probability of accidental detection of un-modeled bridges. In the sequel, for the sake of simplicity, we use 4-way bridge faults in describing the proposed method and in evaluating its effectiveness.

3. Defect aware ATPG

Details of the proposed method to increase the coverage of un-modeled defects are given next.

3.1. Signal probability enhancing cubes

The *signal probability enhancing (SPE) cubes* of a node are minimally specified input vectors which can set a given node to a desired logic value.

We use SPE cubes to increase the 1-probablity or 0-probablity of signal lines with signal probabilities below a threshold. During test generation, after a test *t* is generated we embed SPE cubes of selected nodes into *t* by setting the unspecified values in *t* to the specified values in the SPE cubes of the selected nodes. For example, let *t* = X0X110XXXX. We can embed the SPE cube X0XXX01XXX into *t* and obtain a modified test X0X1101XXX. We can embed another SPE cube 1X01XXXXXX into the modified test to obtain the newly modified test 1001101XXX. The remaining unspecified values in the modified test can be randomly filled before fault simulating the test.

SPE cubes for a node can be obtained by performing backward implications with the node value set to a desired value 0 or 1. In order to reduce the impact on the randomness of the generated test patterns with respect to un-modeled defects, several different SPE cubes for a node can be generated and randomly selected from the generated cubes for embedding into tests. In this work, upto three SPE cubes were generated and stored for each target node.

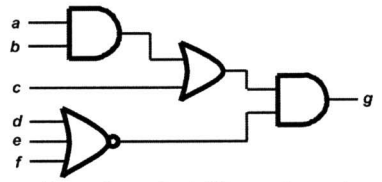

Figure 3. Signal probability enhancing cube example

The nodes targeted for enhancing signal probabilities normally have strongly biased signal probability when the inputs are randomly specified. For example in Figure 3, node *g* has low 1-probability value (5/64) if inputs (*a,b,c,d,e,f*) are randomly specified. The input vector <XX1000> can set node *g* to 1, and is considered as a SPE cube setting for *g* to 1. <11X000> is another SPE cube of *g* to set it to 1. If the circuit of Figure 3 is part of a larger circuit, the low value of 1-probability of *g* can be improved if one of these cubes is applied often during test generation for the circuit.

3.2. The proposed ATPG algorithm

The proposed ATPG uses SPE cubes to improve the overall signal probability profile for the circuit under test during ATPG and therefore improving the bridging fault coverage and test quality.

Since it may be too time consuming to compute the signal probabilities for all circuit nodes as new patterns are generated, only the signal probabilities of the fanout

stems which are outputs of fanout-free regions are computed in our work. These nodes together with a signal value are referred to as *potential signal probability enhancement targets* and are placed in a buffer *Buf1* during a preprocessing step.

During the generation of tests, first a conventional ATPG is run until a preset number of N patterns are generated. During this phase of the APTG *all* unspecified values in the generated tests are filled randomly. The signal probabilities of signals in Buf1 are initially computed from the signal value statistics collected for the first N test patterns. N was set to 512 in our implementation. After the generation of the first N test patterns, the proposed SPE technique is turned on and signal probability enhancement is performed by embedding SPE cubes for selected signals into each newly generated test pattern. After embedding SPE cubes, the remaining unspecified values in the modified tests are randomly filled.

All the potential SPE targets in Buf1 with 0-probability or 1-probability value less than a predefined threshold (0.10 in our implementation) are considered as *SPE candidates* and put into a second buffer *Buf2*. A subset of the SPE candidates with the lowest signal probabilities are selected as *SPE targets* and put into a third buffer *Buf3*. In our implementation we allowed Buf3 to contain upto 1000 SPE targets ordered according to increasing values of their signal probabilities. We attempt to embed one randomly selected SPE cube of each SPE target in Buf3 into a partially specified newly generated test pattern. The SPE targets in Buf3 are updated after each set of 32 new test patterns. This is done by first updating the signal probabilities of the SPE candidates in Buf2 after 32 new test patterns are simulated and then moving upto 1000 signal nodes with the lowest probabilities into Buf3. Similarly, we update the signal probabilities of the potential SPE targets in Buf1 after every 128 new tests are simulated and the SPE candidates in Buf2 are updated by moving the potential SPE targets in Buf1 with signal probabilities less than 0.1 into Buf2.

Up to three SPE cubes are generated only for signals in Buf3. Whenever a signal line is moved into Buf3 we generate SPE cubes only if they were not generated earlier for this signal line.

3.3. Signal probability enhancement maintaining test data compression

When test data volumes are to be reduced by compression techniques such as [17,23] the unspecified values in the tests are filled by the decompressor used. Thus filling unspecified values in tests by embedding SPE cubes may dramatically reduce the achievable test data volume compression. For this reason one should fill only a very limited number of unspecified bits in tests by embedding SPE cubes. For test data compression technique such as [17], one can compute the number of unspecified values that can be filled by embedding SPE cubes. We briefly describe this next.

For a design with *NFF* scan cells, let the targeted input test data compression ratio be *CR* and let the encoding efficiency of the compression scheme be η. Then the allowed maximum number of specified bits in a test is η*NFF/CR*. Given a newly generated test we can allow the unspecified bits to be specified by embedding SPE cubes until the total number of specified bits in the test is less than or equal to η*NFF/CR*.

4. Modified bridging coverage estimate

Quality of tests is typically evaluated by fault coverage metrics. Fault coverage metrics however do not provide a measure of the coverage of actual defects. For this reason, several methods have been proposed in the literature to estimate defect coverage by given test sets. One method is to use fault coverage of un-modeled faults (called surrogate defects) as a measure of defect coverage [8,15,16]. Several methods that do not use simulation of surrogate defects to estimate defect coverage have also been proposed [4,5,8]. A recently proposed measure called *bridge coverage estimate* (BCE) [4] is simple to calculate and was shown to estimate incremental defect coverage by n-detect test sets as measured by the defective parts of an ASIC design detected by tests with increasing numbers of detection of SSL faults. This measure of quality may not give an accurate estimate of the coverage of un-modeled defects but it can be used to compare relative effectiveness of defect coverage by two different test sets. Another use of this measure could be to use the increment in the value of the measure to estimate the additional defect coverage obtained by, for example, n-detect tests as additional tests are generated to increase the number of detections of SSL faults. This will provide a basis to select the value of n for n-detect tests. However it has been noted that BCE may saturate early for some designs and hence may not provide a good estimate of incremental defect coverage by n-detection tests as n is increased [2]. We propose a modification to this measure. Experimental results presented in the next section show that the modified measure gives a better estimate of the incremental defect coverage of test sets.

Definition 1 [4]: Given a test set T and a target fault list F of SSL faults, the bridging coverage estimate (*BCE*) is calculated as follows:

$$BCE = \sum_{i=1}^{n} \frac{f_i}{|F|} \bullet (1 - 2^{-i})$$

where f_i is the number of stuck-at faults detected i times by T, $|F|$ is the total number of stuck-at faults in the target fault list F, and n is the maximum number of detections that a fault can be detected by T. In practice, n is limited by the maximum number of detections that the fault simulation tool keeps track of. The underlying assumption in computing BCE is that the aggressor node of a bridging fault has 50% probability of being in the state to activate the bridge defect when the appropriate stuck-at fault at the victim node is detected.

Consider a bridging fault *{A dom 0}* between nodes A and B. For a given test set, assuming the 0-probability

value of B is p and the stuck-at fault (A *sa0*) is detected by n different test patterns, the probability to detect this bridging fault by this test set will actually be $1-(1-p)^n$, instead of $(1-2^{-i})$ as used in the calculation of BCE. To overcome this inaccuracy of BCE a modified bridge coverage estimate, called BCE^+ is proposed, which combines multiple detection profile of SSL faults and the circuit signal probability profile for a given set of tests.

Definition 2: Given a test set T, a set of signal nodes S and a target fault list F, the *bridging coverage estimate* BCE^+ is calculated as follows:

$$BCE^+ = \sum_{i=1}^{n} \frac{f_i^{sa0}}{|F|} \bullet \{\sum_{j=1}^{|S|} \frac{1}{|S|}(1-(1-p_{j0})^i)\}$$
$$+ \sum_{i=1}^{n} \frac{f_i^{sa1}}{|F|} \bullet \{\sum_{j=1}^{|S|} \frac{1}{|S|}(1-(1-p_{j1})^i)\}$$

where f_i^{sa0} (f_i^{sa1}) is the number of *sa0* (*sa1*) faults detected i times by test set T, p_{j0} (p_{j1}) is the 0-probabiltiy (1-probalility) of signal j in the subset S of the circuit signals considered and $|S|$ is the cardinality of the set S.

5. Experimental results

We implemented the proposed defect aware ATPG procedures by incorporating them into a state of the art commercial ATPG. The modified ATPG is referred to as SPE_ATPG.

In the first experiment we performed, for each one of four circuits studied we generated two sets of n-detect tests for SSL faults using the original ATPG and the SPE_ATPG for n = 1 to 10. Both test sets were then fault simulated on 100K randomly selected bridges using 4-way bridging fault model. The 100K bridges result in 400K 4-way bridging faults. The results are given in Table 1 on the next page. In Table 1 we first give the circuit name, the number of gates *#G* and the number of scan cells *#SC* in the circuit. In the following columns we report the number of tests *#TP*, the run times *RT* and the percent of the 400K 4-way bridging faults detected by the test sets. The column with the heading *Dn* includes the data for n-detect test sets, n =1 to 10. The data for the ten pairs of n-detect test sets are arranged in two consecutive rows, with the data for the original ATPG in the first row and the data for the SPE_ATPG in the second row. The run times reported are normalized run times obtained by dividing the run times by the time to generate the standard single detection tests for SSL faults using the original ATPG. It can be seen that for all circuits every n-detect test set obtained by SPE_ ATPG achieves higher bridging fault coverage than the corresponding test sets obtained by the original ATPG. The higher bridging fault coverage is obtained with minimal increase in run time and test pattern count. On the average, the *1*-detect test sets of SPE_ATPG have 0.1% more patterns than the *1*-detect test sets of the original ATPG and the pattern count increase is 0.8% for *10*-detect test sets. The average increase in run time for SPE_ATPG is 8.8% for *1*-detect tests and it is 7.4% for *10*-detect tests. The average increase in bridging fault coverage (BFC) obtained by using SPE technique is 1.05% for *1*-detect test sets and 0.81% for *10*-detect test sets.

The following additional observation can be made from the data given in Table 1. Using SPE techniques one can derive higher quality tests that can achieve the same or higher un-modeled defect coverage using smaller run times and pattern counts. For example the *10*-detct test set for circuit C1 generated by using the original ATPG has 34,151 tests and a relative run time of 7.37. These tests achieve bridging fault coverage of 95.06. However using SPE techniques one can achieve 95.53 bridging fault coverage by *3*-detect tests which contain 11,692 tests and use a relative run time of 2.19.

The next experiment we performed was to validate the use of techniques such as SPE to improve the coverage of un-modeled defects. The data in Table 1 shows that the SPE based test sets achieve higher bridge defect coverage. The data given in Table 2 shows that *1*-detect tests for SSL faults generated by the original ATPG and the SPE_ATPG both detect essentially all interconnect open faults. This data was obtained by simulating the *1*-detect tests on interconnect opens modeled by multiple line stuck-at faults. To perform this experiment we incorporated into the ATPG the procedures in [19]. An interconnect open at a circuit node with a fan-out of k is modeled by $(2^{k+1} - 2)$ multiple stuck-at faults [19]. Implicit simulation procedures given in [19] permit efficiently determining the multiple stuck-at faults detected by a given test set. Since simulation procedures cannot determine the un-testable faults we also modified the ATPG to determine if the multiple stuck-at faults remaining after simulating *1*-detect test sets are detectable. In Table 2, after the circuit name, we give the number of multiple stuck-at faults corresponding to all possible single interconnect open faults. Next we give the number of test patterns (#TP) and the number of multiple stuck-at faults that are not detected (#UndOF) by the *1*-detect test sets derived by the original ATPG and the SPE_ATPG. The number of undetected faults given includes the faults that can be detected if additional tests are generated by an ATPG modified to generate tests for multiple-stuck faults and the faults aborted by the ATPG. It can be seen that a negligible portion of multiple stuck-at faults that model interconnect open defects are not detected by standard SSL test sets.

Table 2. Coverage for interconnect open defects

Ckt	#TotOF	Orig_D1		SPE_D1	
		#TP	#UndOF	#TP	#UndOF
C1	7.970E8	5400	18	5352	8
C2	1.198E9	2252	465	2296	1799
C3	2.246E9	2126	327	2101	929
C4	2.700E9	7937	1228	7985	1770

The third experiment we performed was to evaluate the relative quality of tests generated using SPE_ATPG when we restrict the total number of specified bits in tests to be less than or equal to a maximum value. For this experiment we assumed the

Table 1. Conventional n-detect test sets vs. SPE n-detect test sets

Ckt		D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
C1	#TP	5400	8311	11591	14870	18191	21411	24744	27716	31104	34151
		5352	8229	11692	15028	18245	21456	24777	27970	31180	34165
	RT	1.00	1.47	2.03	2.68	3.51	3.95	4.69	5.31	6.49	7.37
#G=344K		1.09	1.58	2.19	2.87	3.51	4.42	4.99	5.72	6.53	7.31
#SC=20K	BFC	91.05	92.53	93.38	93.90	94.25	94.49	94.66	94.86	94.98	95.06
		93.11	94.70	95.53	96.05	96.37	96.62	96.75	96.86	96.96	97.01
C2	#TP	2252	3597	5092	6600	7964	9292	10768	12145	13649	15119
		2296	3663	5206	6735	8157	9494	10961	12418	13847	15550
	RT	1.00	1.43	1.99	2.67	3.42	4.19	4.91	5.73	6.52	7.23
#G=543K		1.06	1.55	2.12	2.81	3.58	4.42	5.20	6.00	6.78	8.72
#SC=45K	BFC	95.03	96.35	97.02	97.38	97.66	97.85	98.01	98.13	98.22	98.26
		95.40	96.72	97.38	97.72	97.98	98.14	98.29	98.40	98.48	98.53
C3	#TP	2126	2694	3707	4536	5380	6249	7071	7946	8753	9669
		2101	2669	3731	4557	5423	6327	7072	8034	8873	9733
	RT	1.00	1.35	1.88	2.41	3.15	3.73	4.46	5.26	6.04	6.54
#G=1.21M		1.10	1.56	2.22	2.67	3.32	3.95	4.61	5.30	6.30	6.88
#SC=70K	BFC	92.80	94.44	95.43	95.98	96.34	96.57	96.77	96.92	97.04	97.15
		94.17	95.66	96.53	96.95	97.31	97.51	97.69	97.81	97.89	98.01
C4	#TP	7937	11156	15399	19492	23254	27212	30959	35304	39299	43253
		7985	11419	15643	19706	23592	27487	31498	35706	39946	43632
	RT	1.00	1.20	1.63	2.13	2.73	3.38	4.08	4.74	5.42	6.10
#G=1.21M		1.13	1.35	1.84	2.33	2.95	3.59	4.24	4.94	5.61	6.35
#SC=70K	BFC	93.88	94.61	95.11	95.38	95.59	95.77	95.89	96.00	96.08	96.18
		94.26	94.89	95.35	95.63	95.82	95.98	96.09	96.17	96.27	96.34

test data compression technique of [17] with a target compression ratio of 32X and encoding efficiency of 0.95. This restricts the number of specified bits to be less than 3% of the number of circuit scan cells. Using this limit on the number of specified bits in tests we generated *1*-detection test sets with and without using the SPE techniques. The results of this experiment are given in Table 3. After the circuit name we give the number of test patterns, the SSL fault coverage (SAFC) and the fault coverage of 400K randomly selected 4-way bridging faults (BFC) for the original ATPG and the SPE_ATPG. From Table 3 it can be seen that compared to the original ATPG tests the SPE_ATPG tests achieve higher bridging fault coverage and the same SSL fault coverage. For three of the four circuits the pattern counts are less for the SPE based tests. These results show that SPE techniques can be used to improve the ability of tests to detect un-modeled defects without reducing achievable test data compression.

Table 3. Defect aware ATPG under EDT (C.R.=32X)

Ckt	Orig_D1_32X			SPE_D1_32X		
	#TP	SAFC	BFC	#TP	SAFC	BFC
C1	5757	97.13	90.95	5860	97.13	92.74
C2	2755	98.38	94.97	2663	98.38	95.53
C3	2502	98.74	92.61	2475	98.74	93.74
C4	9116	98.66	93.82	8857	98.66	94.20

The fourth experiment we performed was to investigate the effect of SPE techniques when the faults targeted for test generation may include SSL faults as well as bridging faults extracted from the circuit layout. For circuit C1 for which we have layout we extracted 800K realistic bridges. Using the extracted bridges and SSL faults we generated tests with and without using SPE techniques in the following way. First *1*-detect test set was generated which was augmented to detect realistic bridges. Next we augmented these test sets to *3*-detect all SSL faults. These four test sets were fault simulated on additional 400K randomly selected 4-way bridging faults. The results of this experiment are reported in Table 4. The coverage of realistic bridges are reported under the columns BFC^{Ex} and the coverage of randomly selected bridges are shown under column BFC^{Rd}. BFE^{Ex} gives the fault efficiency computed as the percentage of detected faults out of the detectable faults obtained by subtracting the fault proven to be un-testable by the ATPG. It can be seen that all test sets essentially achieve comparable coverage of realistic bridging faults since they were targeted by ATPGs. However the SPE based test sets achieve higher coverage of un-modeled (un-targeted) bridging faults.

Table 4. SPE hybrid ATPG for design C1

Test Set	#TP	BFC^{Ex}	BFE^{Ex}	BFC^{Rd}
Orig_D1H	13278	90.17	99.75	93.64
SPE_D1H	12431	90.18	99.76	95.28
Orig_D3H	17441	90.17	99.75	94.44
SPE_D3H	16353	90.18	99.76	96.20

The last set of data we present is related to the bridge coverage estimate measures BCE and BCE^{+}. An objective of such measures is to determine the incremental defect coverage obtained when additional tests are added to a given test set. In Figures 4(a) through 4(d) we plot the ratio of increase in bridging fault coverage (ΔBFC) and the increase in BCE (ΔBCE) and BCE^{+} (ΔBCE^{+}) as the value of n in n-detect tests is increased. When BCE saturates the ratio ΔBFC/ΔBCE

could be very large and for this reason the vertical axis in Figures 4(a) – 4(d) uses logarithmic scale. From these plots we see that BCE^+ provides a better estimate of the increase in un-modeled defect coverage as additional tests are added as noted by the fact that the plots for $\Delta BFC/\Delta BCE^+$ are flat while the plots for $\Delta BFC/\Delta BCE$ rise steeply which indicates BCE saturation.

6. Conclusions

In this work we proposed a method to generate tests to obtain high quality tests for manufacture test. The proposed method enhances the probability of desired signal values to increase accidental detections of static defects. Results on several industrial circuits show that the proposed method can be readily incorporated in to an existing ATPG and the resulting test sets achieve higher un-modeled defect coverage. The increase in defect coverage is obtained with minimal impact on run time, pattern count and achievable test date compression ratios. A new measure to estimate test quality, called BCE^+, which is shown to give a good estimate of incremental defect coverage was also proposed.

In the future methods to increase accidental detection of defects such as open transistors, resistive opens and certain high resistance bridges which effect circuit delays will be investigated.

References

1. M. Abramovici and M. Breuer, "A practical approach to fault simulation and test generation for bridging fault," *IEEE Trans. Computers*, vol.34, no.7, 1985, pp. 658-663.
2. E. M. Amyeen et al., "Evaluation of the quality of N-detect scan ATPG patterns on a processor," in *Proc. ITC*, 2004, pp. 669-678.
3. K. Baker et al., "Defect-based delay testing of resistive vias-contacts, a critical evaluation," in *Proc. ITC*, 1999, pp. 467-476.
4. B. Benware et al., "Impact of multiple-detect test patterns on produce quality," in *Proc. ITC*, 2003, pp. 1031-1040.
5. R. D. Blanton, K. N. Dwarakanath and A. B. Shah, "Analyzing the effectiveness of multiple-detect test sets," in *Proc. ITC*, 2003, pp. 876-885.
6. S. Chakravarty and A. Jain, "Fault models for speed failures caused by bridges and opens," in *Proc. VTS*, 2002, pp. 373-378
7. S. Chakravarty et al., "Experimental evaluation of scan tests for bridges," in *Proc. ITC*, 2002, pp. 509-518.
8. J. Dworak et al., "Defect-oriented testing and defective-part-level prediction," in *Design and Test of Computers*, vol.18, no.1, 2001, pp. 31-41.
9. P. Engelke et al., "Automatic test pattern generation for resistive bridging fautls," in *Proc. ETS*, 2004.
10. H. Konuk, "Voltage- and current-based fault simulation for interconnect open defects," *IEEE Trans. CAD of IC & Sys.*, vol.18, no.12, 1999, pp. 1768-1779.
11. S. C. Ma, P. Franco, and E. J. McCluskey, "An experimental test chip to evaluate test techniques experimental results," in *Proc. ITC*, 1995, pp. 663-672.
12. S. Ma, I. Shaik and R. S. Fetherston, "A comparison of bridging fault simulation methods," in *Proc. ITC*, 1999, pp. 587-595.
13. P. C. Maxwell and R. C. Aitken, "Biased voting: a method for simulating CMOS bridging faults in the presence of variable gate logic thresholds," in *Proc. ITC*, 1993, pp. 63-72.
14. S. D. Millman and S.J. P. Garvey, "An accurate bridging fault test pattern generator," in *Proc. ITC*, 1991, pp. 411-418.
15. I. Pomeranz and S.M.Reddy, "Stuck-at tuple-detection: a fault model based on stuck-at faults for improved defect coverage", in *Proc. VTS*, 1998, pp. 289-294.
16. I. Pomeranz and S. M. Reddy, "Definitions of the numbers of detections of target faults and their effectiveness in guiding test generation for high defect coverage," in *Proc. DATE*, 2001, pp. 504-508.
17. J. Rajski, et al,, "Embedded deterministic test for low cost manufacturing test," in *Proc. ITC*, 2002, pp. 301-310.
18. J. Rajski, "Key note speech", in *ITC*, 2003.
19. S. M. Reddy et al., "On testing of interconnect open defects in combinational logic circuits with stems of large fanout," in *Proc. ITC*, 2002, pp. 83-89.
20. S. Sengupta et al., "Defect-based tests: a key enabler for successful migration to structural test," *Intel Technology Journal*, Q.1, 1999.
21. T. M. Storey and W. Maly, "CMOS bridging faults detection", in *Proc. ITC*, 1990, pp. 842-851.
22. L. C. Wang, K. M. Butler and M. R. Mercer, "On efficiently and reliably achieving low defective part levels," in *Proc. ITC*, 1995, pp. 616-625.
23. P. Wohl, et al., "X-tolerant compression and application of scan-ATPG patterns in a BIST architecture," in *Proc. ITC*, 2003, pp.727-736.

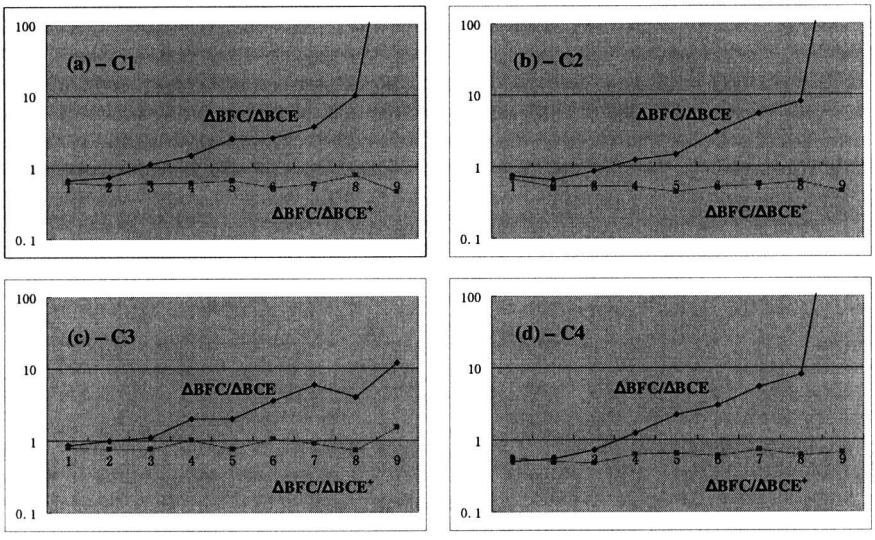

Figure 4. Plots of ΔBFC/ΔBCE and ΔBFC/ΔBCE+