SIMULATION-BASED APPROXIMATE GLOBAL FAULT COLLAPSING

Hussain Al-Asaad and Raymond Lee
Computer Engineering Research Laboratory
Department of Electrical & Computer Engineering
University of California
One Shields Avenue, Davis, CA 95616-5294

ABSTRACT

To generate tests for a digital circuit, the test generation tool is initially provided with the circuit description in a netlist format and then it creates a list of faults that need to be targeted for detection. For large circuits, the number of faults can become very large. It is thus beneficial to minimize the number of faults whenever possible. Fault collapsing is the process of reducing the number of faults by using equivalence and dominance relationships among faults. Exact fault collapsing can be easily applied locally at the logic gates, however, it is not feasible to apply it globally for large circuits. In this paper, we present an approximate global fault collapsing technique that is based on the simulation of random vectors. Experimental results show that our method reduces the number of faults drastically with feasible resources.

Keywords: *Global fault collapsing, simulation, physical fault testing.*

1 INTRODUCTION

To test a digital circuit, an automatic test pattern generation (ATPG) tool generates a test set that targets possible physical faults. As the complexity of the digital circuit increases, the possible number of physical faults increases that consequently leads to a significant slow down of the test generation process using the ATPG tool. One approach for considerably reducing the length of the testing process is fault collapsing. Fault collapsing [1] is the process of reducing the number of faults by using equivalence and dominance relationships among faults. Exact fault collapsing can be easily applied locally at the logic gates, however, it is not feasible to apply it globally for large circuits.

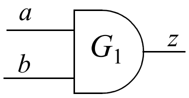
In this paper, we develop an approximate global fault collapsing technique that is based on the simulation of random vectors. We review fault collapsing in Section 2 and present our method in Section 3 and describe some of its implementation details. We present some experimental results in Section 4 and finish with a summary.

2 FAULT COLLAPSING

In physical fault testing, physical defects are abstracted into a logical fault model. The most widely-used logical fault model is the Single Stuck-Line (SSL) model [1]. Under this model, every single signal line can become permanently fixed (stuck) at a logical 1 or 0 value. The model is simple and technology-independent. It represents a large number of different physical faults, and tests derived for SSL faults detect many design errors/faults. In this paper, we only consider SSL faults, however, our method is applicable to several other fault models.

Fault collapsing reduces the number of faults using two relationships among faults: fault equivalence and fault dominance. Two faults are considered equivalent if the faulty functions produced by the two faults are equal. Alternatively, the two faults are equivalent if they can be detected by the same tests. In this case, there is no way to distinguish between the two faults. For example, the SSL fault *a* stuck-at 0 represented by *a*/0 in Figure 1 is equivalent to the fault *z*/0. If two faults are equivalent then one of the faults can be dropped from the fault list since the detection of the other fault guarantees the detection of the dropped fault.

A fault *f* is considered to dominate another fault *g* when every test for *g* is also a test for *f*. For example, the fault *z*/1 dominates the fault *a*/1 in Figure 1 since



Inputs		Correct	Faulty functions					
a	b	function (z)	$a/0$	$a/1$	$b/0$	$b/1$	$z/0$	$z/1$
0	0	0	0	0	0	0	0	1
0	1	0	0	1	0	0	0	1
1	0	0	0	0	0	1	0	1
1	1	1	0	1	0	1	0	1

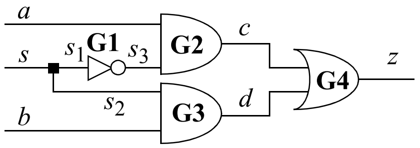
Figure 1 Fault collapsing for a 2-input AND gate.

the only test vector 01 for $a/1$ (shaded in the figure) is also a test for $z/1$. If a fault f dominates a fault g, then the fault f can be dropped from the fault list since the detection of g guarantees the detection of f.

By applying fault collapsing to the AND gate in Figure 1, we can reduce the number of faults from six to three. The faults $a/0$ and $b/0$ are dropped since they are equivalent to $z/0$. Moreover, the fault $z/1$ is dropped since it dominates both $a/1$ and $b/1$. The collapsed fault list is thus $\{a/1, b/1, z/0\}$. A test set that detects the faults in the collapsed list can be derived from the table in Figure 1 as {01,10,11}. This test detects all faults in the collapsed fault list and consequently all six faults in the AND gate.

There are two approaches to fault collapsing: local and global. The local fault collapsing method computes the collapsed fault list for individual gates and then collects the collapsed fault lists for the gates to form the overall collapsed fault list for circuit. For example, by using fault collapsing over the gates in the circuit shown in Figure 2, we get the results shown in the figure. Both stuck at faults on line s (called a stem since it branches to other lines) need to be considered because s is not an input or output of any gate. Using local fault collapsing, we combine the faults on the gates to form the collapsed fault list for the circuit as $\{s/0, s/1, s_3/0, s_3/1, a/1, b/1, s_2/1, c/0, d/0, z/1\}$. Therefore, by using local fault collapsing we were able to reduce the fault list from 18 to 10.

Global fault collapsing is similar to local fault collapsing, except that we perform the same process of fault collapsing on the entire circuit as opposed to individual gates. In other words, we look for equivalent and dominance relationships among all faults in the circuit. For example, to perform global fault collapsing for the circuit in Figure 2, we compute a table for all faulty functions (called a fault table) as shown in Figure 2. It is simpler to start with the local col-



Gate G1: $\{s_3/0, s_3/1\}$
Gate G2: $\{a/1, s_3/1, c/0\}$
Gate G3: $\{b/1, s_2/1, d/0\}$
Gate G4: $\{c/0, d/0, z/1\}$
Stem s: $\{s/0, s/1\}$

Inputs			Correct	Faulty functions									
s	a	b	output z	$s/0$	$s/1$	$s_3/0$	$s_3/1$	$a/1$	$b/1$	$s_2/1$	$c/0$	$d/0$	$z/1$
0	0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	1	0	0	1	0	0	1	0	1	0	0	1
0	1	0	1	1	0	0	1	1	1	1	0	1	1
0	1	1	1	1	1	0	1	1	1	1	0	1	1
1	0	0	0	0	0	0	0	0	1	0	0	0	1
1	0	1	1	0	1	1	1	1	1	1	1	0	1
1	1	0	0	1	0	0	1	0	1	0	0	0	1
1	1	1	1	1	1	1	1	1	1	1	1	0	1

Figure 2 A simple multiplexer circuit with a list of its gate faults and the resulting fault table.

lapsed fault list since it has less faults than the original fault list for the circuit. We then drop faults from the local collapsed fault list using equivalence and dominance relationships. The faults $s/0$, $b/1$, $z/1$ are dropped since they dominate $s_3/1$. Also, the faults $s/1$, $a/1$ can are dropped since they dominate $s_2/1$. The fault $s_3/0$ is dropped since it is equivalent to $c/0$. The global collapsed fault list for the circuit is thus $\{s_2/1, s_3/1, c/0, d/0\}$. Hence, by using global fault collapsing we were able to reduce the number of faults from 18 to 4. This is in effect a 77.78% reduction from the original fault list.

Local fault collapsing can be easily scaled to large circuits. However, global fault collapsing cannot be feasibly done for large circuits. This is the case due to the expensive computations and memory needed to determine equivalence and dominance relationships among the faults in the overall circuit. In the next section, we describe a method that determines an approximate global collapsed fault list.

3 Approximate Global Fault Collapsing

In approximate global fault collapsing, a large set of random vectors is used to reduce the number of faults instead of using the complete vector set for the circuit. The idea behind approximate collapsing is that the resulting faults after the simulation is an approximation of the faults from exact global fault collapsing of the circuit. As more and more vectors are applied for the simulation, the results appear more and more similar to those of the exact global fault collapsing.

Every fault in the circuit is classified into one of the following types: redundant (RD), equivalent (EQ), dominating (DM), and remaining (RM). A fault is classified as RD if no test exists for it. In other words, a redundant faulty function is the same as the correct function. A fault is classified as EQ if it is equivalent to at least one RM fault. A fault is classified as DM if it dominates at least one RM fault. A circuit global collapsed fault list is the set of all RM faults.

Our approximate global fault collapsing method works as follows. We first label all the faults in the circuit as RD faults since we have no information about the detectability of the faults. We then apply a single random vector and determine the faults detected by it and then update the type of every fault. The process is repeated for several iterations until no change is reported in the types of faults.

The type of a fault can change to EQ, DM, or RM as the random vectors are applied. However, no fault will change its type to RD as the simulation advances. Since, once a fault changes from RD to EQ, DM, or RM, then the fault is detected by at least one vector. Hence, it is not possible for the fault to change its type back to RD. So, the number of RD faults decreases as more random vectors are applied.

Figure 3 depicts the number of the different types of faults as the simulation progresses for the circuit c432. The circuit c432 is one of the ISCAS-85 [2] benchmark circuits. The local collapsed fault list for c432 is 524 faults. In the beginning they are all marked as being redundant since none of the faults are detected. After the application of 50 random vectors, the number of redundant faults drops significantly to 75 faults initiating a trend that continues until the end of the simulation: the steady increase in the number of remaining faults and the decrease in equivalent and redundant faults. Eventually, if every possible vector were applied, the number of redundant faults would gradually converge to the exact number of redundant faults.

Our preliminary approximate global fault collapsing tool benefited from the use of existing CAD tools. For example, to compute the local fault collapsed list for the ISCAS-85 circuits, we used FSIM [3], which is a parallel pattern single fault simulator for combinational circuits. Our tool is written in several PERL scripts. The scripts communicated with each other through the use of coma delimited text files. The parent script was written in UNIX to take advantage of the many native system calls that would be made,

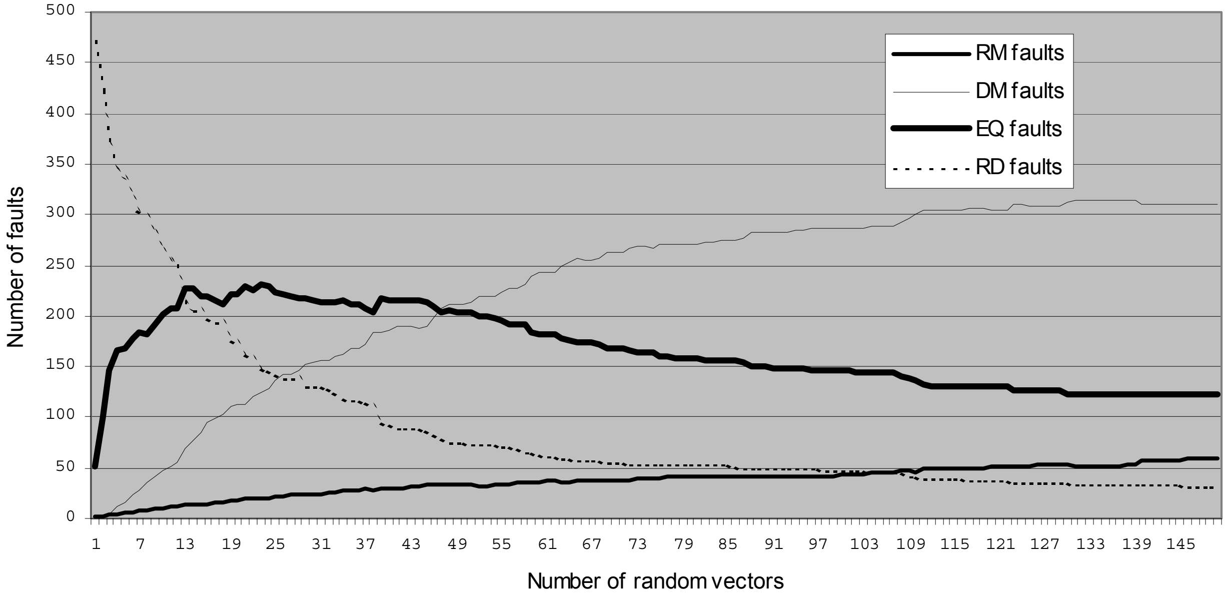


Figure 3 Approximate global fault collapsing for the circuit c432.

such as invoking FSIM and managing the countless intermediate files. The primary reason PERL was chosen for the main script is because of the broad tools it provided for parsing files and its flexibility in manipulating character strings: precisely the type of data generated by the fault simulators. Moreover, arrays in PERL are dynamic and simple to manipulate.

To begin with, a two dimensional array was constructed with rows representing vectors and columns representing faults which grew as the simulation ran. The main PERL script reads in each line of a vector file containing nothing more than unique vectors of the appropriate length. The script terminates when the final vector has been processed. For small circuits, every possible vector is enumerated. However, as the number of inputs grew, the vector list tended to explode exponentially in size. The first circuit (c17) having only five inputs simply required 32 vectors. The larger circuits, such as c499, contained 41 inputs, resulting in 2^{41} possible vectors. In these cases, large numbers of unique random vectors are applied.

In initiating the global fault collapsing, FSIM first generates a file containing all the possible faults. As the main PERL script reads in each vector from the vector file, it calls FSIM to create a file containing all the stuck at faults not detected by that vector. A parsing script compares the file against the one listing all the possible faults and creates a fault file with the faults detected by that vector. This file follows the same format as the one created by FSIM. As we progress in the simulation, the number of RD faults will decrease substantially with the continued addition of vectors. Although the number of EQ faults initially increases, it later decreases as the faults adopt unique identities over time.

Our first approximate global fault collapsing tool turned out to be resource intensive and memory hungry process. Even with only 1,000 test vectors, many of the smaller benchmark circuits required several hours to simulate. To enhance our tool, several modifications were made to the scripts. First of all, arrays were no longer used and were replaced by binary strings of 1's and 0's. Each fault was given its own detection string. As the simulation progressed, a 1 or a 0 was concatenated to each string depending on whether the fault was detected or not. Comparisons for dominance and equivalence were quickly checked using a combination of logic functions and native PERL string functions instead of using loops. Another optimization was to check whether the string contains all zeros. If so then the current fault can be immediately concluded to be of type RD. At last, if the fault is neither EQ nor DM, then it is a RM fault.

Even before any two faults are compared, the new script runs an assortment of checks to determine if indeed a comparison is necessary. The goal is to avoid a brute force comparison at any cost. If a fault is of type RD and is not detected by the current vector, then it remains an RD fault. If the fault is an RM fault and the current vector does not detect it, then it stays an RM fault. If two faults are considered equivalent, and their current detection values are equal, then they retain their current values.

4 Experiments and Results

The simulations were run with a large number of vectors beginning with a vector set of 1,000. Our optimized script resulted in a considerably large speed up in simulation time and allowed us to increase the number of vectors up to at least 55,000 or until the types of faults showed little or no change from one iteration to the next. Figure 4 and Figure 5 depict the initial progress of our approximate global fault collapsing method for the circuits c880 and c1908, respectively, as we add more random vectors. The circuits c880 and c1908 are a subset of the ISCAS-85 benchmark suite [2].

Test generation for RM faults produced test sets with high coverage of all faults. For example in the 32 input circuit c432, after applying 94 random vectors, there were 42 remaining faults, 299 dominant, 44 equivalent, and 43 redundant. After generating a test set for RM faults using ATALANTA [4], we were able to detect 88.359% of all detectable faults (RM, EQ, and DM). After applying 659 random vectors, the RM faults became 114 and the RD faults became 4. After generating a test set for the 114 RM faults, a 98.664% fault coverage is achieved out of a possible 99.236%. Eventually tests for RM faults detect all detectable faults. The above experiment proves that the RM faults are an excellent approximation of the global collapsed fault list.

When circuits do become huge in size, even the process of approximate global collapsing of faults eventually becomes tedious and time consuming. A method for expediting the process of global fault collapsing is to take the middle ground between global

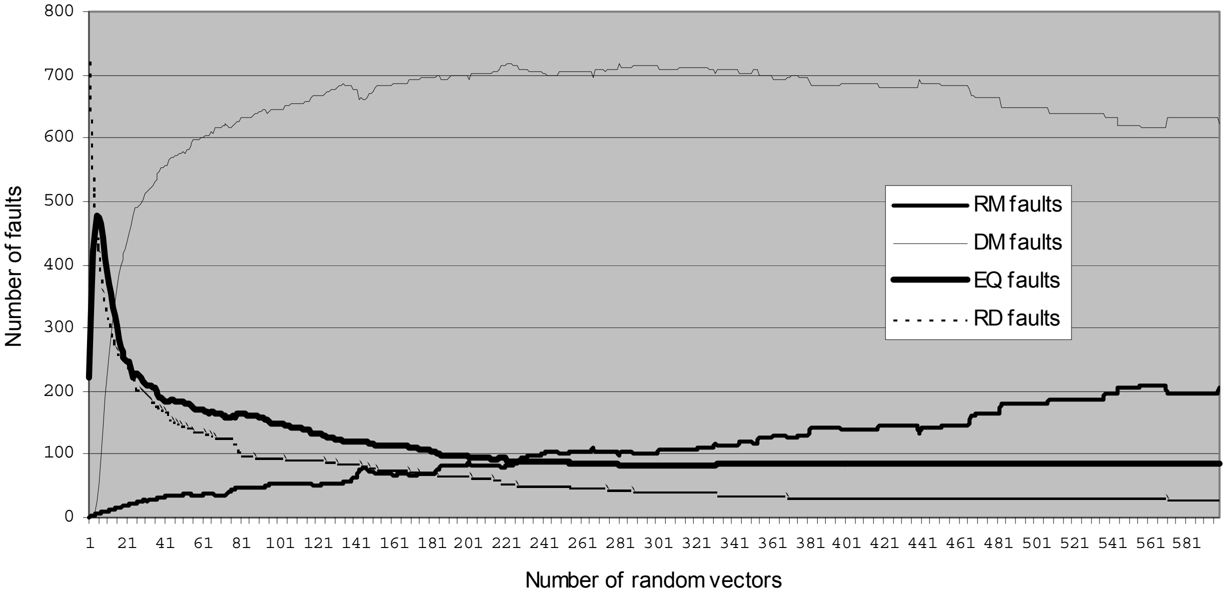


Figure 4 **Approximate global fault collapsing for the circuit c880.**

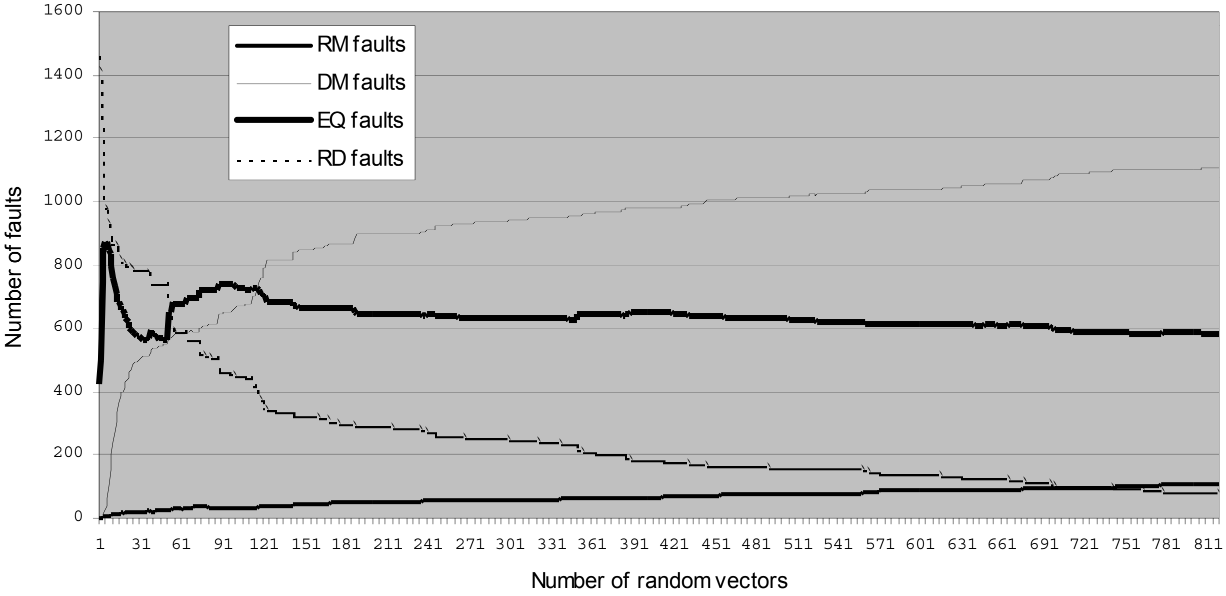


Figure 5 **Approximate global fault collapsing for the circuit c1908.**

and local fault collapsing. In this hybrid process, we take a complex circuit and partition it into smaller modular components. We then perform approximate global fault collapsing for each of the components using the scripts described earlier so that we end up with a list of the remaining faults that characterize each of the components. Once this is accomplished, we recombine the entire circuit and target all the sets of remaining faults from each of the components. The premise is that the combined remaining faults are an approximation of the globally collapsed fault list.

One benefit of this process is time. Smaller simpler circuits (components) are being simulated and they can be simulated in parallel on different computers. When the simulation of all the components are completed and their remaining faults are determined, the overall circuit can then be constructed and all of its remaining faults targeted. Moreover, a library of components with their remaining faults can be constructed so that it can be used for other designs.

An experiment was performed to evaluate the effectiveness of component fault collapsing. Three circuits of the ISCAS-85 benchmark suite were combined for this experiment (c17, c432, and c499) in the configuration shown in Figure 6. Each of the circuits was individually globally fault collapsed and the remaining faults extracted. Next, they were combined into a larger circuit c0m and the remaining faults were targeted.

The number of faults in c0m using local fault collapsing as determined by FSIM is 1236. However, the number of RM faults using approximate global fault collapsing for the components is 601. Test generation for c0m with local fault collapsing took 0.150 seconds and produced 100 vectors with an overall coverage of 98.948%. However, test generation for the RM faults in c0m took 0.083 seconds and produced 83 vectors with an overall coverage of 92.961%. So, by using global fault collapsing for the components we were able to detect most faults with a smaller test set that is generated in half the time needed to detect all local collapsed faults. Although the overall coverage of faults is less after using approximate global fault collapsing for the components, it can be made very close to the maximum coverage if more random vectors are used in determining the RM faults in the components.

Another advantage to the use of global fault collapsing for the components is that it can be customized depending on the characteristics of the target circuit. Local fault collapsing is rigidly set into using the gate level components despite the size of the circuit, so larger circuits may produce less coverage than those globally collapsed. Global fault collapsing demands that all vectors be tested to be conclusive so the price for the ideal minimal vector set is that large circuits require large amounts of time. In contrast, the number of random vectors in approximate fault collapsing can be raised to achieve a more accurate result, or lowered to speed up the process but ensure that the percentage of faults falls within a certain tolerance range. Components inside the circuit can be partitioned in different ways to achieve varying or even mixed granularity depending on the complexity of the circuit. Super fine partitioning of components basically to the extreme of each gate being one component results in local fault collapsing, while the most coarse approach of treating all components as one big circuit results in the other end of the spectrum, global fault collapsing. A mixed approach of the two can be applied to achieve degrees of accuracy, speed, and attention to detail that the process exerts.

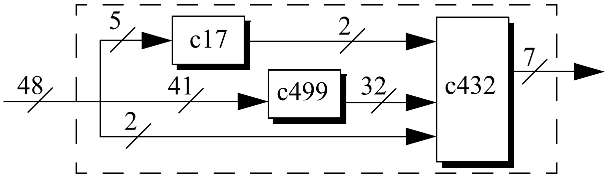


Figure 6 The circuit c0m with its c17, c432, and c499 components.

In summary, there are two methods for collapsing the faults in a circuit. Local fault collapsing is simple to implement, but does not reduce the faults as efficiently as exact global fault collapsing. Global fault collapsing is highly desirable, but is not practical because it requires extensive resources in terms of time and memory. In this paper, we have presented an approximate global fault collapsing method which produces a more compact fault list—an approximation of the global collapsed fault list. Experimental results comparing our method to local fault collapsing show that our method achieves comparable accuracy in less time.

References

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

[2] F. Brglez and H. Fujiwara, "A neutral netlist of 10 combinational benchmark circuits and a target translator in fortran", *Proc. International Symposium on Circuits and Systems,* 1985, pp. 695-698.

[3] H. K. Lee and D. S. Ha, "An efficient forward fault simulation algorithm based on the parallel pattern single fault propagation", *Proc. International Test Conference*, 1991, pp. 946-966.

[4] H. K. Lee and D. S. Ha, "On the generation of test patterns for combinational circuits", Department of electrical Engineering, Virginia Polytechnic Institute and State University, Tech. Rep. 12-93, 1993.