Improving Diagnostic Resolution of Delay Faults in FPGAs by Exploiting Reconfigurability

Jayabrata Ghosh-Dastidar* and Nur A. Touba

Computer Engineering Research Center
Department of Electrical and Computer Engineering
University of Texas, Austin, TX 78735-1084
{dghosh, touba}@cat.ece.utexas.edu

Abstract

Given an FPGA that has failed to meet its timing specification, techniques are proposed to efficiently diagnose the cause of the faulty behavior. An initial list of suspect configuration logic blocks (CLBs) and interconnects is generated using six-valued fault-free simulation and critical path tracing. The initial list of suspects is then reduced by exploiting the reconfigurability of an FPGA. Experimental results indicate dramatic reduction in the size of the suspect list.

1. Introduction

Deep submicron technology has enabled the density of configurable logic blocks (CLBs) and interconnects in an FPGA to be greatly increased. However, as a result of the greater densities and more aggressive clocking strategies, FPGAs have become more susceptible to delay faults. With ever increasing clock frequencies, small delay defects that were previously tolerable are now starting to cause timing failures. There is a need for new tools that can accurately diagnose and locate delay defects in FPGAs.

Delay fault diagnosis is significantly more difficult than stuck-at fault diagnosis as a delay fault model depends on the size of a delay defect and hence is often harder to define. Previous research has been done on delay fault diagnosis for general integrated circuits. Girard, *et al.* [Girard 92] proposed an efficient procedure based on critical path tracing [Abramovici 83] from a 6-valued simulation. In [Ghosh-Dastidar 98], techniques were proposed for handling multiple delay faults and a ranking and pruning strategy were presented based on information extracted from passing vectors and static timing information. Adaptive techniques were proposed in [Ghosh-Dastidar 99] where adjacency vectors are used to improve diagnostic resolution. In [Hsu 98], techniques were proposed to handle path delay faults.

An FPGA differs from a general integrated circuit in its capability for reconfiguration of the logic in the circuit-under-test (CUT). This unique feature is exploited in a very systematic and efficient way in the proposed method to arrive at a more precise set of suspects. The CLBs are reprogrammed, and then the modified circuit is tested using the same test set that had originally caused the circuit to fail. As the same test set is being re-used, it eliminates the time consuming process of having to generate additional diagnostic vectors for improving the resolution. This technique is targeted towards the common case of a single point delay defect that increases the delay through a CLB or an interconnect causing it to exceed its timing specification.

* Currently employed at Altera Corp. Research was performed as a graduate student at the Univ. of Texas, Austin.

2. Generating Suspect Set

The idea of performing critical path tracing using a 6-valued algebra to identify a set of suspects that may explain an observed faulty output was proposed in [Girard 92]. For a test sequence, each two-pattern test for which the circuit-under-test produced a faulty output is simulated using a 6-valued algebra based on the H6 algebra [Hayes 86].

The 6-valued symbols that are used are the following: *S0* for static zero, *S1* for static one, *R1* for a rising transition, *F0* for a falling transition, *X0* for static-0 hazard, and *X1* for a static-1 hazard. The advantage of using this 6-valued algebra is that it does not depend on any gate propagation delay or delay fault size. From each faulty output, critical path tracing is performed to identify the suspects (i.e., critical lines) that may have caused the faulty value. For each two-pattern test, *t,* which gives a faulty output response at output *j*, the set of suspects obtained from critical path tracing will be denoted as *SUSPECTS(t,j)*. The intersection of all the sets *SUSPECTS(t,j)* constitutes the final set of suspects and is denoted as *CRITICAL_SUSPECTS*. Any member of *CRITICAL_SUSPECTS* can explain all of the observed faulty behavior.

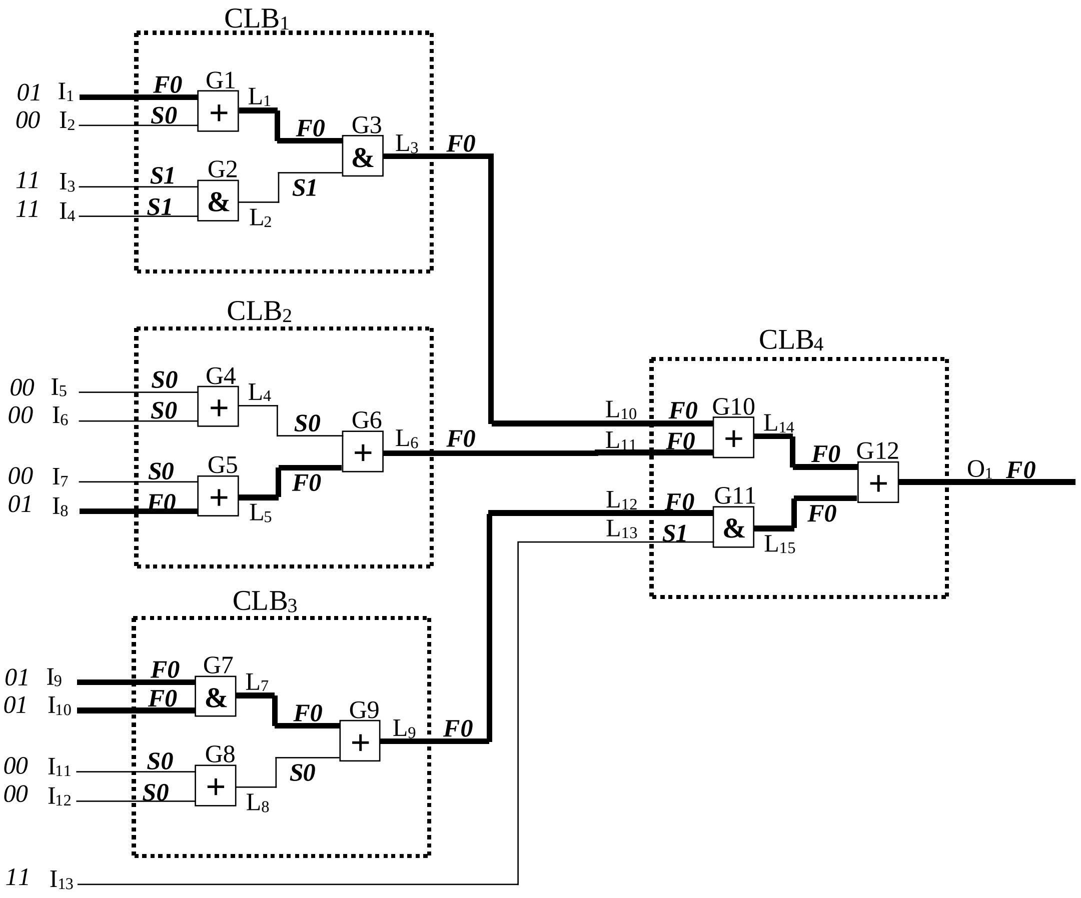

Figure 1. 6-Valued Simulation for Vector Pair *(1011000111001, 0011000000001)*
Faulty CLB: *CLB_1*
CRITICAL_SUSPECTS: *$\{O_1, CLB_1, CLB_2, CLB_3, CLB_4, L_{10}, L_{11}, L_{12}, I_1, I_8, I_9, I_{10}\}$*

In the example in Fig. 1, the CUT consists of 4 CLBs in an FPGA that have been configured to perform the specified logic function. Assume *CLB_1* has a delay fault. The primary output

O_1 has faulty value on application of the test vector pair *(1011000111001, 0011000000001)*. The 6-valued simulation is performed, and then critical path tracing starts from output *O1*. All the critical lines are marked with bold in Fig. 1. The set *SUSPECTS(t,1)* = { O_1, CLB_1, CLB_2, CLB_3, CLB_4, L_{10}, L_{11}, L_{12}, I_1, I_8, I_9, I_{10} }. Since there is only one suspect set there is no need to perform a set intersection to construct the set *CRITICAL_SUSPECTS*. So *CRITICAL_SUSPECTS* = { O_1, CLB_1, CLB_2, CLB_3, CLB_4, L_{10}, L_{11}, L_{12}, I_1, I_8, I_9, I_{10} }. Here the final set of suspects consist of all the 4 CLBs, 3 interconnects, 1 output, and 4 inputs. So in this case the diagnostic resolution is not very good. The goal of this work is to improve the diagnostic resolution.

3. Reducing Suspect Set

The previous section described the procedure for generating the initial set of suspects. This section describes how the initial set can be reduced using the reconfiguration capability of an FPGA. The general steps consist of modifying the CUT and then re-applying the original test set. This process is iteratively continued until it is not possible to gain any further information. In each loop, one CLB is selected from the set *CRITICAL_SUSPECTS* and is reprogrammed so that it becomes a constant logic 1 or constant logic 0 node. The CLB is reprogrammed by changing the contents of its look-up table (LUT). No changes are made to the placement and routing in the FPGA configuration, so physically, the configuration remains the same. Then the original test set is applied. For the entire duration of the test, that particular CLB holds a static value. The intention here is to remove that particular CLB from participating in the timing related behavior of the CUT. Since the CLB is holding a static value, it cannot affect the delay in the CUT. Under these conditions, if the CUT still fails one or more of the tests, then this CLB can be removed from the set of suspects because it cannot be the cause of the faulty behavior.

Note that if the modified CUT does not fail for any of the tests, no concrete conclusion can be drawn. This is because the CLB under consideration could be the actual defect site, and since it was held at a steady value, the faulty behavior did not materialize. But it could also be the case that the CLB under consideration is not the actual defect site, but holding it at a steady value fails to either excite or propagate the delay fault to the output. However, if the modified CUT does still fail, then there is no way for the CLB under consideration to be actual defect site, and hence it is "vindicated" and can be removed from the suspect set.

If a suspect CLB is vindicated and can be removed from the set of suspects, then the interconnect directly connected to its output can also be removed. The reason for this is that any delay fault in that interconnect could not be excited when the vindicated CLB is held at a static logic value, and since the CUT still failed under those conditions, obviously the interconnect cannot be the source of the failure. The same holds true for any CLB or interconnect in the transitive-fanin of the vindicated CLB that has no other structural path to a primary output except through this CLB. The transitive-fanin of the vindicated CLB includes any interconnect or CLB that has a direct or indirect structural path to the vindicated CLB. For the subset of interconnect and CLBs in the transitive-fanin of the vindicated CLB whose output can only reach a primary output only through the vindicated CLB, they are also vindicated because there is no way for them to cause the CUT to fail when the vindicated CLB is held at a static value.

In the procedure, the set of CLBs in the suspect set are ordered in a depth-first manner so that the CLBs closest to the primary output come first. The reason for processing the CLBs near the primary outputs first is that if one of those CLBs can be removed from the suspect set, then it may also result in a lot of CLBs and interconnect in its transitive-fanin being removed as well. This would quickly reduce the suspect set and hence make the procedure faster. Note that the CLBs near the primary outputs have the largest transitive-fanin.

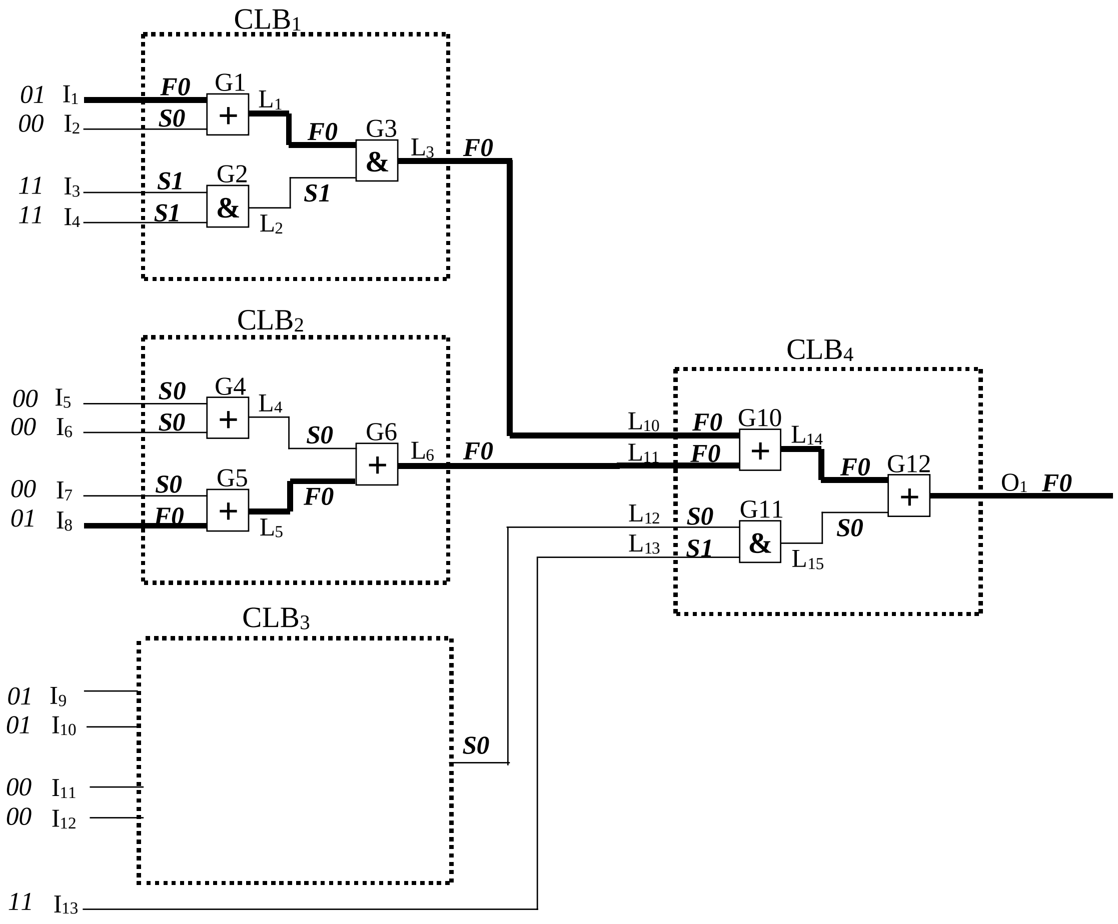

Figure 2. 6-Valued Simulation for Vector Pair *(1011000111001, 0011000000001)* with *CLB3* Reprogrammed to be Constant Logic 0
Faulty CLB: *CLB1*
Reduced CRITICAL_SUSPECTS: *{O_1, CLB_1, CLB_2, CLB_4, L_{10}, L_{11}, I_1, I_8 }*

The example in Fig. 2 is a modified version of the example in Fig.1 in which *CLB_3* is reprogrammed to become a constant logic 0 node. Fig. 2 shows the logic values in all the lines after the modification. Since *CLB_1* is the actual defect site, and the excitation and propagation of that fault is not affected by this modification, the modified CUT will fail for the application of the vector pair *(1011000111001, 0011000000001)*. So according to the explanation given above, *CLB_3* and interconnect *L_{12}* can be eliminated from *CRITICAL_SUSPECTS*. Also inputs *I_9* and *I_{10}* exclusively feed *CLB_3*, and so can also be eliminated from *CRITICAL_SUSPECTS*. So the reduced set *CRITICAL_SUSPECTS = {O_1, CLB_1, CLB_2, CLB_4, L_{10}, L_{11}, I_1, I_8 }*. The same arguments hold for modifying *CLB2* to a constant logic 0 node. So the reduced set *CRITICAL_SUSPECTS = {O_1, CLB_1, CLB_4, L_{10}, I_1 }*. It is not possible to reduce the suspect set further, as holding *CLB_1* or *CLB_4* to a constant logic value will not allow the circuit to fail the test. So the final suspect set consists of *{O_1, CLB_1, CLB_4, L_{10}, I_1 }*.

Consider the case in which *CLB3* is modified to a constant logic 1 node. As is evident from the logic values on the lines, this modification will not allow the circuit to fail, as it does not preserve the propagation condition of the fault. So this particular modification does not help reduce the suspect set.

The steps of the procedure are as follows:

1. Create an ordered list by ordering all CLBs in the suspect set in a depth-first manner so that the CLBs closest to the primary outputs come first.
2. Select the first CLB from the ordered list and reprogram its logic function so that it implements a constant logic 1, i.e., its output value is always logic 1 regardless of the value of its inputs.
3. Apply the original set of test vectors. If the circuit still fails timing, then remove the CLB from the suspect set along with the interconnect directly connected to its output. Also, remove any interconnect or CLB from the suspect set that is in the transitive-fanin of this CLB and has no other structural paths to a primary output except through this CLB.
4. Now reprogram the CLB's logic function so that it implements a constant logic 0 and repeat step 3.
5. Remove the CLB from the ordered list. If the ordered list still contains CLBs, then loop back to step 2.

4. Experimental Results

Extensive experiments were performed to validate the ideas proposed in this paper. Random delay faults were injected in the largest 5 of the ISCAS 85 [Brglez 85] benchmark circuits. In each case, the benchmark circuit was mapped to an FPGA library to perform the experiments. Two-pattern test sets were generated for the circuits using the *Soprano* ATPG tool [Lee 90].

Table 1. Experimental Results for Diagnosis of Delay Faults

Circuit	Size of Failing Cone Intersection	Size of Original *CRITICAL_SUSPECTS*	Size of Reduced *CRITICAL_SUSPECTS*
C2670	203	30	1
	204	31	5
	322	79	5
C3540	466	97	6
	477	127	12
	477	148	16
C5315	275	5	4
	275	36	13
	275	39	8
C6288	1008	348	12
	1034	430	22
	1010	560	26
C7552	179	25	8
	179	40	3
	179	51	8

Table 1 summarizes the results. For each circuit, the results for three different delay faults are shown (in three rows) which give a typical profile for what we observed in our experiments on a large number of randomly injected delay faults. The first column shows the circuit name. The second column shows the number of suspect CLBs that are obtained if only a simple cone tracing is done for each failing output, where the intersection of all the cones is used to form the set of suspect CLBs. Column 3 shows the number of CLBs in the original *CRITICAL_SUSPECTS* set obtained using the procedure described in Section 2 (which was proposed in [Girard 92]). Column 4 shows the number of CLBs in the reduced suspect set after using the proposed procedure.

As can be seen, in all cases the suspect set is significantly reduced by employing the proposed approach. Note that this improvement comes at no extra cost in terms of additional diagnostic test vector generation. The circuit modification carried out in each case is very simple, and the exact same test set can be re-used to improve the diagnostic resolution.

5. Conclusions

In this work, the reconfigurability of an FPGA is exploited to greatly improve the diagnostic resolution of delay faults. The diagnosis procedure is simple and easy to apply. This work has applications for both manufacturing test as well as user configuration test.

For manufacturing test, if a part is found to not meet timing specifications, then diagnosis is required to determine the cause of the faulty behavior. This procedure can be used to narrow down the search space to better guide direct probing techniques. This speeds up the failure analysis process and can save a lot of time.

For user configuration test, if a user generates a configuration of an FPGA for a particular application and finds that the timing specifications are not meet due to the presence of a delay defect in the hardware. The user can use the proposed approach to narrow down the set of suspect CLBs and interconnect, and then reconfigure the FPGA so that it does not use these particular suspect resources. By avoiding the faulty components within the FPGA, the user can still make use of the FPGA to implement a particular application.

6. References

[Abramovici 83] Abromovici, M., P. R. Menon, and D. T. Miller, “Critical Path Tracing - An Alternative to Fault Simulation”, *Proc. of 20th Design Automation Conference*, pp. 214-220, 1983.

[Brglez 85] Brglez, F., and H. Fujiwara, “A Neutral Netlist of 10 Combinational Benchmark Circuits and a Target Translator in Fortan,” *Proc. of Int. Symp. Circuits and Systems*, pp. 663-698, 1985.

[Ghosh-Dastidar 98] Ghosh Dastidar, J, and N.A. Touba, "A Systematic Approach for Diagnosing Multiple Delay Faults", *Proc. of IEEE Symposium on Defect and Fault Tolerance,* pp. 211-216, 1998.

[Ghosh-Dastidar 99] Ghosh-Dastidar, J, and N.A. Touba, "Adaptive Techniques for Improving Delay Fault Diagnosis", *Proc. of IEEE VLSI Test Symposium,* pp. 168-172, 1999.

[Girard 92] Girard, P., C. Landrault, S. Pravossoudovitch, “A Novel Approach to Delay-Fault Diagnosis”, *Proc. of 29th Design Automation Conference,* pp. 357-360, 1992.

[Hayes 86] Hayes, J.P., “Digital Simulation with Multiple Logic Values”, *IEEE Trans. on Computer-Aided Design*, Vol. 5, No. 2, pp. 274 -283, Apr. 1986.

[Hsu 98] Hsu, Y. C., and S.K. Gupta, “A New Path-Oriented Effect-Cause Methodology to Diagnose Delay Failures,” *Proc. of International Test Conference*, pp. 758-767, 1998.

[Lee 90] Lee, H.K., and D.S. Ha, “SOPRANO: An Efficient Automatic Test Pattern Generator for Stuck-Open Faults in CMOS Combinational Circuits,” *Proc. of 27th Design Automation Conference,* pp. 660-666, 1990.