Diagnosis of Open Defects in FPGA Interconnect

Mehdi Baradaran Tahoori
Center for Reliable Computing
Stanford University, Stanford, CA 94305.
mtahoori@crc.stanford.edu

Abstract

In this paper, we present coarse-grain and fine-grain diagnosis techniques to identify a faulty element in FPGA interconnects. The fault model we use is stuck-open and resistive-open for interconnects. The presented technique requires only a small number of configurations while offering high resolution diagnosis. We implemented this technique on real FPGA chips and verified it using fault emulation.

1. Introduction

High resolution diagnosis plays a major role in failure analysis and the yield enhancement process.

In deep sub-micron technology, opens are the most common type of defect. As reported in [Needham 98], 58% of costumer-returned parts are suspected of having open defects, specially in contact vias, meaning that open defects have not yet been addressed sufficiently. An open defect is a discontinuity in the connection between two circuit nodes that should be completely connected. A minor discontinuity results in a resistive connection, where as a major discontinuity can be treated as a connection of infinite resistance (complete-open).

Due to the reconfigurability of FPGAs, a fault in an interconnect can be avoided by using another configuration which implements the same functionality but avoids the faulty elements. Therefore, a fast and high resolution diagnosis technique can be exploited to allow the use of defective chips, and can also be used in fault tolerance schemes [Huang 01a].

We use resistive-open and complete-open fault models for wires and the stuck-open model for programmable interconnect points (PIPs). A PIP stuck-open fault causes the PIP to be permanently open regardless of the value of the SRAM cell controlling the PIP.

In this paper, we present a two-step diagnosis technique to precisely identify the faulty element(s) in FPGA interconnects. The coarse-grain step localizes the fault to a small portion of the FPGA or a set of resources (e.g. a routing path), whereas the fine-grain step precisely locates the fault inside that portion of FPGA or that set of resources. An efficient search technique is exploited in the fine-grain step so as to minimize the number of configurations required. This technique can be used either by the manufacturer during failure analysis or by an FPGA user for application-specific diagnosis or fault tolerance.

We have implemented this technique on the most recent Xilinx FPGAs, the VirtexII family, and verified our technique by fault injection on real chips using the fault emulation method [Toutounchi 01].

The rest of this paper is organized as follows. In Sec. 2 we present background including previous work and an explanation of the FPGA model we use. In Sec. 3 we present our coarse-grain diagnosis technique, followed by our fine-grain diagnosis technique in Sec. 4. In Sec. 5 we present implementation results followed by conclusion in Sec. 6.

2. Background

2.1. Previous Work

There has been research done on the diagnosis of faults in FPGAs, some of which focus on faults in logic blocks [Abramovici 00][Inoue 98][Mitra 98][Stroud 97][Wang 97], while others focus on diagnosis of faults in interconnects [Das 99][Yienlei 98][Huang 96][Lombardi 96][Liu 95].

In [Das 99], an application-dependent technique is presented for diagnosis of interconnect faults in FPGAs. There are also some application-independent diagnosis techniques in FPGAs [Yienlei 98][Huang 96][Lombardi 96][Liu 95]. The latter techniques rely on the regularity in the structure of switch matrices of older generations of FPGAs, such as Xilinx XC3000 and XC4000 series, which cannot be applied to more general structures of switch matrices in the more recent Virtex and Virtex II families.

2.2. FPGA Model

The FPGA model we use in this paper is a two dimensional array of *configurable logic blocks* (CLBs) consisting of logic blocks and switch matrices. There are four logic blocks in each CLB connected to the switch matrix through input and output MUXes (IMUX and OMUX). Each logic block consists of look-up tables (LUTs) and programmable sequential elements. Switch matrices provide the connectivity to different CLBs, while logic blocks contain the combinational and sequential programmable logic. CLBs are connected through horizontal and vertical wiring channels of different lengths, called *line segments*. Inside each switch matrix are *programmable interconnect points* (PIPs); a pass transistor controllable by a user-programmable SRAM cell. These PIPs provides selective connectivity between pairs of line segments connected to the switch matrix [Xilinx 01].

3. Coarse-Grain Diagnosis

The goal of the coarse-grain diagnosis phase is to isolate the fault to a small portion of the FPGA. For some applications, such as some fault tolerance techniques in FPGAs [Huang 01a], this phase is sufficient, where as in others, such as failure analysis, a fine-grain localization of the fault is necessary after this phase.

Test-configuration generation for FPGAs is typcially decomposed into test generation for logic and test generation for interconnects [Renovell 00]. In the test configurations for interconnects only *transparent logic* (i.e. identity function) followed by a flip-flop is implemented in logic blocks.

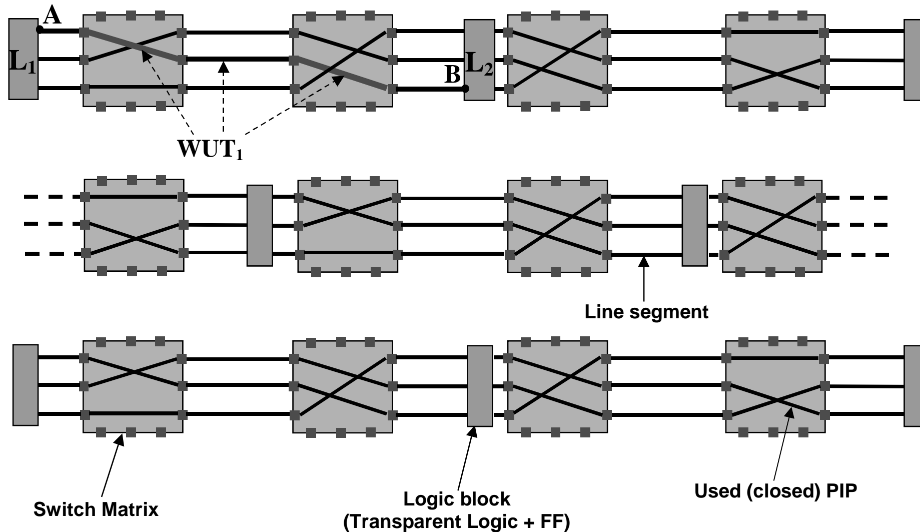

Figure 1 A test configuration for interconnect with only transparent logic.

An example of such an interconnect test configuration is shown in Fig. 1. The test configuration consists of several wires under test (WUTs) in the entire FPGA. A WUT consists of the routing path (PIPs and line segments) connecting the output of a logic block to the input of another logic block in the test configuration. The logic blocks implement transparent logic followed by a flip-flop. For example in Fig. 1, WUT_1 extends from point *A*, an output of logic block L_1, to point *B*, an input of logic block L_2. In an actual interconnect test configuration, multiple WUTs are implemented in parallel in each CLB in order to minimize the number of configurations by covering as many resources as possible.

Note that this test configuration can be viewed as parallel shift registers. The logic value of a WUT will be captured in the flip-flop connected to it in the next clock cycle. Hence, if a WUT is faulty, the content of the flip-flop connected to it is also faulty. By observing the output of the flip-flops after applying test vectors, the faulty WUT(s) can be identified. This output observation can be done by either scanning out the contents of flip-flops or by exploiting the *readback* feature of Xilinx FPGAs [Xilinx 02]. In the first case, the value observed at the PO connected to each chain corresponds to the content of a unique flip-flop in the chain at each test clock cycle. Hence, the test clock cycle at which the fault is observed at the PO identifies the faulty WUT. In the second case, which is a faster mode, the content of all flip-flops after applying each input vector can be read out and the faulty WUT can be identified much faster.

In this technique, the fault is localized to a WUT. The diagnosis granularity depends on the length of the WUT, in terms of the number of resources used in each WUT in the test configurations. This length is proportional to the distance between two consecutive used flip-flops in the test configuration.

Note that in this diagnosis phase, no extra test configuration is generated or additional test vector applied. This phase in performed just by post-processing the tester data for the set of test configurations and test vectors that have already been applied for interconnect testing (only for failing parts).

4. Fine-Grain Diagnosis

The input to the fine-grain diagnosis flow is a defective WUT, which is the result of the coarse-grain diagnosis scheme. The goal of this part of the diagnosis flow is to exactly identify the faulty resource, i.e. PIP or line segment, on the faulty WUT.

Because open faults in different resources of a WUT have the same logic effect, they are equivalent faults, and therefore the exact location of the fault cannot be identified using traditional logic diagnosis techniques. For example, if an open fault happens on any PIP or line segment on the WUT_1 shown in Fig. 1, the same effect is captured in the flip-flop of logic block L_2, and all these open faults are indistinguishable from the fault effect captured in that flip-flop. Thus, we exploit the reconfigurability and programmability features of FPGAs to solve this problem. We propose a new technique which is called *Remove/Reroute*.

The basic idea for this technique is as follows. In each configuration, a portion of the WUT is removed from the routing configuration, *remove*, and the WUT is rerouted using some resources other than those removed resources, *reroute*. If the new WUT still fails, those removed resources are defect-free, thus the fault is located on the non-removed resources. Otherwise, the exact opposite conclusion is true.

We can use some search technique, such as linear search or binary search, to exactly identify the faulty resource. The number of configurations and the number of steps depends on the search algorithm.

4.1. Remove/Reroute Technique

Figure 2 shows the basic concept of this technique. In Fig. 2.a, a WUT is shown as a part of a test configuration which is diagnosed to have an open fault. In Fig. 2.b, a portion of this WUT is removed and the WUT is rerouted without using those removed resources. In this example, the fault is located in the removed resources, therefore the new rerouted WUT will pass the test.

There are some implementation issues with this technique. Typically line segments are not directly programmable; the only programmable resources in the FPGA interconnects are PIPs [Xilinx 02]. Hence, to remove a line segment from a WUT, both incoming and outgoing PIPs for that line segment must be removed from the WUT. For example in Fig.3, to remove the line segment between *B* and *C*, both the PIPs *(A,B)* and *(C,D)* must be removed from the WUT. The dotted PIPs inside the switch matrix are those connected to *B* and *C* but not used (i.e. turned off) in this configuration.

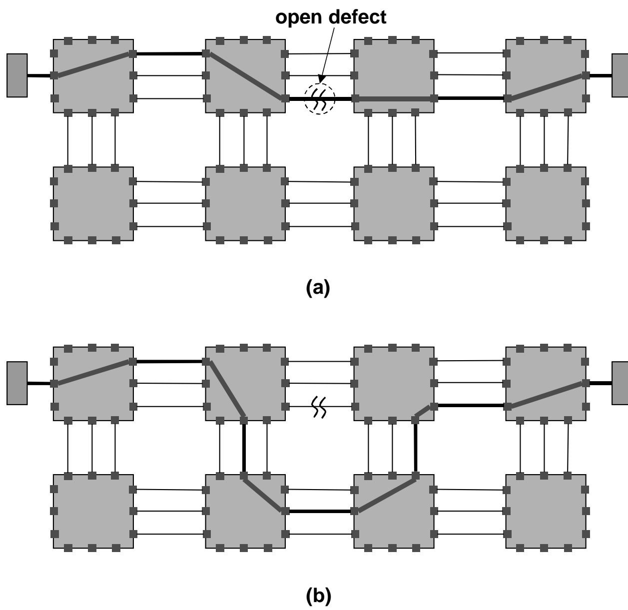

Figure 2 (a) A WUT diagnosed to be defective (b) new WUT after removing and rerouting.

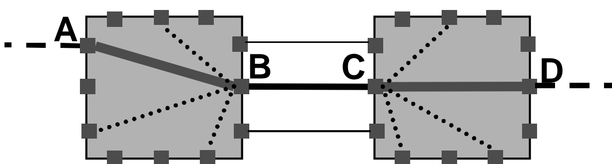

Figure 3 A portion of a WUT

In our implementation, both remove and reroute phases are automated using some features of an internal place and route tool at Xilinx Inc. In this tool, some PIPs of the FPGA can be marked so as to be not used by the place-and-route tool in completing the rerouting of the design. In order to reroute a WUT without using a particular line segment, we must mark all the PIPs in the FPGA that are connected to that line segment, so that the place-and-route tool does not use them in the rerouting phase. Note that marking only those PIPs connected to that line segment in the original configuration of the WUT is not sufficient. Figure 4 shows an example of the rerouting of the WUT shown in Fig. 3. Although the new configuration for this WUT does not use either PIP *(A,B)* or *(C,D)* from the original configuration, line segment *(B,C)* is still used, through usage of *(P,B)* and *(C,Q)*. Therefore all the PIPs in the FPGA which are connected to *B* and *C,* must be marked to not be used in the rerouting.

4.2. Search Techniques

There are two search methods to be used for finding exact failing resource using remove/reroute technique, namely linear search and binary search. There are some limitations with both techniques for this application.

In linear search, only one resource is removed from the WUT at each configuration. The first non-failing configuration determines the faulty element. Therefore, a WUT consisting of *N* resources requires *N* test configurations to be generated and *N/2* steps on average (*1* step for the best case, *N* steps for the worst case).

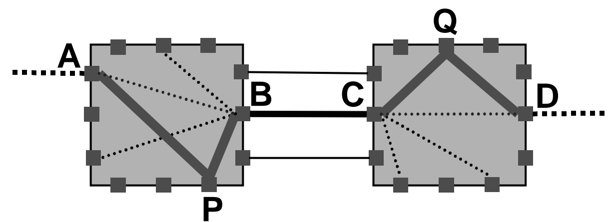

Figure 4 A new rerouting for the WUT shown in Fig.3.

In binary search, the number of removed resources is half of the total number of resources in the previous step. For a WUT of *N* resources, *N/2* of the resources are removed in the first step, forming the suspected region. At each step this region shrinks by a factor of *2*, and the position of this region depends on the result of previous steps. Therefore, log_2N steps are needed to determine the faulty resource. Note that in this method, we form a binary decision tree of height $\log_2$N. Hence the number of nodes in the tree which corresponds to the number of test configurations is *N–1*. All these test configurations must be pre-generated before test application because configuration generation time is much more than test application time. Hence, the test storage is almost the same as linear search. Another drawback of binary search is that the selection of next test configuration depends on the result of previous test configuration. This may slow down the test application time. It would be much faster if all the test configurations are loaded in the burst mode and results are collected and analyzed later.

We propose a new search technique for this problem, in which both the number of steps and test configurations are logarithmic with the number of resources. Also, the test configuration to be used at each step is pre-determined, unlike binary search. The idea of this search technique is similar to Walsh-Rademacher codes, but some modifications are performed to be applicable for FPGA diagnosis. We call this search technique *overlapped search*.

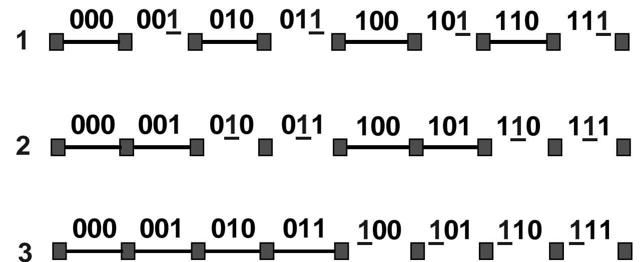

Figure 5 Three configurations in overlapped search for a WUT of 8 resources.

In this technique, in each test configuration exactly *N/2* resources are removed, and only log_2N configurations are needed. The failing test configurations uniquely identify the faulty resource.

The detail of this technique is as follows. For a WUT of *N* elements, we have log_2N test configurations, which are called configuration *1* through configuration log_2N. Assume for simplicity that *N* is a power of *2*, without loss of generality. Consider the binary representation of the resources in the WUT. As there are *N* resources, log_2N bits are sufficient. In each test configuration *i*, the resources with bit *i* set are removed from the WUT and the WUT is rerouted without using those resources. Figure 5 shows an example of this technique for a WUT of 8 resources after the removal phase. As can be seen in this figure, only three configurations are needed, and in each configuration, exactly four resources from original WUT are removed. If we denote a failing configuration as **0**, and a non-failing configuration as **1**, the log_2N test configurations correspond to the binary representation of the faulty resource in the WUT (consider configuration 1 as LSB and configuration log_2N as MSB). Based on a single fault assumption, the faulty resource is uniquely diagnosed using this technique. For example, if the second resource (marked with **001** in Fig. 5) is faulty, the first configuration will pass the test while the other two fail. Thus the fault pattern is **001** which indicates the faulty element.

This technique not only offers the minimum number of steps and configurations, but also the next configuration to test is pre-determined. The second feature enables us to load the configurations rapidly and reduces test time.

5. Implementation Results

We implemented this technique for diagnosis of open faults on Xilinx VirtexII FPGAs. The removing and rerouting phases are implemented by exploiting the internal place-and-route tool as described in Sec. 4. For the coarse-grain diagnosis step, we used interconnect test configurations internally developed at Xilinx Inc.

The fine-grain diagnosis flow gets the original test configuration and the faulty WUTs as input. The other portions of the configuration not relevant to those WUTs are removed from the configuration. This simplifies the task of the place-and-route tool in rerouting the WUTs, as more unused resources are available for rerouting. The test configurations are generated based on the method described in the paper.

The presented method is verified by injecting faults on real FPGA chips using the fault emulation technique [Toutounchi 01]. The open fault on a particular PIP can be emulated by changing the value of the memory cell controlling the PIP (pass transistor), from **1** (turned on) to **0** (turned off). Note that the value of these memory cells are part of the configuration data, and hence are programmable.

6. Summary

In this paper, we presented a two-step diagnosis method for high resolution localization of open faults in FPGA interconnects. The first step, coarse-grain diagnosis, is the by-product of interconnect testing of FPGA in which only transparent logic and flip-flops are implemented in logic blocks. The second step, fine-grain diagnosis, is performed by removing some resources from a defective WUT and rerouting the WUT without using those resources. An efficient search technique is exploited to identify the faulty resource in the minimum number of configurations. Our technique was implemented on real FPGA chips and also verified using fault emulation method.

This technique can be used either for failure analysis and yield enhancement process by manufacturer, or as a method for diagnosis of faults in user applications.

Acknowledgement

The Author would like to thank Professor Edward J. McCluskey from Stanford CRC for supervision of this project. This work was supported by Xilinx Inc. under contract number 2DSA907.

References

[Abramovici 00] Abramovici, M., C. Stroud, “BIST-Based Detection and Diagnosis of Multiple Faults in FPGAs,” Proc. Int’l Test Conf., 2000.

[Das 99] Das, D., N. A. Touba, “A Low Cost Approach for Detecting, Locating, and Avoiding Interconnect Faults in FPGA-Based Reconfigurable Systems,” Proc. Int’l. Conf. on VLSI Design, 1999.

[Hamilton 99] Hamilton, G., G. Gibson, S. Wijesuriya, C. Stroud, “Enhanced BIST-Based Diagnosis of FPGAs via Boundary Scan Access,” Proc. VLSI Test Symp., pp. 413–418, 1999.

[Huang 01a] Huang, W.-J., and E.J. McCluskey, “Column-Based Precompiled Configuration Techniques for FPGA Fault Tolerance,” Proc. 2001 IEEE Symposium on Field-Programmable Custom Computing Machines, Rohnert Park, CA, Apr. 30 - May 2, 2001.

[Huang 96] Huang, W.K., X.T. Chen, and F. Lombardi, “On the Diagnosis of Programmable Interconnect Systems: Theory and Application,” Proc. VLSI Test Symp., pp. 204-209, 1996.

[Inoue 98] Inoue, T., S. Miyazaki and H. Fujiwara, “Universal Fault Diagnosis for Lookup Table FPGAs,” IEEE Design and Test of Computers, pp. 39-44, January-March, 1998.

[Liu 95] Liu, T., F. Lombardi, and J. Salinas, “Diagnosis of Interconnects and FPICs Using a Structured Walking-1 Approach,” Proc. VLSI Test Symp., 1995, pp. 256-261.

[Lombardi 96] Lombardi, F., D. Ashen, X. Chen, and W.K. Huang, “Diagnosing Programmable Interconnect Systems for FPGAs,” Proc. Int’l Symp. on FPGAs, pp. 100-106, 1996.

[Mitra 98] Mitra, S., P.P. Shirvani, and E.J. McCluskey, “Fault Location in FPGA-Based Reconfigurable Systems,” Proc. IEEE Intl. High Level Design Validation and Test Workshop, La Jolla, CA, Nov. 12-14, 1998

[Needham 98] Needham, Wayne, C. Prunty, E. H. Yeoh, “High Volume Microprocessor Test Escapes, An Analysis of Defects Our Tests Are Missing,” Proc. Int’l Test Conf., pp.25-34, 1998.

[Renovell 00] M. Renovell, Y. Zorian, “Different Experiments in Test Generation for XILINX FPGAs,” Proc. Int’l Test Conf., 2000.

[Stroud 97] Stroud, C., E. Lee, and M. Abramovici, “BIST Based Diagnostics of FPGA Logic Blocks,” Proc. Int’l Test Conf, 1997, pp. 539-547, 1997.

[Toutounchi 01] Toutounchi S. et. al., “Fault Emulation, A Method of FPGA Test,” US Patent pending, April 2001.

[Wang 97] Wang, S. J., et al., “Test and diagnosis of faulty logic blocks in FPGAs,” Proc. ICCAD, pp. 722-727, 1997.

[Xilinx 01] “The Programmable Logic Data Book 2001,” Xilinx Inc., 2001.

[Yinlei 98] Yinlei Yu, Jian Xu, Wei Kang Huang, F. Lombardi, “A Diagnosis Method for Interconnects in SRAM Based FPGAs,” Proc. Asian Test Symp., pp. 278-282, 1998.