FPGA implementation of Fault Diagnosing of Interconnects in SRAM Based FPGA's

Kishore Pinninti
Assistant Professor, Department of ECE, VNR VJIET, Hyderabad, India

ABSTRACT

This Paper presents a method of fault detection and location in the interconnects of a Field Programmable Gate Array. The proposed testing scheme uses a test manager which defines a part of the chip as the pattern generator and the other half as response analyzer. The chip is reconfigured several times to cover all portions of interconnects. The outcome of each reconfiguration is a bit which provides a pass or fail. Testing is done in two phases, phase one involves several reconfigurations intended to detect various faults in the interconnect structure. The test manager provides the required test sequence in each configuration. This phase involves extensively testing the complete interconnect structure for all possible faults namely configurable interconnect points stuck on, configurable interconnect points stuck off, wire stuck-at-1,wire stuck-at-0,two adjacent wires short and wires open..

Keywords*: *FPGA, testing, fault model, fault diagnosis

I. INTRODUCTION

With the wide application of FPGAs, its testing technology has been developing rapidly. There are many recent articles on the FPGA testing [1]-[5]. Some researchers proposed methods for testing the logic block, while some others deal with the interconnection resources. In [2] and [3], only the verification test is discussed. The method proposed in [4] requires too many programming steps, while only fault detection is discussed in [5]. This paper proposes a method that needs at most five programming steps to diagnose faults in FPGA interconnection resources. A single fault model is used in the discussion. The configurations in the first three programming step is similar to the configurations in [5-7]. With the proposed method, the accuracy of fault location is a single segment for a segment stuck-at or stuck-open fault, a segment pair for a bridge fault and a switch for switch stuck-on(off) fault under the single fault assumption. The following parts of this paper are organized as follows: The second section introduces the simplified model of the interconnection resource and its single fault model. Section 3 is an intensive description and analysis of the proposed programming method. Application to XILINX FPGAs of the proposed method is given in Section 4 and the last section summarizes the paper.

II. STRUCTURE MODEL OF FPGAS

The basic structure of an FPGA consists of four parts: Logic Blocks, Interconnection Resources, I/O Blocks and Programming Circuits. The basic structure without programming circuits is shown in

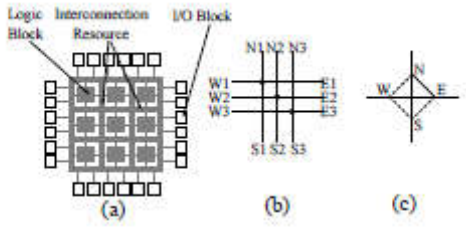

Figure 1. The Basic Structure of an FPGA

Figure 1(a). The Interconnection Resource consists of tracks and switch matrices. There are horizontal and vertical tracks. Every track is composed of all the segments and switchboxes arranged in a straight line, while several switches form a switch matrix. The area between two switch matrices is referred to as **segment block**. There are segments in each segment block. A switch matrix has terminals in each direction (side). We assume that the switchboxes only exist on the diagonal of a switch matrix. Figure 1(b) shows a switch matrix with three switchboxes and three terminals in each side. A switchbox is represented by a small dot in the figure. The model of a switchbox is shown in Figure 1(c). There are six switches in a switchbox, which can connect W and N, and S, N and E, S and E, Wand E and N and S respectively when they are turned on. A switch that is used to connect W and N (N and E, , N and S) is referred to as switch NW (WS, NE, SE, WE or NS). A switch is on a track if it can connect two segments in the track. A thin line is used to denote a switch that is OFF and a thick line

to represent a switch that is ON, as shown in Figure 1(c). In a single switchbox, more than one switch is permitted to be on. In our discussion, a single fault model is used. It is assumed that there is at most one fault in a FPGA's interconnection resources. The switch programming faults need special test and are not discussed here. We assume that the programming circuit had already undergone the test and no fault had been found, so they will not be considered in this paper. It is also assumed that floating output is reported as '0'. The following four type of single faults are considered. (1) Single segment stuck-at-1(0) fault. (2) Single segment stuck-open fault. (3) Single switch stuck-on(off) fault. (4) Single bridge fault(connects two segments). One of the following two kinds of bridge faults can occur: (i) Parallel Bridge (PB) fault: A bridge fault between two adjacent parallel segments in a segment block. (ii)Orthogonal Bridge (OB) fault: A bridge fault between a vertical segment and a horizontal segment surrounding a switch matrix. Line-OR output of the bridge fault is assumed.

III. FAULT DETECTION AND LOCATION

Fault detection means the discovery of something wrong in a digital system. Fault location means the identification of the faults with components, functional modules or subsystems, depending on the requirements. Fault analysis includes both fault detection and fault location.

Fault detection in a logic circuit is carried out by applying a sequence of test inputs and observing the resulting outputs. Therefore the cost of testing includes the generation of test sequences and their application. One of the main objectives is to minimize the length of the test sequence. Any fault in a non redundant 'n' input combinational circuit can be completely tested by applying all 2^n input combinations to it. However 2^n increases very rapidly as 'n' increases. For example for acircuit with n = 60 and 10000 tests per second the total test time for the circuit would be about 3.5 million years. A complete truth table exercise of a logic circuit is not necessary. Only the number of input combinations which detect most of the faults in the circuit is required.

3.1 Phase path method

The fault location in FPGA's has been carried out by phase path method. The location of the fault has been carried out in six different phases. As the target FPGA is a XILINX 4000E, which has a symmetrical architecture of CLB's we consider a 3 X 3 CLB structure as shown in Figure 2 below for the experimental purpose and this can be extended for a n X n structure.

Steps for fault location

For the location of the fault in a 3 X 3 CLB structure as shown in the figure below we shall consider the faulty switch to be the CLB-5. Various steps in this are as follows.

PHASE I: This phase covers the path in forward stair case direction, where three stair case paths are considered. The first or the central chain consists of the CLB's 1-2-5-6-9. the upper chain consists of the CLB 3 and the lower chain consists of the CLB's 4-7-8. As we considered the CLB 5 as the faulty switch, the central chain CLB flags are incremented by 1, and shown in the Figure 3.

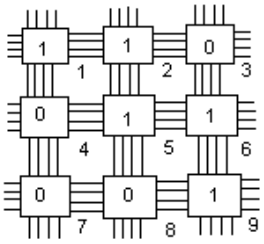

Figure 2..Structure of a 3 X 3 CLB

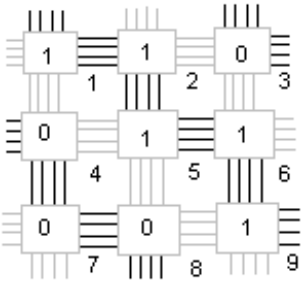

Figure 3. Configuration of Phase I

PHASE II: In this method the chains used are shifted by 1 step and the chains are: central chain 1-4-5-8-9, upper chain 2-3-6 and lower chain 7. Since we are considering the CLB 5 as the faulty the central chain is the faulty and all the CLB's flags are incremented by 1 and the new flag values are as shown in the Figure 4.

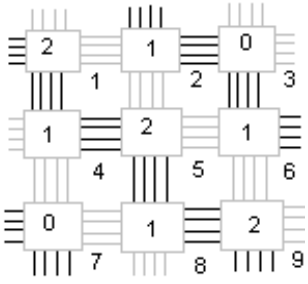

Figure 4.Configuration of Phase II

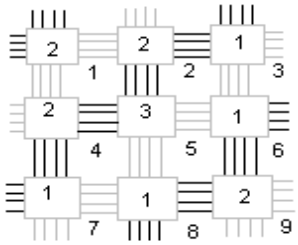

Figure 5. Configuration for Phase III

PHASE III: In this phase we use the reverse stair case technique for the location of the fault. The three chains used for the location of the fault are central chain 3-2-5-4-7, upper chain 1 and lower chain 6-9-8. As seen the central chain consists of the faulty switch which is 5 and all its CLB's are incremented by 1 and the new flag values are shown in the Figure 5.

PHASE IV: In this phase we use the reverse stair case method but by shifting the CLB's by 1 step. The paths or chains used are central chain 3-6-5-8-7, upper chain 2-1-4 and lower chain 9. In these paths the faulty path is again the central path as it contains The switch 5 (faulty) and all its CLB flags are incremented by 1. The configuration for this phase is as shown in the Figure 6.

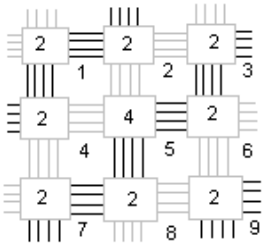

Figure 6. Configuration for Phase IV

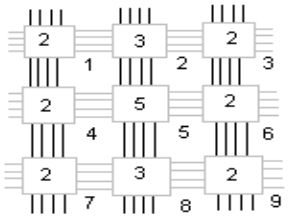

Figure 7. Configuration for Phase V

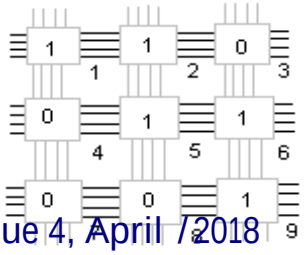

Figure 8. Configuration for Phase VI

PHASE V: In this phase we us the vertical paths for locating the faults in the CLB structure. The paths used are thus path-1: 1-4-7, path-2: 2-5-8 and path-3: 3-6-9. in this path the path-2 with CLBS 2-5-8 is the faulty path and these 3 CLB flags are incremented by 1, and the configuration is shown in the Figure 7.

PHASE VI: In the phase VI the last of the phases use for the location of the faults we use the horizontal paths of the interconnects in locating the faults. The paths used are central path: 4-5-6, upper path: 1-2-3, and the lower path: 7-8-9. In this phase the faulty path is the central path with CLBs 4-5-6. Thus the flags for these CLBs are incremented by1 and the configuration is as shown in Figure 8.

3.2. Analysis of the steps

If we closely observe all the six phases, we find that we consider all the 9 CLBs for test in each of the six phases with a different set of input output ports. By testing with all these 6 phases we can clearly pass through the faulty switch in all of the phases only once and its flag is incremented by 1 in all the phases. Thus the CLB or switch with the flag count of 6 (the highest possible value as we use 6 phases) is the faulty switch and this need only to be reprogrammed for getting the non faulty FPGA configuration. Thus we observe that six phases are sufficient to test and locate any fault occurred in either CLB or the Interconnects or the I/O lines in the targeted FPGA.

IV.ARCHITECTURE OF FAULT ANALYSIS

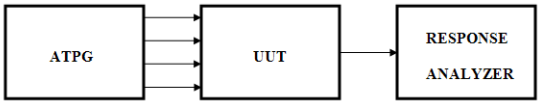

Figure 9. Architecture of fault analysis

4.1 Automatic Test Pattern Generation (ATPG)

Automatic test pattern generator (ATPG) is the device which is used to generate the required test patterns to be given to the Unit under Test, in this case the FPGA to conduct an efficient test to cover all the faults under consideration. Generally to conduct a comprehensive test we need to give all the possible input combinations to test. But as complexity increases this

type of test is not viable due to the number of test patterns and the time taken for the testing of these patterns. We need to give only those patterns so as to get an efficient result in a short time.

The system is generating test patterns using Linear Feed Back Shift Register (LFSR) technology which generates sequence of test patterns to be used for testing the unit under test. The LFSR generates random sequences of which only specific sequences are selected for testing specific faults. For example, to test the bridging faults occurred in a global line of a program FPGA we use only test sequences which are exactly inverse of each other and also the bits next to each other are inverse.

4.2. Unit Under Test (UUT)

The systems which are programmed and need to be tested are called as the Unit under Test (UUT). In this case we are using a programmed FPGA as the UUT for which the test equipment has been developed, and the target UUT or FPGA used is XC4000E series.

4,3, Response Analyzer

In the response analyzer we test the output obtained from the UUT we the expected output from the test pattern given so as to get the result. We consider the logic of 1 for a fail or faulty condition and 0 for a pass or a non faulty condition. For analyzing different types of faults we use different logics and also the logic differs for a single bit and a multi bit data buses. In the present test we are considering the logic for testing the stuck at faults mainly in long lines.

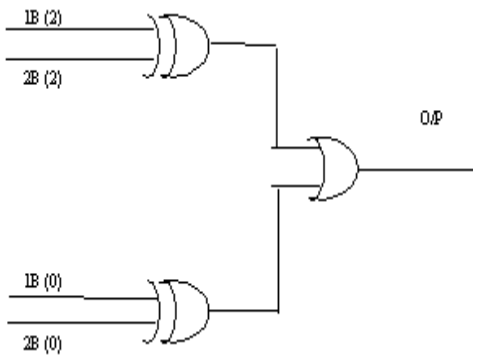

Figure 10. Response Analyzer logic diagram for Long Line Testing

Firstly for a long line testing where we use a logic consisting of "xor" gates and an "or" gate as shown in the figure for considering the stuck at faults. Here the bit sequence for 1B (1 down to 0) is from the UUT and the bit sequence from 2B (1 down to 0) is for the expected out put values. By doing XOR gate for each bit if there were to be a stuck at fault at any of the bit values when a same bit value appears it may not give any fault but if any opposite bit value to that of the stuck value occurs the XOR outputs a 1 and this is carried to the output of the OR gate giving rise to a 1 i.e. fault condition. This helps in detect either stuck at 1 or stuck at 0 faults in any of the bit lines.

V. RESULTS AND CONCLUSIONS

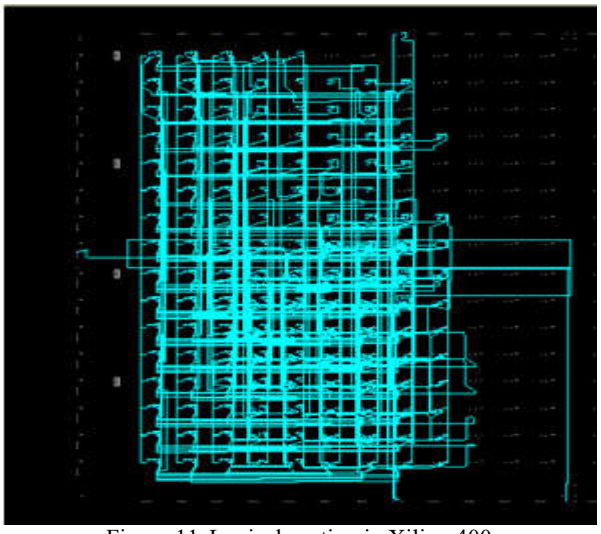

Figure 11. Logical routing in Xilinx 400

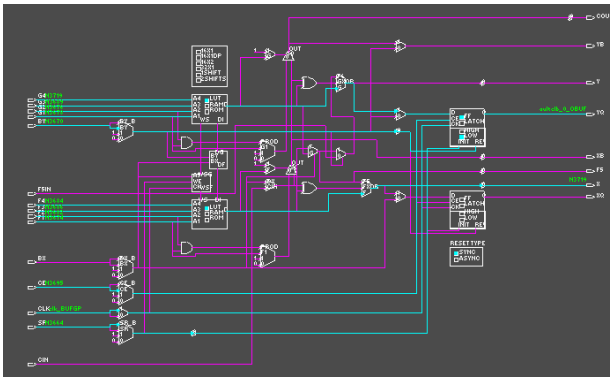

Figure 12. Logical placement in Xilinx 4000

In this paper, we aimed to detect and locate the bridging faults in FPGA interconnect. For this Xilinx xc4000E series FPGA is used as a model. The testing has been done to achieve minimum number of reconfigurations and maximum fault location capabilities. The results from each reconfiguration are stored in the on chip RAM. By looking at this RAM lookup tables, users will be able to locate the fault. Various kinds of faults are simulated using VHDL

Programming. The testing phases are successfully tested upon these faults and the simulation results show that the proposed model works as per requirement. This model can be extended to locate multiple faults and the resolution can be made as high as needed. The number of reconfigurations in this case depend on the extent of resolution required.

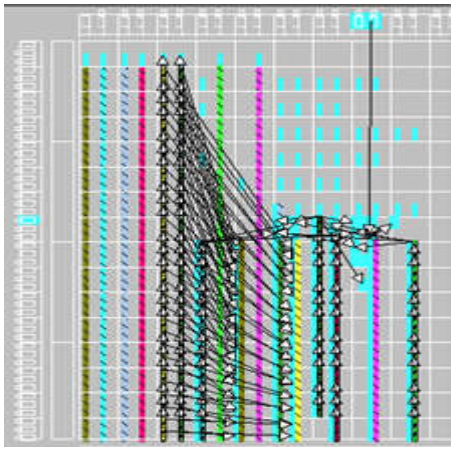

Figure 14. Logical floor planning in Xilinx 4000

References

[1]. G Harris and R.Tessier "Testing and diagnosis of interconnect faults in cluster based FPGA architecture", IEEE transactions on Circuits and Systems", Vol No.21, November, 2002.

[2]. [Mehdi Baradaran Tahoori,"Interconnect Testing of FPGA", March 2002.

[3]. Digital System Testing and Testable Design", M. Abromovici, M.A.Breuer A.D.Friedman, 2001.

[4]. Thomas H Coremen, Charles E Leiserson, Ronad L Rivest and Clifford Strein, "Introduction to Algorithms", pp 540-547, 2 edition, The MIT Press, 2001.

[5]. M. Renovell, J. Figueras and Y. Zorian. "Testing the Interconnect Structure of Unconfigurated FPGA", *IEEE European Test Workshop*, pp. 125-129, Sete, France, June 1996.

[6]. M. Renovell, J. M. Portal, J. Figueras, Y. Zorian, "Testing the Interconnect of RAM-Based FPGAs", *IEEE Design & Test*, pp. 45-50, January-March, 1998.

[7]. H.Michinishi, et. al., "Testing for the programming circuits of LUT based FPGAs", *Proc. Asia Test Symp.*,1999.