A Method of FPGA Interconnect Resources Testing by Using XDL-Based Configuration

Xiang-Fen Wang, Shi-Hong Si, Cheng Gao, Jiaoying Huang
School of Reliability and Systems Engineering
Beihang University
wangxf@buaa.edu.cn

***Abstract*—FPGA is widely used in military and aerospace applications and FPGA testing is the most effective means to ensure the reliability of them. Interconnect Resources testing is one of the most important parts of FPGA testing since that most of the faults occur on Interconnect Resources. FPGA needs to be configured as specified circuits before being tested and conventional HDL-based configuration can not achieve controllability of the resources to be tested. In order to solve this problem, we propose a XDL-based configuration.**

In this paper, grammar and design flow of XDL were analyzed firstly. Then structure of interconnect resources of Xilinx Spartan-3 FPGA and their description using XDL were studied. According to the structure of the interconnect resources, a BIST structure was built to implement the test of them. Through C++ programming, we got XDL programs automatically. With four configurations, we implemented the testing of hex lines across CLBs and not across CLBs as well as double line across CLBs and not across CLBs. The method could solve the problem of uncontrollability of the resources to be tested efficiently.

Keywords-FPGA; Interconnect Resources testing; BIST; XDL-based configuration

I. INTRODUCTION

FPGAs (Field Programmable Gate Arrays) are programmable devices that are configured to implement any user-defined functions [1]. Because of their short turnaround time and programmability, they have been widely used in military and aerospace applications which need high reliability. FPGA testing is the most effective means to ensure the reliability of them.

According to their different internal resources, FPGA testing can be broadly divided into CLB (Configurable Logic Block) testing, IOB(Input/Output Block) testing, BRAM(Block RAM) testing, IR(Interconnect Resource) testing etc[2]. Since 80% of the configurable memory bits of a FPGA associated with Interconnect Resources, practically, the probability of faults happened to Interconnect Resources is much greater than those of other resources [3]. Therefore, Interconnect Resources testing becomes one of the most important parts of FPGA testing.

FPGA needs to be configured as specified circuits before being tested. Conventional HDL(Hardware Description Language)-based configuration needs to be synthesized and optimized through development tools, thus would cause uncontrollability of the resources to be tested [4]. In the past years, there are many researches concentrating on algorithms for interconnect resources testing. But most of them does not discuss how to control the resources to be tested during configuration [5-7]. In paper [8], an in-house designed software to achieve the control of the resources to be tested is proposed, but this method is not universal. To solve this problem, we propose a XDL-based configuration.

This paper will be structured as follows. In section II, XDL and XDL-based configuration flow are introduced. In section III, we focus on talking about the structures of interconnect resources of Xilinx Spartan-3 XC3S400 FPGA which is chosen as an example. The description of the interconnect resources using XDL is explain meanwhile. In section IV, BIST Structure and Test method of FPGA Interconnection Resources are described in detail. In section V, we will discuss our approach to implement the configuration and testing of hex lines and double lines. Section VI is the conclusion of the paper.

II. XDL-BASED CONFIGURATION METHOD OF XILINX FPGAS

XDL (Xilinx Design Language) is a readable text language provided by Xilinx Corporation [9]. It describes FPGA bottom resources and their behaviors through certain grammatical rules.

This work was supported by technical basis research projects in Science and Industry Bureau of China under Grant No.Z132012A001 and Z1320130013, and Natural Science Foundation of China under Grant No.61201028 and 61376042.

978-1-4799-7958-5/14/$31.00 ©2014 IEEE

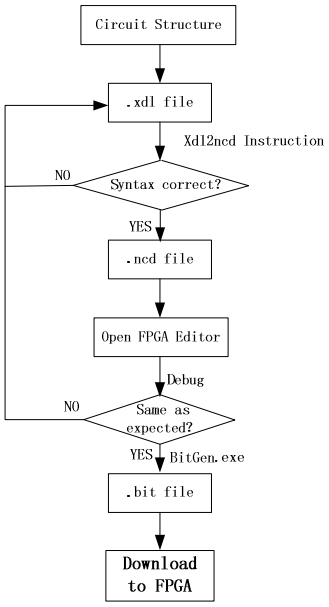

Figure 1 XDL configuration flow

XDL grammar includes instances and networks definition. All resources of a FPGA that can achieve certain logical function must be defined as instances. Then these instances can be connected by the definition of networks according to certain rules. XDL configuration flow is shown in Figure 1. First, XDL program is written according to needed circuit functions and structure of the resources of a specific FPGA. Then .xdl file is converted to .ncd file on ISE platform using xdl2ncd instruction. After syntax examination, .ncd file can be opened with FPGA Editor to make sure that the design is the same as expected. Finally, .ncd file will be converted to .bit file by BitGen.exe and downloaded to FPGA to complete the configuration.

III. INTERCONNECT RESOURCES OF SPARTAN-3 FPGA AND XDL DESCRIPTION

Interconnect Resources are the programmable networks of signal pathways between the inputs and outputs of functional elements within the FPGA, such as IOBs, CLBs, DCMs and BRAMs. They have a huge quantity and play a vital role. In the whole resources of the latest FPGA, more than 80% of the transistors are included in the Interconnect Resources [11]. As shown in figure 2, Interconnect Resources include routings and PIPs(Programmable Interconnect Points). Routings connect to each other through PIPs which are mainly located in switch matrices and return matrices.

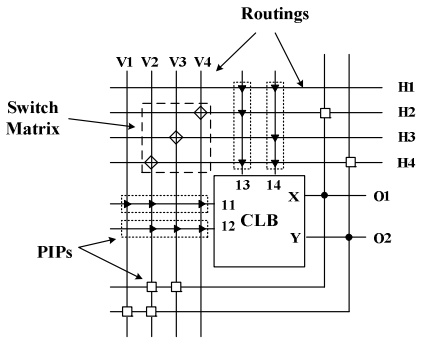

Figure 2 Spartan-3 FPGA interconnect resources

A. Routings

Xilinx Spartan-3 FPGAs have four kinds of routings: long lines, hex lines, double lines, and direct lines. Long lines connect to one out of every six CLBs. Hex lines connect one out of every three CLBs. Double lines connect to every other CLB. Direct lines afford any CLB direct access to neighboring CLBs.

Testing of long lines and direct lines is relatively easy, so we focus on testing of hex lines and double lines in this paper. Different from long lines, hex lines and double lines are unidirectional, so they are only driven from one end of the route. Using XDL, these three endpoints of the line can be named as BEG (starting point), MID (midpoint) and END(end point). Signals input from BEG and output from MID and END. In every switch matrix, there are eight hex lines and eight double lines in the direction of East, South, West and North respectively. In XDL program, we use E, S, W and N to denote the four directions of East, South, West and North. Meanwhile we use "2" and "6" to denote double lines and hex lines. Eight lines can be indicated by the numbers 0 to 7. For example, the starting point of the fourth double line in north can be described as N2BEG3.

B. Switch Matrix

All input and output ports of FGPA routings and functional elements are located in switch matrices. Through the configuration of PIPs in the switch matrices, we can implement the interconnection of different internal resources of a FPGA. There are 34 rows and 32 columns of switch matrices in a XC3S400 FPGA. 32 rows and 28 columns of them lie on the die along with CLBs and can be named as RiCj(R means Row, C means Column, i means the number of rows and j means the number of columns) in XDL. 2 rows and 2 columns of the switch matrices lie along with IOBs. In XDL, we can use T(Top), B(Bottom), L(Left), and R(Right) to indicate their positions and R(Row) and C(Column) to indicate the row or column where they are located. The rest 2 columns of the switch matrices lie along with BRAMs and can be described as BRAMRiC1 and BRAMRiC2(i means the number of rows) in XDL. The four switch matrices located in the corner of the die can be described as TL(Top Left), TR(Top Right), BL(Bottom Left) and BR(Bottom Right). The switch matrix connection points, the so called intermediate nodes, are depicted in Figure 3 [12].

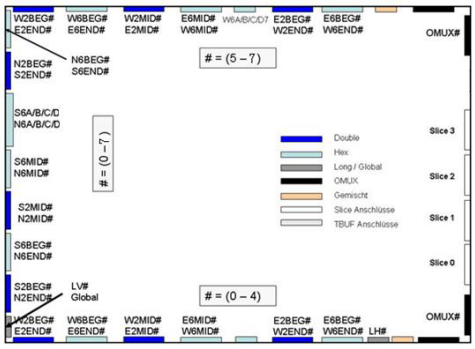

Figure 3 Alignment of Switchmatrix connection points

C. *Return Matrix*

As shown in Figure 4, there is a return matrix in the end of every row and column. Return matrix provide configurable connection between the routings of the FPGA on the boundary of the device. Only hex lines and double lines can get in and out of return matrices.

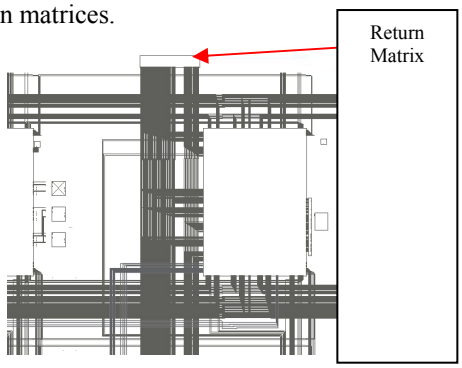

Figure 4 Return matrix

IV. BIST STRUCTURE OF FPGA INTERCONNECTION RESOURCES

The programmable characteristic of FPGA makes it feasible to construct BIST(Built-in Self Test) structures to test the internal resources [13]. A BIST structure includes a TPG(Test Pattern Generation), a CUT (Circuit Under Test) and an ORA (Output Response Analysis) [14]. TPG and ORA can be got through the configuration of a CLB [15]. Each CLB of a Spartan-3 FPGA comprises four slices and every slice consists of two LUTs(Look Up Tables) and two flip-flops. Figure 5 shows the structure of TPG and ORA. The TPG is composed by a LUT and a flip-flop. The LUT is configured as a inverter with its output connected with the flip-flop. The signal coming from the output of the flip-flop is divided into two branches: one is feed back to the input of the LUT and the other is transmitted to the CUT as test vector. The ORA is composed by another pair of LUT and flip-flop of the same slice. This LUT is configured as an XOR gate and response of the CUT and test vector will do XOR operation here. When response of the CUT and test vector are different, it means a fault exists in the CUT and the ORA will output the result of judgment which is denoted by logic 1.

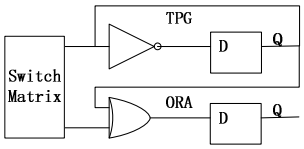

Figure 5 Structures of TPG and ORA

From the descriptions of the third part we know routings of FPGA can be divided into vertical lines and horizontal lines. Figure 6 and 7 show the BIST structure for testing of vertical lines and horizontal lines. The line under test start from an original switch matrix and connected with the next line segment which has the same number in the next switch matrix. Continue to do so until it reaches a return matrix, the line changes to the opposite direction. At last, the line under test comes back to the original switch matrix. We can see the CUT consists of two lines (two hex lines or two double lines) in different directions. Each CLB has four slices and every slice can built a BIST structure. Since a BIST structure can test two lines, each CLB can test eight lines.

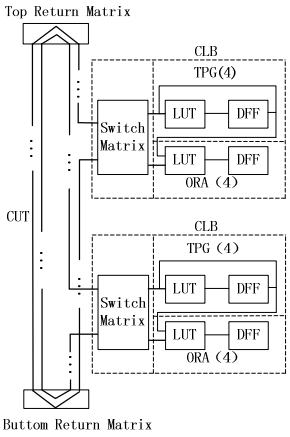

Figure 6 BIST structure for vertical lines

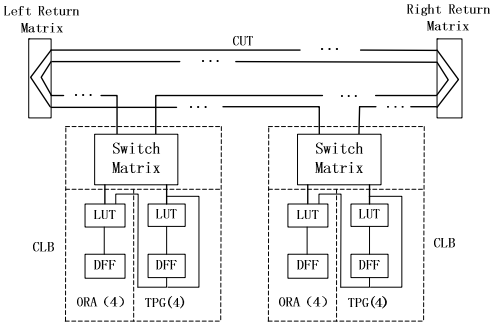

Figure 7 BIST structures for horizontal lines

Since some rows or columns have no CLBs, such as the BRAM columns and IOB columns and rows, we need to use CLBs of other rows or columns to test these lines. Thus, the testing of hex lines and double lines can be divided into four parts: testing of hex lines across CLBs, testing of hex lines not across CLBs, testing of double lines across CLBs and testing of double lines not across CLBs.

V. IMPLEMENTATION OF XDL-BASED TESTING CONGIGURATION

On Xilinx XC3S400 FPGA, we implemented the testing of the above four kinds of routings through XDL-based configuration programs. The process was as follows.

Step 1: Select CLBs. According to the locations of the lines, we selected suitable CLBs to construct BIST structures to test them. Take double lines across CLBs for instance, the first and second rows of CLBs from the first column to the 26th column were selected to test the lines from the first column to the 26th column while the third and fourth rows of CLBs from the 27th column to the 28th column were selected to test the lines from the 27th column to the 28th column.

Step 2: Construct TPG. In a selected CLB, we construct two groups of TPGs and set the initial values of the flip-flops different to test two adjacent lines.

Step 3: Construct ORA. In a selected CLB, we construct two groups of ORAs to analysis the outputs of the CUTs. The initial values of the flip-flops were set as 0. The judgment method was described in part IV.

Step 4: Connect CUT. The lines under test were connected together through switch matrices and formed integrated CUTS with the starting points connected to TPGs and end points connected to ORAs.

Step 5: Get XDL Programs. Since the quantity of the resources of a FPGA was huge, it was difficult to describe all of them with XDL. XDL programs were generated used C++ code and then verified according to the flow in figure 1.

Step 6: Configure FPGA and Read Back Results. XDL programs were converted to .bit files by Bitgen.exe and then downloaded to FPGA. A read back file was created by ISE platform. When power on, the read back file could read back testing results from ORAs. Then we could detect and located faults of the routings of the FPGA.

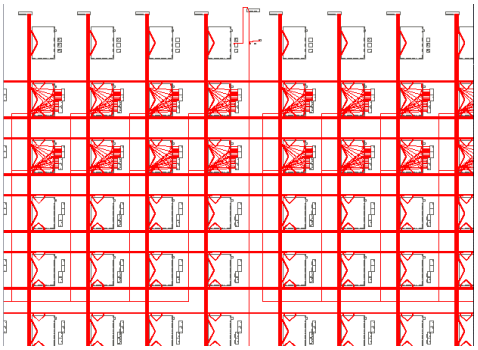

Figure 8 Testing circuit of hex lines across CLBs

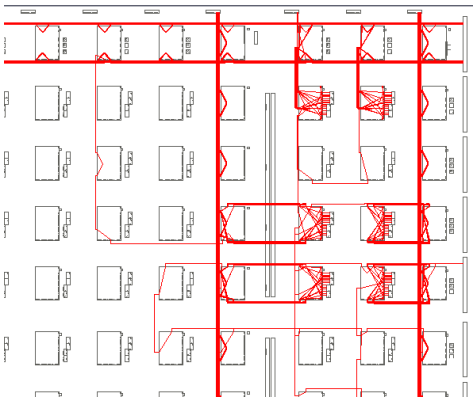

Figure 9 Testing circuitof hex lines not across CLBs

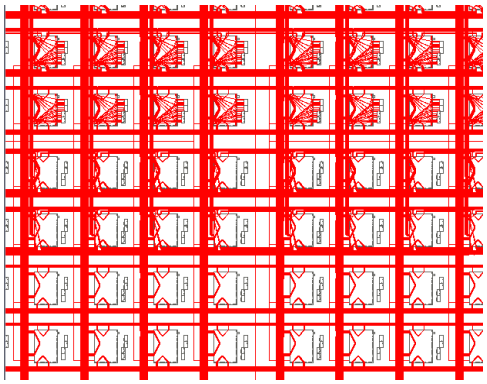

Figure 10 Testing circuit of double lines across CLBs

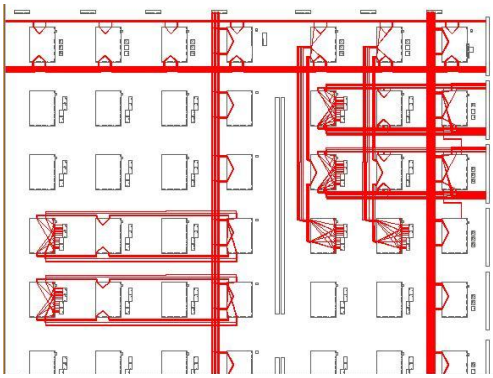

Figure 11 Testing circuit of double lines not across CLBs

Testing circuits of hex lines across CLBs, hex lines not across CLBs, double lines across CLBs and double lines not across CLBs were shown in figure 8, 9, 10 and 11. We could see figure 8, 9 and 10 showing the circuits as expected. Figure 11 showed a rather complicated circuit. This was because there were some open hex lines in Xilinx Spartan-3 XC3S400 FPGAs. When testing hex lines, signals should keep away from these open lines. Double lines could avoid these open lines, but hex lines could not connect with double lines. A method was turning hex lines to double lines through CLBs. So we got the testing circuits of figure 11.

VI. CONCLUSIONS

A XDL-based test configuration method was proposed and presented in the paper. The structure of the interconnect resources of Xilinx Spartan-3 XC3S400 FPGA which was chosen as an example was studied in detail. And the description of the interconnect resources using XDL was discussed also. According to the structure of the interconnect resources, a BIST structure was built to implement the testing of them. With only four configurations, the testing covered all of the hex lines and double lines. XDL programs were generated automatically through C++ programming which saved times remarkably. Placement and routing results demonstrated that the method could solve the problem of uncontrollability of the resources to be tested efficiently.

ACKNOWLEDGMENT

This work was supported by technical basis research projects in Science and Industry Bureau of China under Grant No.Z132012A001 and Z1320130013, and Natural Science Foundation of China under Grant No.61201028 and 61376042.

REFERENCES

[1] Niamat, M. Y., Arunjit Sahni, and M. M. Jamali. "A built in self test scheme for automatic interconnect fault diagnosis in multiple and single FPGA systems." Circuits and Systems, 2007. MWSCAS 2007. 50th Midwest Symposium on. IEEE, 2007.

[2] Rozkovec, Martin, Jiri Jenicek, and Ondrej Novak. "Application dependent FPGA testing method." Digital System Design: Architectures, Methods and Tools (DSD), 2010 13th Euromicro Conference on. IEEE, 2010.

[3] Tahoori, Mehdi Baradaran, et al. "Fault grading FPGA interconnect test configurations." Test Conference, 2002. Proceedings. International. IEEE, 2002.

[4] Sarker, Md Abdul Latif, and Moon Ho Lee. "Synthesis of VHDL code for FPGA design flow using Xilinx PlanAhead tool." Education and e-Learning Innovations (ICEELI), 2012 International Conference on. IEEE, 2012.

[5] Sun, Xiaoling, and P. I. E. T. E. R. Trouborst. "A unified global and local interconnect test scheme for Xilinx XC4000 FPGAs." Instrumentation and Measurement, IEEE Transactions on 53.2 (2004): 368-377.

[6] Tahoori, Mehdi Baradaran, and Subhasish Mitra. "Automatic configuration generation for FPGA interconnect testing." 2013 IEEE 31st VLSI Test Symposium (VTS). IEEE Computer Society, 2003.

[7] Lin, Teng, et al. "Application-dependent interconnect testing of Xilinx FPGAs." Solid-State and Integrated-Circuit Technology, 2008. ICSICT 2008. 9th International Conference on. IEEE, 2008.

[8] Liao, Y. B., et al. "Full coverage location of logic resource faults in A SOC co-verification technology based FPGA functional test environment." ASIC, 2009. ASICON'09. IEEE 8th International Conference on. IEEE, 2009.

[9] Beckhoff, Christian, Dirk Koch, and Jim Torresen. "The Xilinx Design Language (XDL): tutorial and use cases." Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), 2011 6th International Workshop on. IEEE, 2011.

[10] Spartan-3 Generation FPGA User Guide (v1.8). Xilinx Inc. June 13, 2011

[11] The Programmable Logic Data Book 2003, Xilinx Inc. San Jose, CA , 2003.

[12] Claus, Christopher, et al. "An XDL-based busmacro generator for customizable communication interfaces for dynamically and partially reconfigurable systems." Workshop on Reconfigurable Computing Education at ISVLSI. 2007.

[13] Das, N.; Roy, P.; Rahaman, H.; Dasgupta, P., "Build-in-Self-Test of FPGA for diagnosis of delay fault," Quality Electronic Design (ASQED), 2011 3rd Asia Symposium on , vol., no., pp.54,61, 19-20 July 2011

[14] Ruan, Aiwu, et al. "A Built-In Self-Test (BIST) system with non-intrusive TPG and ORA for FPGA test and diagnosis." Microelectronics Reliability 53.3 (2013): 488-498.

[15] Harris, Ian G., and Russell Tessier. "Testing and diagnosis of interconnect faults in cluster-based FPGA architectures." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 21.11 (2002): 1337-1343.