NTNU
Innovation and Creativity

Accurate Delay Test of FPGA Routing Network by Branched Test Paths

Elena Davydova Dikkanen

Master of Science in Electronics

Submission date:	September 2007
Supervisor:	Einar Johan Aas, IET
Co-supervisor:	Hideo Fujiwara, Nara Institute of Science and Technology, Japan Michiko Inoue, Nara Institute of Science and Technology, Japan

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

Problem Description

VLSI testing is performed to ensure that a designed integrated circuit meets customer requirements after it has been manufactured. The purpose of testing is to find defects that may have been introduced during the fabrication process.

Accuracy of test is directly related to the quality of a product, and has been receiving increasing attention in recent research. Particularly, a large amount of effort has been directed toward the capability of detecting small defects, because their number is expected to grow in the future technologies.

One of the difficulties with detection of small defects is that they do not always give functional errors. Very often only the delay of signals is increased. As a result, the defective circuit operates fault-free at lower frequencies, but fails at higher frequencies. Delay test is an efficient tool for detecting defects that cause delay faults.

The recently proposed branch-adding technique uses test paths with extra branches to improve detectability of defects in the stem of a test path. However, detectability of defects in the branches of a test path is not considered. As a result, the test accuracy in several parts of the circuit under test is unknown. This assignment involves the following:

- Examine accuracy of detection of defects tested by branches of a test path
- Develop a test method for the routing network of FPGA

Assignment given: 26. February 2007
Supervisor: Einar Johan Aas, IET

Abstract

This Master's thesis documents a new test method for detection of small delay faults in FPGA routing network. The main purpose of the test is accurate detection of faults in all parts of the network. The second aim is minimizing test application time.

The work of the thesis consisted of four parts. First, a literature study was performed to get background knowledge of FPGA architecture and basics of testing. Second, detection accuracy was defined and measured in SPICE for test paths with different number of fan-out. Third, test configurations were developed. And finally, detection accuracies for the proposed test method were calculated.

The SPICE measurements were performed on an interconnect model of FPGA. They revealed that detection accuracy of defects tested by branches of a test path is less than detection accuracy of defects tested by stems of a test path. In addition, it was observed that detection accuracy is best in the beginning of a test path.

In the proposed test method detection accuracy is improved by testing all segments outside switch matrices by test path stems, and applying test patterns to all bidirectional segments in both directions. A comparison to two previous test methods showed that the proposed test method is more accurate while keeping the same number of test configurations. The detection accuracy can be improved further by allowing more test configurations.

Acknowledgements

I thank **Prof. Einar Johan Aas** from the Norwegian University of Science and Technology, and **Professors Hideo Fujiwara** and **Michiko Inoue** from the Nara Institute of Science and Technology for supervision of this work. I also thank **Professors Trond Ytterdahl** and **Tajeshwar Singh** for their useful comments on the SPICE simulation setup and measurement.

In addition my gratitude goes to **Dr. Satoshi Ohtake**, **Tomokazu Yoneda** and all the **members of the Computer Design and Test Lab**. at Nara Institute of Science and Technology for their feedback on my work and for taking care of me during my stay in Japan.

Preface

This Master's thesis is the result of a research work carried out at Nara Institute of Science and Technology in Japan during the first half year of 2007. The research started as a continuation of an earlier project work on test of low-power dual-supply FPGAs, and the original aim was to develop a delay test that would take into consideration the variation of voltage in this type of FPGA. To understand the impact of voltage variations on circuit delay, it was planned to perform electrical simulations. However, that research met difficulties with acquiring electrical information about the dual-supply FPGA. The problem was redefined into accurate delay test for a general type FPGA in May 2007.

A paper has resulted from the new research work and was presented on Fault Tolerant Computing Workshop in Atagawa, Japan in July 2007, under the title ”Segment Delay Test in FPGA Interconnects Considering Fan-Outs”.

Trondheim, September 4th, 2007

Elena Davydova Dikkanen

Contents

Abstract I

Acknowledgements II

Preface III

Contents IV

1. Introduction 1
- 1.1 Motivation and scope of research 1
- 1.2 Contribution 1
- 1.3 Organization of the thesis 1

2. FPGA architecture and test 3
- 2.1 FPGA architecture 3
- 2.2 Defect categories 4
- 2.3 Delay fault models 5
- 2.4 Basics of delay testing 6
- 2.5 Accurate test by short test paths 7
- 2.6 Accurate test by branched test paths 7

3. Detection accuracy of branched test paths 8
- 3.1 Definition of detection accuracy 8
- 3.2 SPICE measurement 8
  - 3.2.1 SPICE setup 8
  - 3.2.2 22 nm predictive technology model (PTM) 9
  - 3.2.3 Results and discussion 10

4. New test method 14
- 4.1 Overview 14
- 4.2 Test configurations 15
- 4.3 Test procedure 16
- 4.3 Test procedure 17

5. Test quality of the new method 18
- 5.1 Fault coverage 18
- 5.2 Detection accuracy 18
- 5.3 Comparison with previous work 22
  - 5.3.1 Comparison with the test by short test paths [Peng 04] 22
  - 5.3.2 Comparison with the test by branched test paths [Tahoori 02] 23

6. Discussion 25

7. Conclusion 25

References 26

Appendix A 28

A1. 22 nm PTM transistor model: 28
A2. SPICE script 30
FPGA interconnect model 0. Simple test path with no extra branches. 30
FPGA interconnect model 1. Test path with 1 extra branch. 31
FPGA interconnect model 12. Test path with 2 transistors and 1 extra branch. 33
FPGA interconnect model 2. Test path with 2 extra branches. 36
FPGA interconnect model 22. Test path with 2 transistors and 2 extra branches. 39

1 Introduction

1.1 Motivation and scope of research

A Field Programmable Gate Array (FPGA) is a semiconductor device containing a large set of regularly distributed blocks of logic interconnected by routing network. Each logic block can be programmed to perform some function and each segment of the routing network can be programmed to provide a connection. By choosing different functions and connections, FPGA device can implement different electronic designs. As a result, FPGAs are being used in a wide range of commercial applications, in such areas as networking, digital signal processing, reliable computing and consumer electronics [Altera 07] [Xilinx 07].

The main trend in FPGA development has been to increase operating frequency and the number of programmable resources while reducing transistor sizes. This facilitates larger and faster designs with smaller end-system size. However FPGAs with small and tightly packed transistors are more vulnerable to defects in fabrication process. Small defects that don't cause problems at lower frequencies, fail FPGA at higher frequencies. To assure that FPGA operates correctly at higher frequencies, several tests have been proposed to detect small defects [Chmelar 03] [Peng 04] [Tahoori 02] [Gao 05]. All these works target defects in routing network of FPGA since it occupies as much as 80 % of FPGA area [Tahoori 04].

The common approach is to check the delay of the segment under test relative to the delay of the fault-free segment. This is done by applying a signal transition at one end of the segment under test and measuring signal value at the other end after the time period equal to the delay of the fault-free segment. Test clock period must allow small delay defects on the segment under test due to process variation. This acceptable delay margin determines the smallest delay fault that can be detected by the test. [Chmelar 03] introduced relative delay measurement by comparing delays of identically configured test paths. For absolute delay measurement, [Peng 04] pointed out that delay margin decreases with the length of the test path and suggested using short test paths. [Tahoori 02] observed that delay increase in segments with resistive open defects can be boosted by adding extra branches to the test path and proposed a test utilizing branched test paths. [Gao 05] checked the impact of branch-adding technique on resistive shorts, another important type of defects, and showed that adding extra branches doesn't affect detection of such defects. However, resistive open defects were found to be the major type of defects escaping tests [Needham 98], making branch-adding technique nevertheless attractive.

1.2 Contribution

In this thesis branch-adding technique has been developed further through examination of detection accuracy in the branches of a test path. Detection accuracy was defined as the smallest defect size that can be detected and measured in SPICE simulations on an FPGA interconnect model. Then, a test was developed with the best detection accuracy, as given by our simulations, in any segment between a logic block output and a logic block input. In contrast, [Tahoori 02] considered only detection accuracy of segments that can be tested by the stem of a test path.

1.3 Organization of the thesis

The thesis is organized as follows.

Chapter 2 gives the theoretical background for the delay testing of FPGA: the general architecture of FPGA, defects and fault models, an overview of delay testing and the related research.

Chapter 3, 4 and 5 present my contributions. Chapter 3 defines detection accuracy, shows the SPICE setup used for measurement of detection accuracy for different test paths and presents the simulation results.

Chapter 4 describes the proposed test method.

Chapter 5 evaluates fault coverage and detection accuracy of the proposed test method.

Chapter 6 discusses the overall results.

Finally, chapter 7 concludes the thesis.

2 FPGA architecture and test

This chapter gives an introduction to delay testing and the recently invented techniques for better test accuracy, and gives an overview of the target FPGA architecture for the test method proposed in this thesis.

2.1 FPGA architecture

Figure 1 shows the general architecture of symmetrical island-style SRAM FPGA. It employs an NxN array of logic blocks interconnected by a routing network. On the periphery, the array is surrounded by input/output blocks. The function of each logic block and connections in the routing network are selected by configuring SRAM memory cells.

The routing network consists of global and local routing resources. The global resources are comprised of parallel wiring channels in the horizontal and vertical directions that span the whole length and width of FPGA. The channels are made from lines interconnected by programmable switch matrices. The channel width between any two switch matrices is *n* lines.

The local routing resources connect logic blocks to the line segments in the channels. They include m lines at each side of a logic block and programmable connectors. There is also a buffer at each input and output of the logic block.

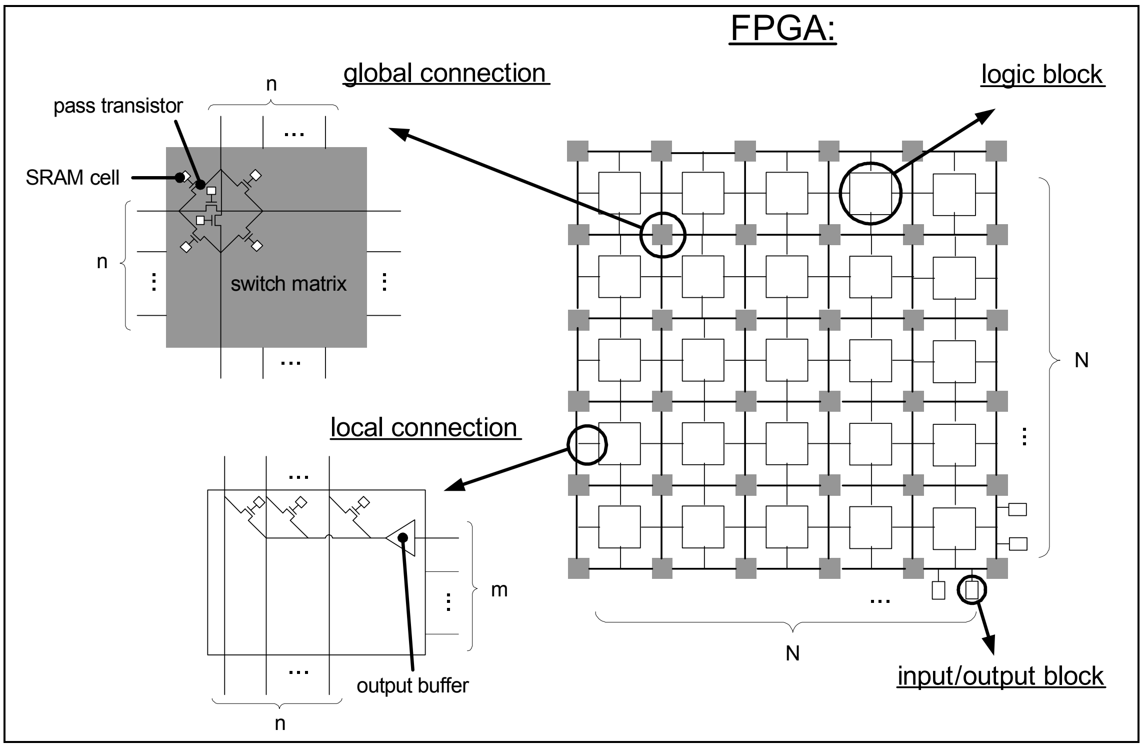


Figure 1. FPGA arcitecture

All programmable connections are realized by pass transistors controlled by SRAM cells. Any path from a logic block output to a logic block input starts at an output buffer, passes through some pass transistors and ends at an input buffer. A fan-out can be added to the path either at the local connectors or in the switch matrices. The maximum fan-out in a switch matrix is 3 branches.

Logic block is not the target of the proposed test. The functionality of the logic block is used for test generation and propagation of test responses. Figure 2 shows the functions employed in our test. Flip-flops apply test patterns to the test path and capture test responses. Look-Up Tables (LUTs) propagate test responses from logic block inputs to flip-flop inputs. LUTs configured as AND/OR gates combine test responses from several test paths. To simplify the problem it is assumed that each side of the logic block has inputs and outputs and that each output is a function of inputs of any side.

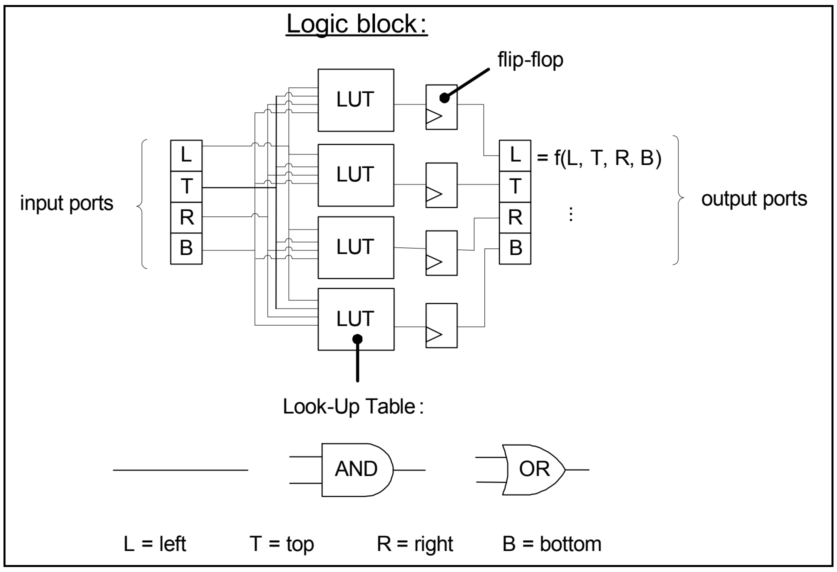


Figure 2. Logic block functionality

2.2 Defect categories

There are two important categories of defects in integrated circuits, including FPGAs, – bridge defects and open circuit defects [Hawkins 94] [Li 01].

Bridge defects include all defects that cause electrical connections across two or more circuit nodes that were designed to be disconnected, for example: physical shorts and ohmic shorts, gate oxide shorts, transistor punchthrough. Figure 3 shows a short in an interconnect layer caused by a particle. Gate oxide short is a short circuit between the gate and the channel or the gate and the drain/source diffusion zones of a CMOS transistor [Renovell 02]. Transistor punchthrough is a drain-to-source short that occurs when the drain depletion region extends across the entire channel length. Bridge defect detection is more efficient with I_{DDQ} test than delay test [Hawkins 94].

Open circuit defects cause unintentional electrical discontinuities for example due to missing metal, faulty patterning, improper etching, mask and fabrication errors. Figure 4 shows SEM[1] photographs of a contact open and a metallization open. Resistive open defects are partial open circuit defects, where a resistor exists between two nodes that should be connected [Li 01], see Figure 5. The value of defective resistance is denoted R_{DEF}. Resistive open defect with infinitely large resistance is identical to open circuit defect.

Resistive open defects are better detected by delay test than I_{DDQ} test [Li 01]. Opens are regarded as the major type of defects escaping tests [Needham 98].

[1] Scanning Electron Microscope

The routing network of FPGA is built from pieces of wire interconnected by pass transistors (sec. 2.1). Resistive open defects can occur both in the wire and in the transistors, and they cause delay of signals propagating through the routing network.

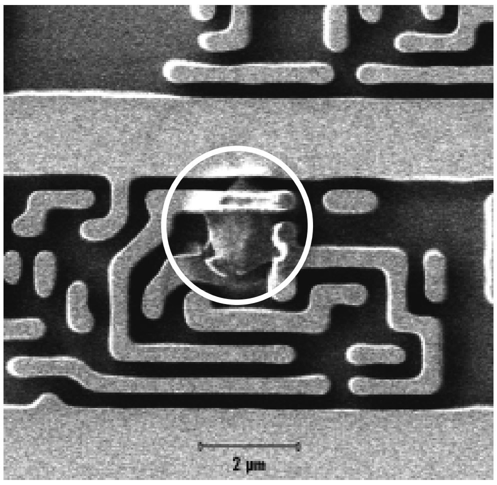


Figure 3. Short across several nets caused by a particle (www.micromagazine.com)

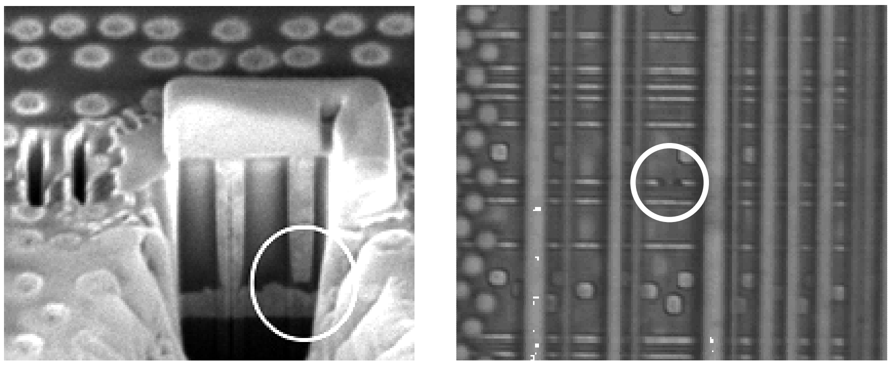


Figure 4. *Left*: incomplete contact etch (www.micromagazine.com); *right*: break on a net (www.ibm.com)

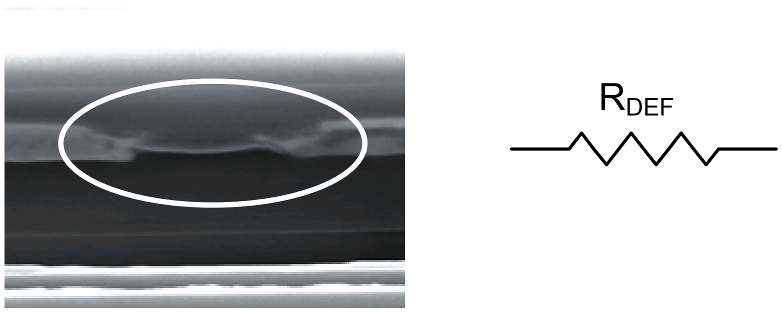


Figure 5. Closer view of the net break on Figure 4 and the circuit diagram symbol of the defect

2.3 Delay fault models

A fault model describes observable effects of defects that that can cause problems for the normal operation of the circuit. Delay fault models take into consideration the effect of defects on delay of signals in the circuit. Delay is a function of resistances and capacitances in the circuit. Resistive open defects increase signal delay by adding extra resistance to the signal path. If the delay increases so much, that the combinational delay of the circuit exceeds the clock period, the circuit fails at the rated frequency, and is said to have a delay fault. By checking delay in all parts of the circuit, defective circuits can be identified. Fault models are used to design tests and measure test quality.

Currently, the most widely known delay fault models include transition delay fault model, gate delay fault model, line delay fault model, path delay fault model and segment delay fault model [Krstic 98]. The first three fault models are used for representing defects located at a single gate. Path delay fault model and segment delay fault model have a more realistic assumption that defects affect several gates. However, path delay fault model produces too many paths that must be tested. As a result only the longest path for each signal is tested.

While in ASIC testing of the longest paths is enough for the verification of performance, in FPGA testing of the longest paths is not sufficient. In FPGA a short line might fail in a configuration if it determines the clock period and has untested defect [Peng 04]. Segment delay fault model is a trade-off between transition delay fault model and path delay model. It is not as good as path delay fault model when it comes to detection of distributed delays, but it gives less number of faults [Heragu 96].

The proposed test is based on segment delay fault model for FPGA [Peng 04]. This model considers slow-to-rise and slow-to-fall delay faults on combinational segments of FPGA. A slow-to-rise delay fault takes place when a low-to-high transition is applied to the beginning of a segment and it does not reach the end of the segment at the time of observation. Similarly, a slow-to-fall delay fault occurs when a high-to-low transition is applied to the beginning of a segment and it does not reach the end of the segment at the time of observation. In FPGA routing network there are three types of combinational segments: pass transistor, line and buffer.

The test of all segment delay faults guarantees that all segments in the routing network have acceptable delays. In turn, it assumes that also paths covering the tested segments are fault-free. This assumption doesn't hold if there are multiple small defects on the path that are not detected by individual segment tests, but combined give an unacceptable delay. However, the probability of this situation is small in practice [Peng 04].

2.4 Basics of delay testing

In electronic circuits the performance is determined by the time it takes for signals to propagate from one memory element to the other, or between a memory element and the input or output of the circuit. This time is called delay. If the signal doesn't arrive at the destination due to a defect, the circuit cannot operate at the designed frequency, and is described as having a delay fault (Figure 6).

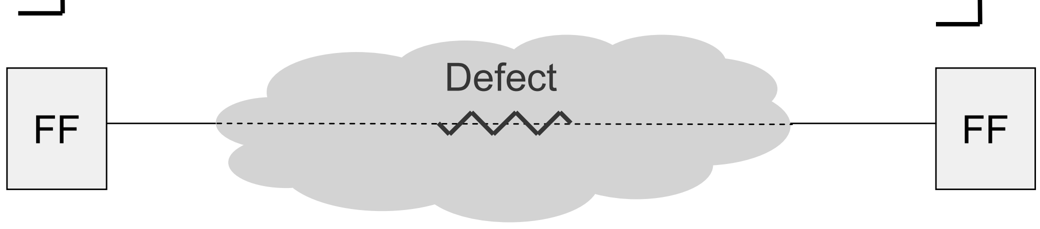


Figure 6. Delay testing

The objective of delay testing is to detect delay faults and ensure that the design meets the desired performance specifications. Delay faults are activated and observed by propagating signal transitions through the circuit [Krstic 98]. This requires application of two test values.

When circuits are designed, they have a delay margin. This means that they are clocked at a lower frequency, allowing that signals have small delays due to process variations. Delay margin is the allowable increase of the propagation time of a signal. A typical value is 10 % [Peng 04].

2.5 Accurate test by short test paths

In [Peng 04] it was shown that short test paths have smaller delay margin than longer test paths and can therefore detect smaller delay faults. A test method was proposed where all segments in FPGA routing network were tested by the shortest realizable test paths, i.e. paths that can be configured in FPGA and has a potential functional application.

2.6 Accurate test by branched test paths

In [Tahoori 02] detection of smaller defects was improved further by adding extra branches to the test path. A SPICE simulation study was performed on an FPGA interconnect model and the results showed that fan-out increases the delay ratio of the test path when defects are located in the stem of the path. Defects in the branches were not considered. However not all parts of FPGA routing network can be tested by stems. As Figure 7 indicates, segments inside switch matrices can only be tested by branches of a test path, if the path is the shortest realizable path. In test method proposed by [Tahoori 02] also some of the segments outside switch matrices are tested only by test path branches.

In this thesis work further SPICE simulations were performed and detection accuracy of the faults located in the branches of a test path was characterized. The details and results of the simulations are given in the next chapter.

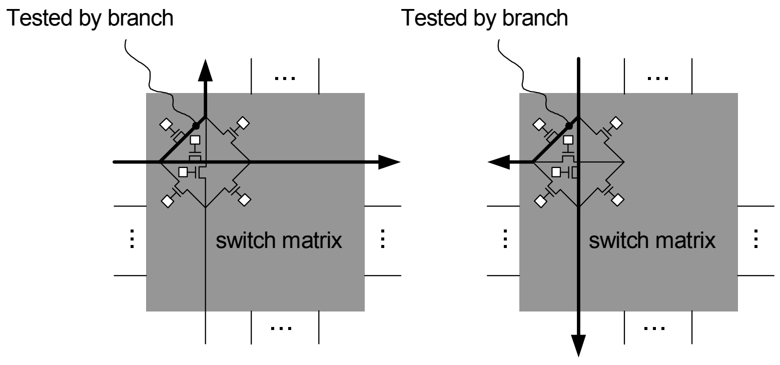


Figure 7. Test of segments inside a switch matrix

3 Detection accuracy of branched test paths

One of the main targets of this work is to examine the capability of branched test paths to detect small resistive open defects. This chapter introduces the definition of detection accuracy and presents the measurement of detection accuracy for short test paths with different number of branches. The results obtained here form the basis for the new test method described in the following chapters.

3.1 Definition of detection accuracy

In this thesis detection accuracy *A* of a test path is defined as the smallest defect size R_{DEF} that can be detected by the path.

When a segment under test contains a defect, the test path covering the segment becomes slower. Let $T(R_{DEF})$, T_M and T_{DO} be delay of the test path with defect of size R_{DEF}, the delay margin of the test path and the delay of the fault-free path, respectively. The delay fault is detected if the delay ratio of the faulty path $T(R_{DEF})/T_{D0}$ exceeds the delay margin ratio T_M/T_{D0}, i.e. the acceptable delay increase due to process variation and safety margins. Thus, the detection accuracy is identical to the defect size at which the delay of the path equals to the delay margin (E. 1):

$$A = min\left\{ R_{DEF} \middle| \frac{T(R_{DEF}) - T_{D0}}{T_{D0}} \geq T_M \right\} \qquad \textbf{E. 1}$$

3.2 SPICE measurement

Detection accuracy of the test path with no fan-out, one extra branch and two extra branches was studied by simulation in SPICE. Below the SPICE setup is explained, the important parameters used in the simulation are described and the measurement results are presented and discussed.

3.2.1 SPICE setup

Figure 8 shows the experimental setup. It represents a model of FPGA interconnect. A connection between two logic block is modelled as a line that goes through a pass transistor and connects two buffers. This conforms to the general architecture of FPGA explained in section 2.1. The pass transistor may be the transistor in the local connection or in the switch matrix.

Extra paths can be added to the line by switching on extra pass transistors. This gives the possibility to use different types of test paths. In this simulation, five types of test paths were studied – a simple test path with no extra branches, test paths with 1 short extra branch and 2 short extra branches, and test paths with 1 long extra branch and 2 longer extra branches. The reason of selecting short test paths is that they are reported to give more accurate delay fault detection, as mentioned earlier (sec. 2.5).

During the simulation, resistive open defects were added as extra resistors, one at a time, to three different locations: (1) - in the stem of the test path, (2) – in the branch of the test path just after the fan-out point, (3) – in the branch of the path after the pass transistor.

Low-to-high and high-to-low voltage transitions were applied to the buffer at the beginning of the line and the delay was measured at the outputs of branch buffers. Delay margin value of 10 % was selected as in [Peng 04]. The detection accuracy was found by adjusting defect resistance until the increase of the delay of the path became equal to the delay margin.

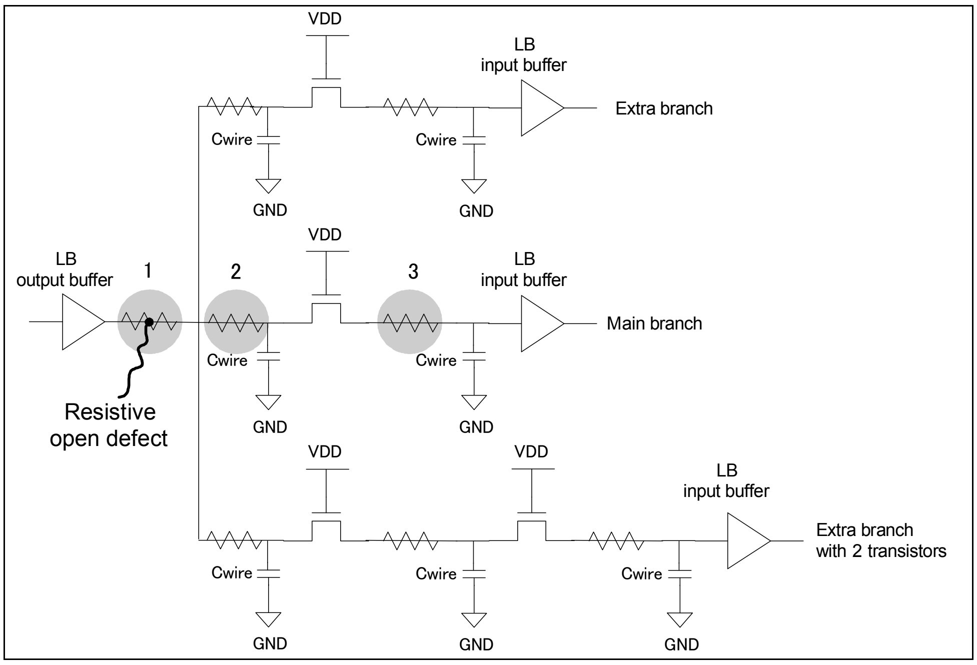


Figure 8. SPICE setup and defect location (LB = logic block)

This interconnect model is similar to the one used in previous work [Tahoori 02]. Note that wire resistance is neglected. Wire capacitance in the branches is modelled separately for each segment, so that the detection accuracy of the branches can be measured more accurately. Capacitor C_{wire} represents total capacitance in each segment. The same capacitance value is used for all segments. All transistors are chosen to have equal sizes and are modelled by the newest 22nm predictive technology model, which is described in the next section. During the simulation, the circuit was powered by 0.8 V supply voltage in accordance with the predicted values for 22nm technology in [ITRS 06]. HSPICE v.2003.03-SP1 simulator was used. Refer to Appendix 1 for SPICE description of the interconnect model and the measurement.

3.2.2 22 nm predictive technology model (PTM)

A transistor model is an important parameter for simulations on electrical level. It describes the electrical behaviour of the transistor, in particular such characteristics as resistance and capacitance that affect the circuit delay. This section gives a brief overview of the 22 nm Predictive Technology Model (PTM), the transistor model that forecasts characteristics of the next generation nanoscale transistors (Appendix 1).

Predictive Technology Model (PTM) was developed by BSIM group at the University of California in Berkeley and the Arizona State University, who also developed BSIM4, the current standard model for MOSFET compatible with all commercial circuit simulators [PTM 2007].

In PTM, ten parameters that determine the major behaviour of a MOSFET during technology scaling are determined by literature survey, such as ITRS[2], and by estimation. They include parameters specified by technology – supply voltage (V_{dd}), oxide thickness

[2] International Technology Roadmap for Semiconductors

(T_{oxe}), effective gate length (L_{eff}), drain and source resistance (R_{dsw}), and threshold voltage (V_{th0}); process parameters – channel doping (N_{ch}) and drain-induced barrier lowering parameter (E_{ta0}); and physical parameters – body effect (K_1), low field carrier mobility (μ_0), and the saturation velocity (V_{sat}). Other parameters that have observable impact on IV characteristics are fit from experimental data for each technology node. Parameters for CV characteristics are predicted based on the current BSIM models [Zhao 06].

In other words, the PTM transistor model is not based on the real process data for 22-nm technology, but uses an extrapolation from the current transistor models. It was selected because it gives an opportunity to evaluate the delay behaviour of the next generation of transistors. However it should be kept in mind that there may be deviations of the real detection accuracy data from those measured here.

3.2.3 Results and discussion

Figure 9, Figure 10 and Figure 11 show the measured delay ratios and detection accuracies for test paths with no fan-out, one extra branch and two extra branches. Both low-to-high and high-to-low transitions are included. There is a difference in measured delay ratios for the two transitions. One possible reason for the difference is the use of equal sizes for both p- and n-transistors in waveform generator, input and output buffers. P-transistors are slower than n-transistors. During low-to-high transition more p-transistors are active resulting in more delay and less delay ratio in defective circuit. Location of the active p-transistors might also contribute to the difference.

For defects in the stem of the test path (Figure 9) it can be seen that adding extra branches increases detection accuracy, in conformity with the results in [Tahoori 02]. Test path with 3 branches is the most optimal. However adding longer branches gives less increase in delay ratio than adding shorter branches.

For defects in the branches (Figure 10 and Figure 11), detection accuracy decreases as the distance from the fan-out point becomes larger. When the defect is located in 3, the detection accuracy is the lowest. Best detection accuracy is achieved for the test path with 1 branch. This result indicates that when the number of branches increases, the detection accuracy in the branches might decrease.

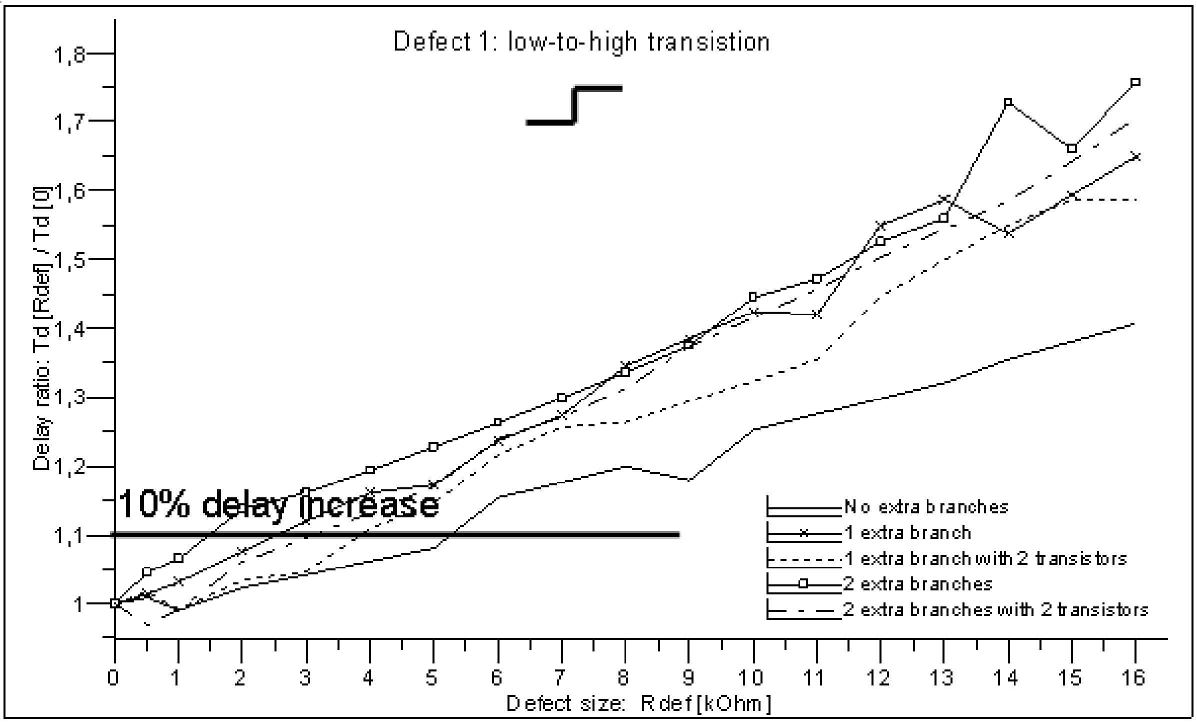
Another interesting observation can be made for the case of 1-branch test path: the detection accuracy is the highest for defects located in the beginning of the path. This means that detection accuracy of test paths can be improved by testing each segment in both directions. In this way, defects located in the end of a test path (location 3) will be retested such that they appear in the beginning of a test path (location 1 or 2). Table 1 shows the improvement of detection accuracy in bidirectional test based on the measured values of detection accuracy.

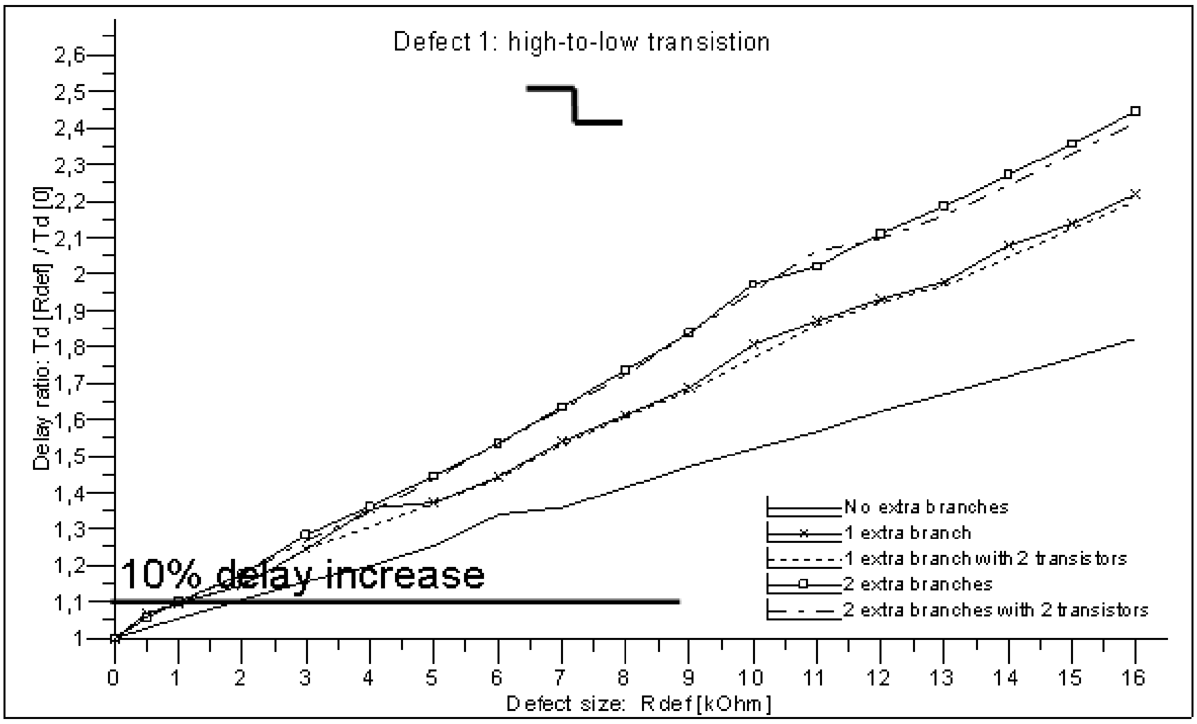
There is a non-monotone increase in delay ratio on Figure 9, Figure 10 and Figure 11 that makes some of the data difficult to compare. The reasons for these variations could not be identified due to the time limit for this project. However, some noise in the data is probably caused by the non-balanced sizes of n- and p- transistors. The delay measurement technique in the SPICE script (see Appendix 1) could also be re-evaluated.

Table 1. Improvement of detection accuracy in bidirectional test

Movement of defect location on Figure 8	1-branch test path		2-branch test path		3-branch test path	
	LH	HL	LH	HL	LH	HL
$3 \rightarrow 2$	14,3 %	66,7 %	0 %	66,7 %	0 %	77,8 %
$3 \rightarrow 1$	14,3 %	66,7 %	57,1 %	66,7 %	66,7 %	88,9 %

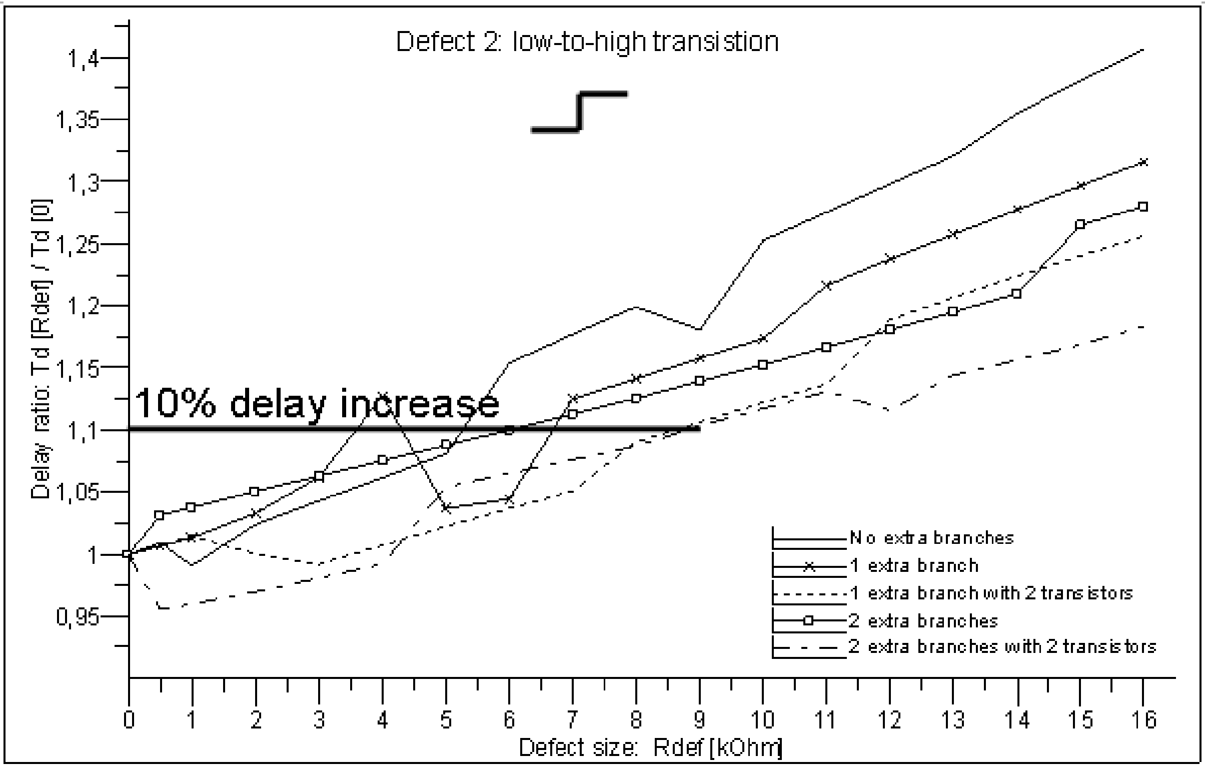
LH = low-to-high transition. HL = hihg-to-low transition

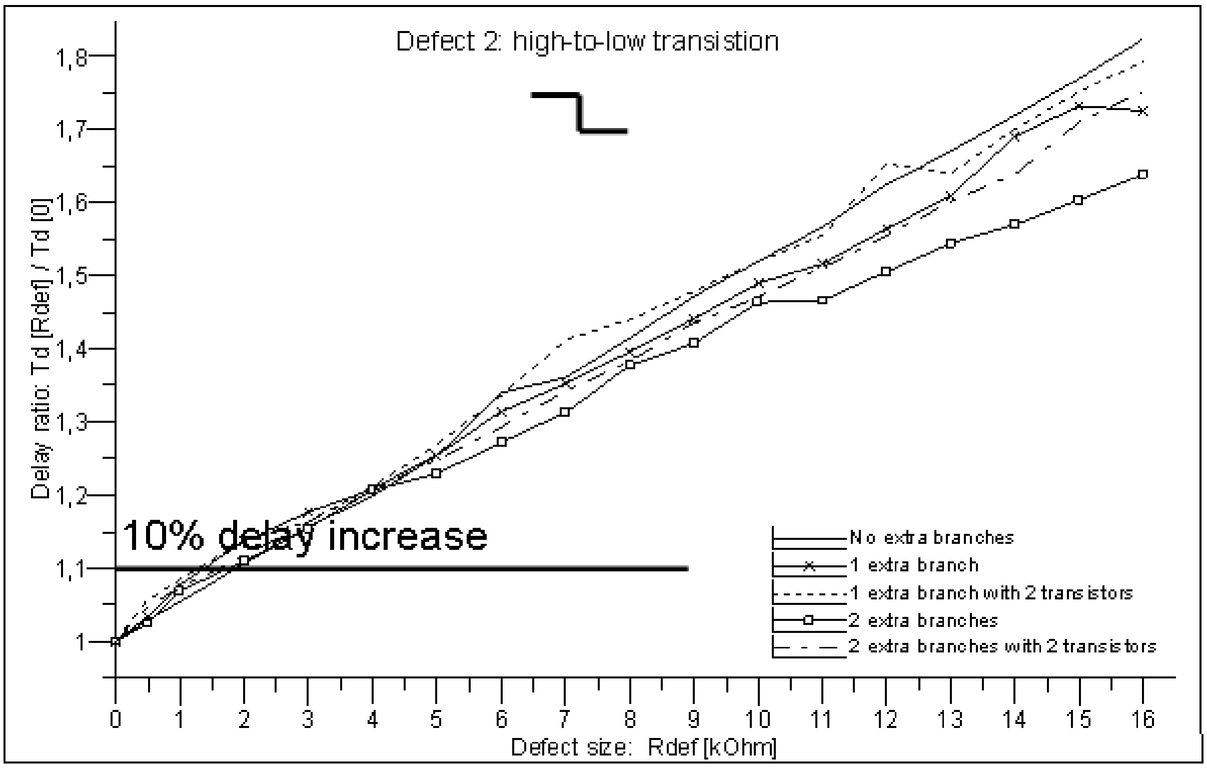




Test path type	LH detection accuracy [kOhm]	HL detection accuracy [kOhm]
1 branch	6	2
2 branches	3	2
3 branches	2	1

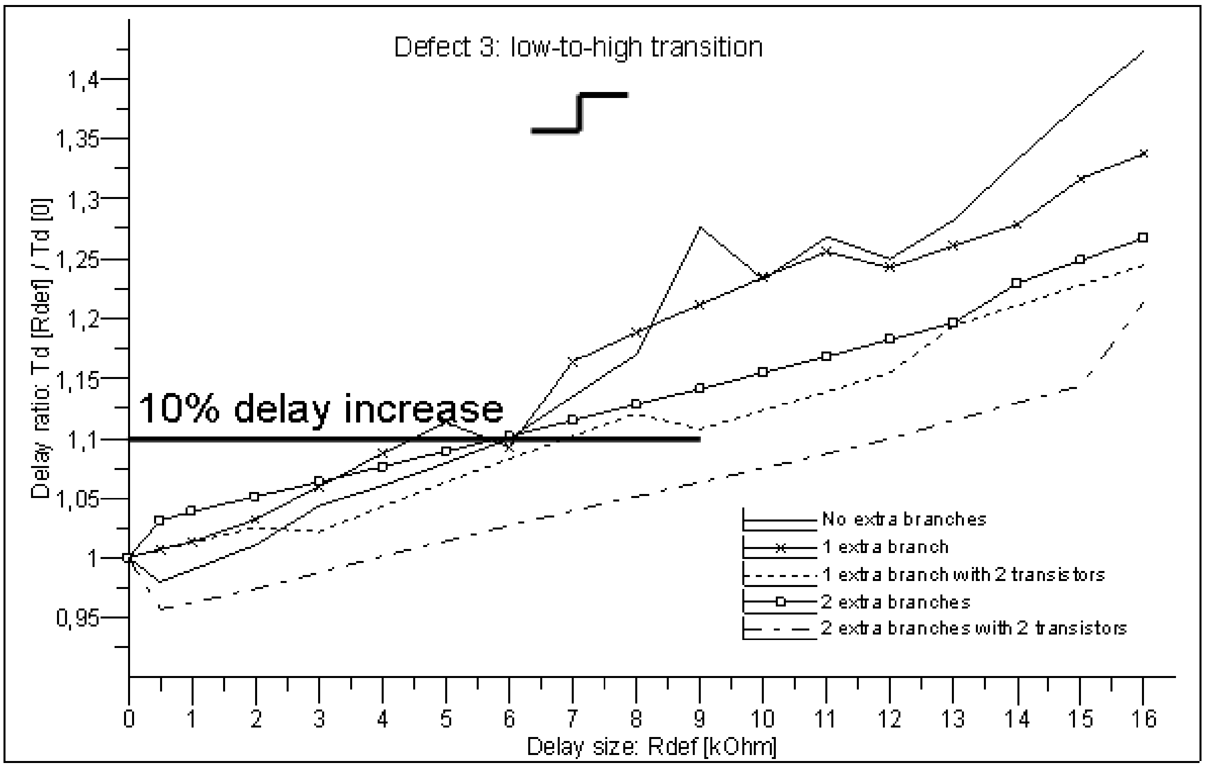
Figure 9. Dependence of delay ratio on additional branches (defect 1)

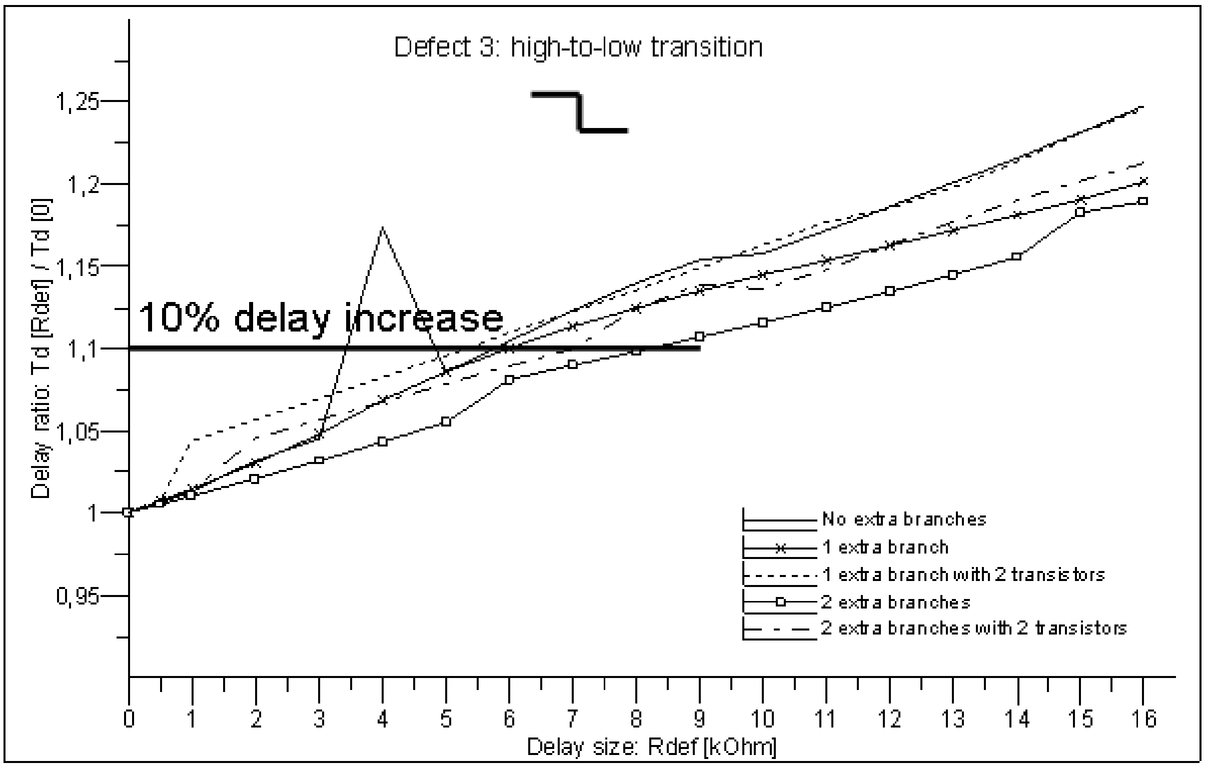




Test path type	LH detection accuracy [kOhm]	HL detection accuracy [kOhm]
1 branch	6	2
2 branches	7	2
3 branches	6	2

Figure 10. Dependence of delay ratio on additional branches (defect 2)





Test path type	LH detection accuracy [kOhm]	HL detection accuracy [kOhm]
1 branch	7	6
2 branches	7	6
3 branches	6	9

Figure 11. Dependence of delay ratio on additional branches (defect 3)

4 New test method

The purpose of this work is to find an accurate test for delay faults in FPGA routing network. In this chapter the ideas in previous work – use of test paths with short length (sec. 2.5) and additional branches (sec. 2.6) are combined to form a basis for a new test method. The measurement of detection accuracy in the preceding chapter showed its dependency on the location of a defect. Based on this result, the test method is then developed further by utilizing bidirectional test.

In the second part of this chapter, configurations of FPGA for test are proposed and test procedure is explained.

4.1 Overview

The main idea of the new test method is to use short test paths with 1, 2 or 3 branches to test all segments between logic blocks of FPGA. As mentioned in sec. 2.5, short test path has a smaller delay margin (slack) than a long test path and therefore can detect smaller delay faults. Test paths with extra branches further improve the detection of small delay faults by introducing increased path capacitance (sec. 2.6).

In FPGA interconnect, the shortest reconfigurable test path starts at the output of a flip-flop in a logic block and ends at the input to a flip-flop in the neighbouring block. Figure 12a, Figure 12b and Figure 12c show examples of the shortest test paths with 1, 2 and 3 branches, respectively. Each test path goes through one switch matrix, where maximum 3 fan-out branches can be connected, as explained in sec. 2.1. Thus, the test paths on Figure 12 are the shortest test paths with all possible number of branches.

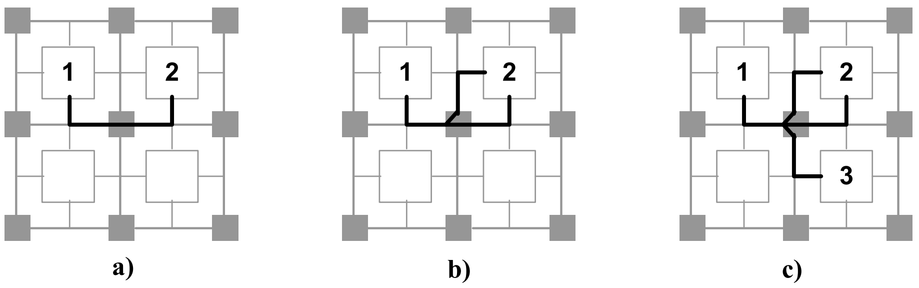


Figure 12. Test paths with: a) 1 branch, b) 2 branches and c) 3 branches

SPICE measurement in the previous chapter showed that detection accuracy of a test path depends on the location of the defect: smaller defects can be detected in the beginning of a test path than in the end of the test path. In the proposed test method, each segment in FPGA is tested in both directions.

When a test path is configured in an FPGA, it can detect any delay fault located in its stem or one of the branches, if the size of the fault is bigger than minimum detectable size. The detection is done by applying a signal transition at the input of a test path and measuring signal value at the output(s) of the test path. In Figure 12 the input of all test paths is the output of a flip-flop in logic block 1, while the outputs of the test paths are flip-flop inputs in logic blocks 2 and 3.

To reduce the total number of test configurations, as many segments as possible must be tested in the same test configuration. The detailed steps of how it was done are explained below. Because test paths with more branches are more difficult to pack, more test configurations are needed for the case of 2-branch and 3-branch test paths.

4.2 Test configurations

To find the minimum number of test configurations for each type of test path, first the requirement of complete test coverage was fulfilled, i.e. the requirement that all segments in FPGA routing network must be tested. This was done by starting with finding the complete test for the segments inside a switch matrix, separately for the case of 1-branch, 2-branch and 3-branch test paths. Because any test path goes through a switch matrix, this approach usually gives almost complete test coverage also for the segments outside switch matrices [Renovell 98] [Peng 04] [Tahoori 03].

Next, all switch matrices of the FPGA were identically configured and the maximum number of switch matrices that can be tested at the same time was identified. This number was limited by the number of available lines in the routing network of FPGA and represented one test configuration. The procedure was then repeated for the remaining switch matrices and switch matrix configurations until they all were covered. Complete segment coverage was verified by drawing a diagram of segments tested in each test configuration and superimposing the diagrams of all test configurations (sec. 5.1).

As previously mentioned, the stem of a test path has better accuracy than its branches. To achieve more accurate detection, the test was extended such that all segments were tested in both directions. This was done by rotating and mirroring the initial test configurations for the 2-branch and 3-branch test paths. In the case of 1-branch test path, no extra configurations were needed for bidirectional test, since the test path is symmetric. Only test procedure was altered to realize bidirectional test for 1-branch case (see sec. 0).

In contrast to [Peng 04], test paths were connected together in chains through logic blocks, such that test patterns could be applied at primary inputs of FPGA and test results observed at its primary outputs. Refer to sec. 0 for comparison of the proposed method to previous test methods.

Finally, it was verified that each port of a logic block was tested both as input and as output (see sec. 5.1).

Figure 13, Figure 15 and Figure 14 show the resulting test configurations for 1-branch, 2-branch and 3-branch test paths, respectively.

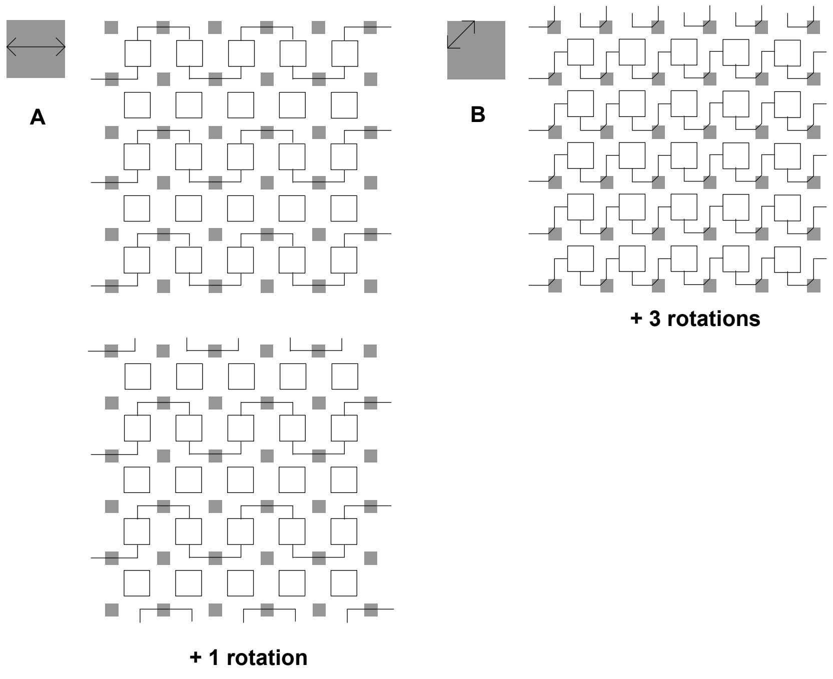


Figure 13. Test configurations for 1-branch test path

It is assumed that logic blocks with one input implement the identity function, and logic blocks with two inputs implement the AND function during rising delay fault test, and the OR function during falling delay fault test.

In total, 8 test configurations were achieved for the 1-branch case, 32 test configurations for the 2- and case and 32 test configurations for the 3-branch case.

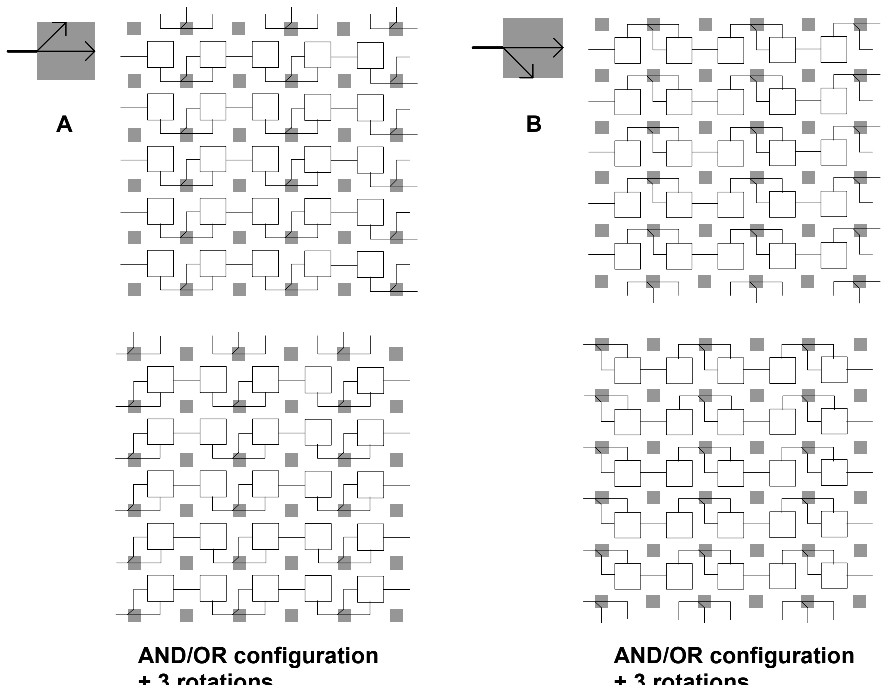


Figure 15. Test configurations for 2-branch test path

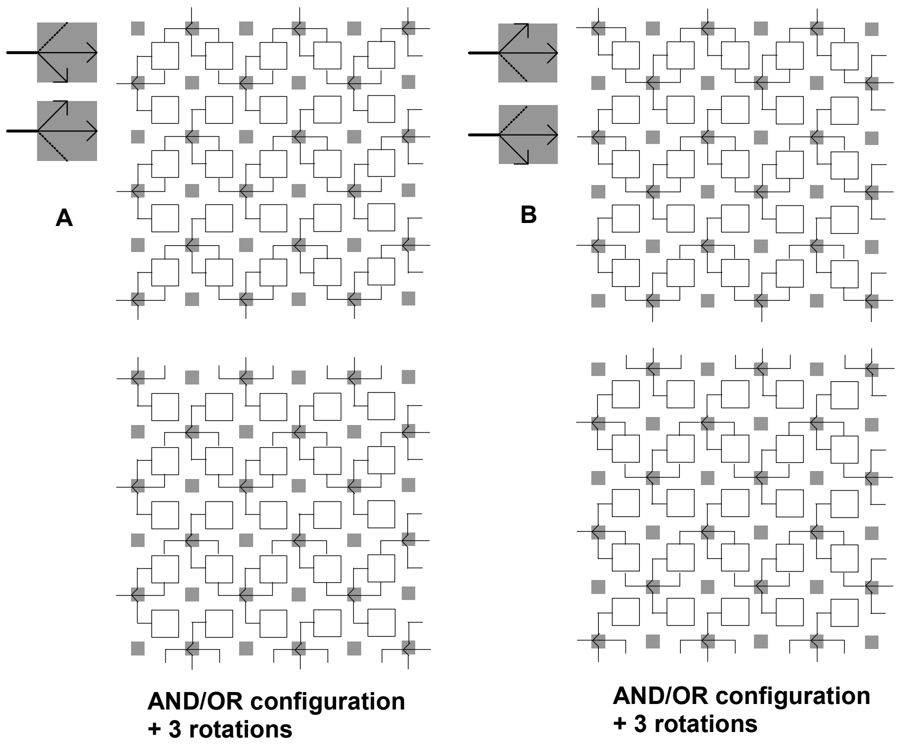


Figure 14. Test configurations for 3-branch test path

4.3 Test procedure

Figure 16 shows pseudo code for testing of FPGA with 1-branch test path. Figure 17 shows pseudo code for testing of FPGA with 2-branch or 3-branch test path.

```
1-branch test path:
1.  LOAD in test configuration
2.  RESET all flip-flops to zero
3.  APPLY rising transition at the primary inputs connected to the beginning of test chains
4.  TOGGLE test clock N times
5.  CHECK values at the primary outputs connected to the end of test chains
6.  APPLY falling transition at the primary inputs connected to the beginning of test chains
7.  TOGGLE test clock N times
8.  CHECK values at the primary outputs connected to the end of test chains
9.  REPEAT the test in the opposite direction by using outputs as inputs, and visa versa.
```

Figure 16. Pseudo code for testing of FPGA with 1-branch test path

```
2-branch and 3-branch test path:
1.  LOAD in AND-configuration
2.  RESET all flip-flops to zero
3.  APPLY rising transition at the primary inputs connected to the beginning of test chains
4.  TOGGLE test clock N times
5.  CHECK values at the primary outputs connected to the end of test chains
6.  LOAD in OR-configuration
7.  SET all flip-flops to 1
8.  APPLY falling transition at the primary inputs connected to the beginning of test chains
9.  TOGGLE test clock N times
10. CHECK values at the primary outputs connected to the end of test chains
```

Figure 17. Pseudo code for testing of FPGA with 2-branch or 3-branch test path

5 Test quality of the new method

This chapter evaluates the quality of the proposed test method, in particular fault coverage and detection accuracy, and compares the method with previous work.

5.1 Fault coverage

The proposed test configurations cover all segments and switch matrix directions. To observe this, consider Figure 18 which shows tested segments after each configuration, for the 1-branch case. Covered segments are marked with black colour. Covered segments inside switch matrices are shown on the magnified switch matrix above each drawing. All segments outside switch matrices are tested already after 4 test configurations. The remaining test configurations exercise all the uncovered segments inside switch matrices, giving complete test coverage. Similar diagrams can be drawn for the 2-branch case and the 3-branch case.

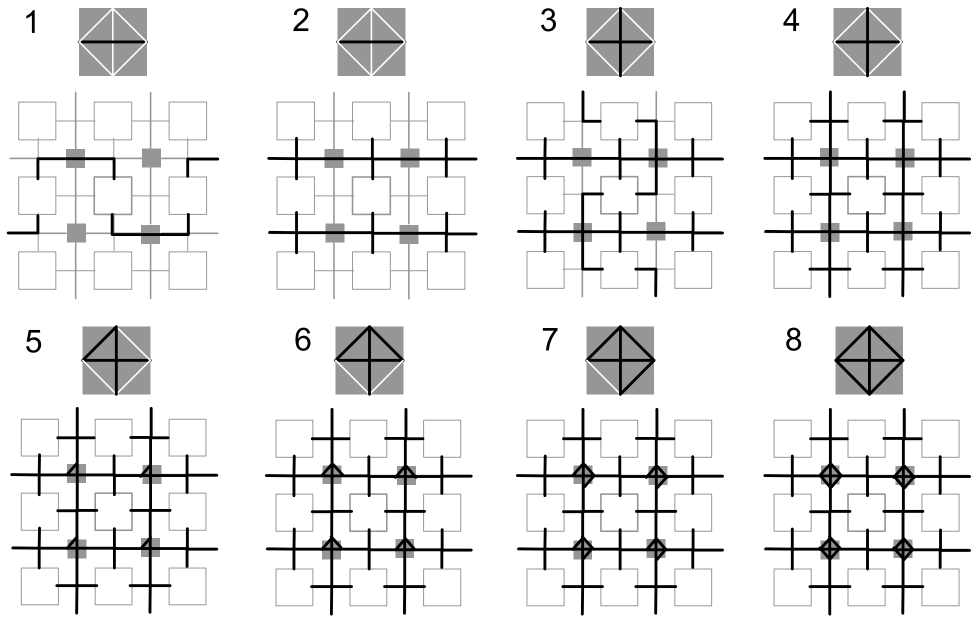


Figure 18. Tested segments after each test configuration, 1-branch test path

The test configurations include test of inputs and outputs on each side of a logic block.

5.2 Detection accuracy

In the proposed test method, all segments outside switch matrices are tested with the detection accuracy of the beginning of a test path. All segments inside switch matrices are tested with the detection accuracy of the middle of the test path.

Experimental data for the interconnect model on Figure 8 are summarized in Table 2. The data shows that 1-branch and 3-branch test paths can detect resistive open defects with the size of 6 kΩ and larger, while 2-branch test path can detect resistive open defects from 7 kΩ, for 22nm process technology.

Table 2. Detection accuracy for the interconnect model on Figure 8

Test path type	Beginning of a test path (Defect location 1)		Middle of a test path (Defect location 2)		End of a test path (Defect location 3)	
	LH [kΩ]	HL [kΩ]	LH [kΩ]	HL [kΩ]	LH [kΩ]	HL [kΩ]
1 branch	6	2	6	2	7	6
2 branches	3	2	7	2	7	6
3 branches	2	1	6	2	6	9

Detection accuracy values in the Table 2 depend on process technology and the electrical and dimensional parameters of wires and transistors in FPGA routing network: capacitance, resistance, wire width and breadth, transistor sizes, etc. A more accurate estimate of the detection accuracy of each test path can be found using theoretical approximations of delay in pass transistor networks. The two widely used tools are the Elmore delay formula and Penfield-Rubinstein-Horowitz theorem [Rabaey 96].

Elmore delay formula expresses the time constant of a cascaded *N*-stage RC chain. Consider 1-branch shortest test path of Figure 12a): the test path goes through three nodes – first through the pass transistor in the local connection at the output of logic block 1, then via a pass transistor in a switch matrix and finally through a pass transistor at the input of logic block 2. We replace each transistor by a resistor with the value equal to its on-resistance and model the test path as a 3-stage RC chain (Figure 19). We assume that all resistors are equal and all capacitors are equal.

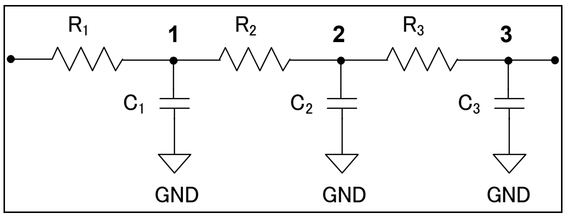


Figure 19. 3-stage RC chain model of 1-branch test path

$$\tau_N = \sum_{k=1}^{Ni} C_k \sum_{m=1}^{k} R_m$$

$$N = 3$$

$$\tau_{1-branch} = C_1R_1 + C_2(R_1 + R_2) + C_3(R_1 + R_2 + R_3)$$

$$C_k = C \,\forall k,$$

$$R_m = R \,\forall m$$

$$\underline{\tau_{1-branch} = 6RC}$$

Using Elmore delay formula we then calculate the time constant for the 1-branch test path: The time constant in the presence of defects increases to:

$$\tau_{1-branch,defect\,1} = 6RC + 3R_{DEF}C$$

$$\tau_{1-branch,\,defect\,2} = 6RC + 2R_{DEF}C$$

$$\tau_{1-branch,\,defect\,3} = 6RC + R_{DEF}C$$

Given the delay margin of 10 %, the detection accuracy for 1-branch test path becomes:

$$A_{1-branch} = min\left\{ R_{DEF} \left| \frac{\tau(R_{DEF}) - \tau_{1-branch}}{\tau_{1-branch}} \geq T_M \right. \right\}$$

$$\frac{\tau(R_{DEF}) - \tau_{1-branch}}{\tau_{1-branch}} \geq 10\%$$

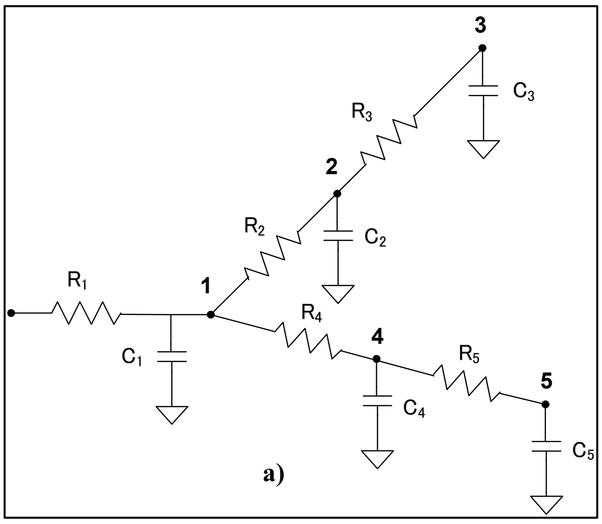
$$\frac{6RC + xR_{DEF}C - 6RC}{6RC} \geq 10\%$$

$$R_{DEF} \geq \frac{0,6R}{x}$$

$$A_{1-branch} = \frac{0,6R}{x} \quad , where \quad x = \begin{cases} 3 & for\ defect\ location\ 1 \\ 2 & for\ defect\ location\ 2 \\ 1 & for\ defect\ location\ 3 \end{cases}$$

Penfield-Rubinstein-Horowitz theorem is a generalization of Elmore delay formula and can be used on an RC tree-network with *N* nodes. We use this theorem to calculate time-constants for the shortest test path with 2 branches (Figure 12b) and the shortest test path with 3 branches (Figure 12). The 2-branch test path includes 5 nodes – local transistor at the output of logic block 1, two pass transistors in the switch matrix, a pass transistor at the input of logic block 1 and a pass transistor at the input of logic block 2. The 3-branch test path includes 7 nodes. Also here we assume equal values for all resistors and all capacitors.

The RC models of the test paths are shown on Figure 20.



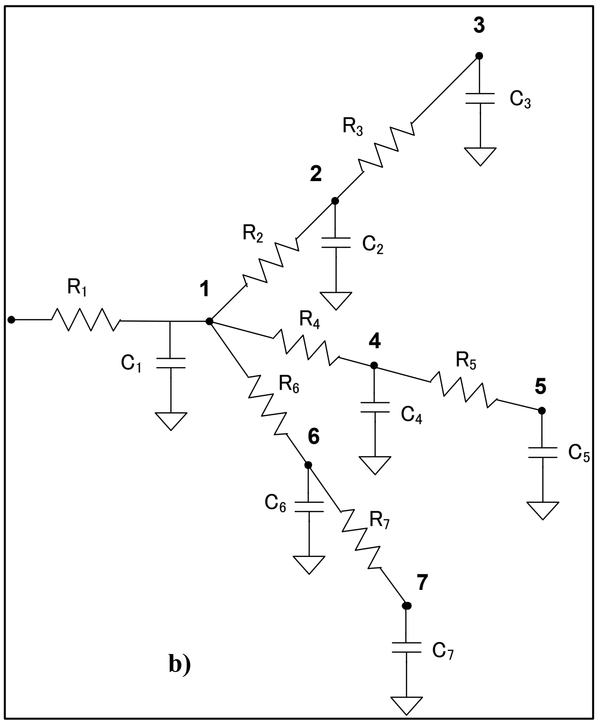


Figure 20. RC models of a) 2-branch test path and b) 3-branch test path)

Time constant for the 2-branch test path is calculated as follows:

$$\tau_i = \sum_{k=1}^{N} C_k V_k(0) R_{i,k}$$

$$N = 5$$

$$i = 5$$

$$\tau_{2-branch} = R_1C_1 + R_1C_2 + R_1C_3 + (R_1 + R_4)C_4 + (R_1 + R_4 + R_5)C_5$$

$$R_1 = R_2 = \ldots = R_5 = R$$

$$C_1 = C_2 = \ldots = C_5 = C$$

$$\underline{\tau_{2-branch} = 8RC}$$

In the presence of defects the time constant increases to:

$$\tau_{2-branch,defect\,1} = 8RC + 5R_{DEF}C$$

$$\tau_{2-branch,defect\,2} = 8RC + 2R_{DEF}C$$

$$\tau_{2-branch,defect\,3} = 8RC + R_{DEF}C$$

Given the delay margin of 10 %, the detection accuracy for 2-branch test path becomes:

$$\underline{\underline{A_{2-branch} = \frac{0{,}8R}{x} \quad \text{, where} \quad x = \begin{cases} 5 & \text{for defect location 1} \\ 2 & \text{for defect location 2} \\ 1 & \text{for defect location 3} \end{cases}}}$$

Time constant for the 3-branch test path is calculated in the similar way:

$$N = 7$$

$$i = 5$$

$$\tau_{3-branch} = R_1C_1 + R_1C_2 + R_1C_3 + (R_1 + R_4)C_4 + (R_1 + R_4 + R_5)C_5 + R_1C_6 + R_1C_7$$

$$R_1 = R_2 = \ldots = R_7 = R$$

$$C_1 = C_2 = \ldots = C_7 = C$$

$$\underline{\tau_{3-branch} = 10RC}$$

In presence of defects the time constant increases as follows:

$$\tau_{3-branch,defect\,1} = 10RC + 7R_{DEF}C$$

$$\tau_{3-branch,defect\,2} = 10RC + 2R_{DEF}C$$

$$\tau_{3-branch,defect\,3} = 10RC + R_{DEF}C$$

Given the delay margin of 10 %, the detection accuracy for 3-branch test path becomes:

$$\underline{\underline{A_{3-branch} = \frac{R}{x} \quad \text{, where} \quad x = \begin{cases} 7 & \text{for defect location 1} \\ 2 & \text{for defect location 2} \\ 1 & \text{for defect location 3} \end{cases}}}$$

Table 3 shows calculated detection accuracy per resistance for the three test paths. The detection accuracies calculated in this chapter show the same tendencies as the measured detection accuracies in Table 2, but give a more accurate relation between the number of branches, the location of defects and the detection accuracies.

Table 3. Detection accuracy per Ω resistance (A) in the routing network of FPGA

Test path type	Defect location 1 (R_1)	Defect location 2 (R_4)	Defect location 3 (R_5)
1 branch	0,2000	0,3000	0,6000
2 branches	0,1600	0,4000	0,8000
3 branches	0,1429	0,5000	1,0000

5.3 Comparison with previous work

In this section, the proposed test method is compared to two previous test methods – one which uses only short test paths (sec. 2.5) and one that exploits branch-adding technique but uses longer test paths (2.6).

5.3.1 Comparison with the test by short test paths [Peng 04]

In this method all segments of the routing network are tested by the shortest test paths between two flip-flops. This is equivalent to the 1-branch test path in the proposed test method (Figure 12a). However test patterns are applied only in one direction. As a result, around 30 % of the routing network is tested with detection accuracy of defect location 3 equal to 0,6 R (Table 3). For complete test of an XC4000 like architecture having *n* = 8 global lines and 2 local input lines, 48 test configurations are required.

In order to compare the new test method with the test by short test paths, the proposed test

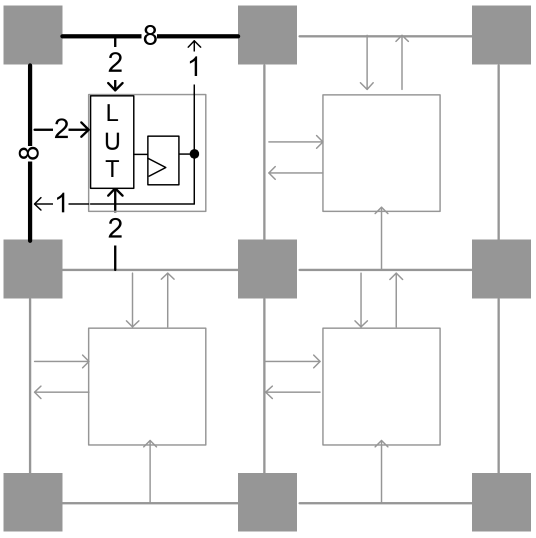


Figure 21. XC4000 architecture

configurations must be adapted to fit the XC4000 architecture (Figure 21). The main difference between this architecture and the general FPGA architecture of Figure 1 is the location of inputs and outputs of the logic block. Taking into consideration this difference, test configurations are modified as shown on Figure 22. The total number of test configurations is 48. All segments except those in the unidirectional input of the logic block (13 %) and in one of the switch matrix directions (9 %), are tested in both directions, with detection accuracy more than 0,3R (Table 2). Thus, in the proposed test, 8 % of all segments in the routing network are tested at 50 % better detection accuracy using the same number of test configurations.

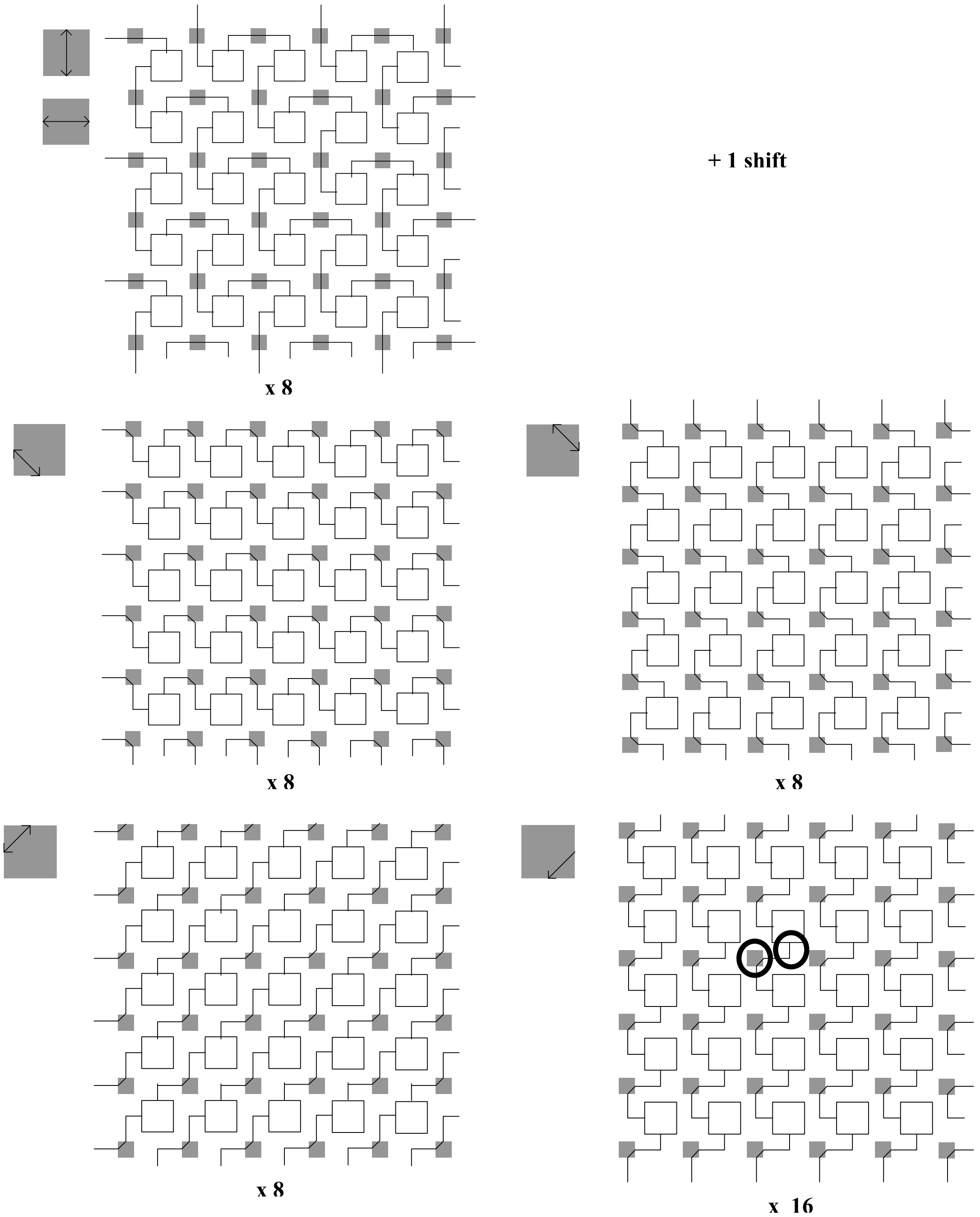


Figure 22. Test configurations for XC4000 FPGA architecture (*encirled:* examples of segments that are tested in only one direction)

5.3.2 Comparison with the test by branched test paths [Tahoori 02]

In previous test method based on branch-adding technique, test paths passed 6 switch matrices [Tahoori 03]. Also the shape of the branched test paths was different: the branches were added at different switch matrices. Figure 23 shows an RC model of the test path.

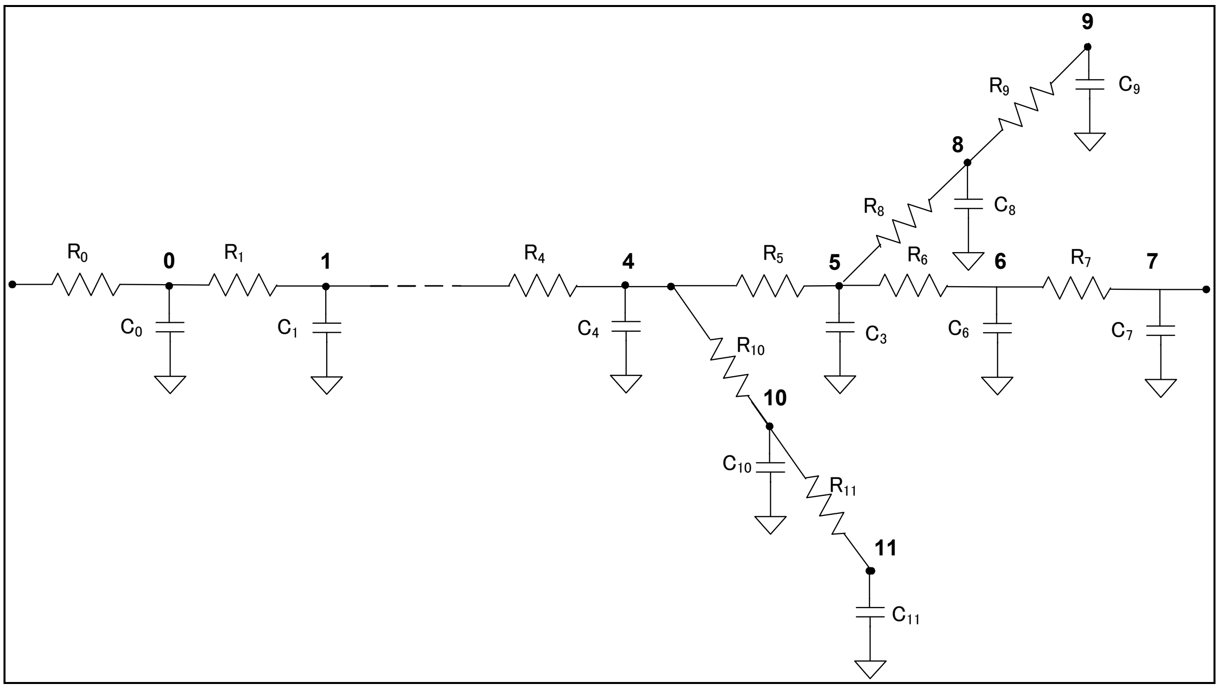


Figure 23. RC model of test path in [Tahoori 02]

Using analysis similar to that in section 3.1 the following detection accuracy is found for the test path in previous test method (Table 4). The data in the table show that the detection accuracy of the test paths used in previous method is lower than detection accuracy of test paths in the proposed method (refer to Table 3).

Table 4. Detection accuracy per Ω resistance (A_0) of the test path in [Tahoori 02]

Test path type	R_{DEF} at R_0	$(A_0- A)/A_0$ %	R_{DEF} at R_5	$(A_0- A)/A_0$ %	R_{DEF} at R_7	$(A_0- A)/A_0$ %
1 branch	0,3000	33,3	0,7200	58,3	3,600	83,3
2 branches	0,4000	60,0	0,9600	58,3	4,800	83,3
3 branches	0,4833	70,4	1,1600	56,8	5,800	82,8

Table 5 shows the improvement of the detection accuracy in the proposed test.

Table 5. Improvement of detection accuracy in the proposed test method $(A_0- A)/A_0$

Test path type	Defect location 1	Defect location 2	Defect location 3
1 branch	33,3 %	58,3 %	83,3 %
2 branches	60,0 %	58,3 %	83,3 %
3 branches	70,4 %	56,8 %	82,8 %

However the number of test configurations has grown. In the previous test method, 8 test configurations were required to test the whole FPGA for all cases of branched test paths. In the proposed test method 8 test configurations for the 1-branch test path, 32 test configurations for the 2-branch test path and 32 test configurations for the 3-branch test path.

6 Discussion

The focus of this master thesis was on the ability of short branched test paths to detect small defects in different parts of FPGA routing network.

From previous research in the field it is known that short test paths with extra branches have superior detection accuracy of faults in the stem of the test path [Peng 04] [Tahoori 02]. In this work detection accuracy of faults in the branches was examined and found to be not as good as the detection accuracy in the stem. Moreover it was observed that the detection accuracy decreased when the location of the defect moved away from the fan-out point or when more branches were added.

The best detection accuracy for faults in the branches was found for the traditional test path with no fan-outs.

Based on the measurement results a new testing approach was suggested, where each segment was tested with the best detection accuracy that could be achieved in that segment. We found that for the presented FPGA architecture all segments outside switch matrices could be tested by test path stems and proposed a test that guarantees that. Segments inside switch matrices could be tested only by test path branches. Using our result that the detection accuracy in a branch is the best close to the fan-out point, we suggested testing each segment inside the switch matrix in both directions. Although this requires more test configurations, the detection accuracy increases as compared to one-directional test.

7 Conclusion

In this thesis a new test method was developed for delay faults in the routing network of an FPGA that targets small resistive open defects. This approach is based on use of short branched test paths and bidirectional application of test patterns. With the proposed test 8 % of all segments in the routing network can be tested at 50 % better detection accuracy compared to the test by short test paths with one-directional application of test patterns. At the cost of four times more test configurations, detection accuracy of branch-adding technique can be increased by more than 50 % for the 2-branch and 3-branch test paths.

References

[Altera 07] Altera Inc., http://www.altera.com, 2007

[Chmelar 03] E. Chmelar, FPGA Interconnect Delay Fault Testing, Proc. Int. Test Conf. (ITC), pp. 1239-1247, 2003

[Gao 05] H. Gao, Y. Yang, X. Ma, G. Dong, Testing for Resistive Shorts in FPGA Interconnects, Proc. of the Sixth Int. Symp. on Quality Electronic Design (ISQED), pp. 159-163, 2005

[Heragu 96] Keerthi Heragu, Janak H.Patel, Vishwani D. Agrawal, Segment Delay Faults: A new Fault Model, Proc. of the 14th IEEE VLSI Test Symposium (VTS '96), pp.32-39

[ITRS 06] International Technology Roadmap for Semiconductors, 2006 Update, http://www.itrs.net/reports.html, 2007

[Krstic 98] Krstic Angela, Cheng Kwang-Ting (Tim), Delay Fault Testing, Kluwer Academic Publishers, Boston 1998, pp.44-50

[Kruseman 04] B. Kruseman, A. K. Majhi, G. Gronthoud, S. Eichenberger, On Hazard-free Pattern for Fine-delay Fault Testing, Proc. Int. Test Conf. (ITC), pp. 213-222, 2004

[Li 01] J. C.-M. Li, C.-W. Tseng, E. J. McCluskey, Testing for Resistive Opens and Stuck Opens, Proc. Int. Test Conf. (ITC), pp. 1049-1058, 2001

[Needham 98] W. Needham, C. Prunty, E. H. Yeoh, High Volume Microprocessor Test Escapes, An Analysis of Defects Our Test are Missing, Proc. Int. Test Conf. (ITC), pp. 25-34, 1998

[Peng 04] Y.-L. Peng, J.-J. Liou, C.-T. Huang, C.-W. Wu, An Application-Independent Delay Testing Methodology for Island-Style FPGA, Proc. of 19th IEEE Int. Symp. on Defect and Fault Tolerance in VLSI Systems (DFT), pp. 478-486, 2004

[PTM 07] Predictive Technology Model, http://www.eas.asu.edu/~ptm/, 2007

[Rabaey 96] J. M. Rabaey, Digital Integrated Circuits: A Design Perspective, Prentice-Hall Inc., Upper Saddle River, 1996

[Renovell 98] M. Renovell, J. M. Portal, J. Figueras, Y. Zorian, Testing the Interconnect of RAM-Based FPGAs, IEEE Design & Test of Computers, pp. 45-50, January-March 1998

[Renovell 02] M. Renovell, J.M. Gallière, F. Azaïs and Y. Bertrand, Modeling Gate Oxide Short Defects in CMOS Minimum Transistors, Proc. of the Seventh IEEE European Test Workshop (ETW), 2002

[Tahoori 02] M. B. Tahoori, Testing for Resistive Open Defects in FPGAs, Proc. Field Programmable Technology Conf. (FPT), pp. 332-335, 2002

[Tahoori 03] M. B. Tahoori, S. Mitra, Automatic Configuration Generation for FPGA Interconnect Testing, Proc. of the 21st IEEE VLSI Test Symp. (VTS), pp. 134-139, 2003

[Tahoori 04] M. B. Tahoori, S. Mitra, Interconnect Delay Testing of Designs on Programmable Logic Devices, Proc. Int. Test Conf. (ITC), pp. 635-644, 2004

[Xilinx 07] Xilinx Inc., http://www.xilinx.com, 2007

[Zhao 06] W. Zhao, Y. Cao, New generation of Predictive Technology Model for sub-45nm design exploration, Proc. of the Sixth Int. Symp. on Quality Electronic Design (ISQED), pp. 585-590, 2006.

Appendix A

A1. 22 nm PTM transistor model:

```
* Beta Version


* PTM 22nm NMOS

.model  nmos  nmos  level = 54

+version = 4.0          binunit = 1            paramchk= 1            mobmod  = 0
+capmod  = 2            igcmod  = 1            igbmod  = 1            geomod  = 1
+diomod  = 1            rdsmod  = 0            rbodymod= 1            rgatemod= 1
+permod  = 1            acnqsmod= 0            trnqsmod= 0

+tnom    = 27           toxe    = 1.2e-9       toxp    = 0.9e-9       toxm    = 1.2e-9
+dtox    = 0.3e-9       epsrox  = 3.9          wint    = 5e-009       lint    = 2.0e-009
+ll      = 0            wl      = 0            lln     = 1            wln     = 1
+lw      = 0            ww      = 0            lwn     = 1            wwn     = 1
+lwl     = 0            wwl     = 0            xpart   = 0            toxref  = 1.2e-9
+xl      = -9e-9
+vth0    = 0.5118       k1      = 0.4          k2      = 0.0          k3      = 0
+k3b     = 0            w0      = 2.5e-006     dvt0    = 1            dvt1    = 2
+dvt2    = 0            dvt0w   = 0            dvt1w   = 0            dvt2w   = 0
+dsub    = 0.078        minv    = 0.05         voffl   = 0            dvtp0   = 1.0e-011
+dvtp1   = 0.1          lpe0    = 0            lpeb    = 0            xj      = 7.2e-009
+ngate   = 2e+020       ndep    = 12e+018      nsd     = 2e+020       phin    = 0
+cdsc    = 0.000        cdscb   = 0            cdscd   = 0            cit     = 0
+voff    = -0.13        nfactor = 2.3          eta0    = 0.0045        etab    = 0
+vfb     = -1.058       u0      = 0.0181       ua      = -5e-010      ub      = 1.7e-018
+uc      = 0            vsat    = 200000       a0      = 1.0          ags     = 0
+a1      = 0            a2      = 1.0          b0      = 0            b1      = 0
+keta    = 0.04         dwg     = 0            dwb     = 0            pclm    = 0.06
+pdiblc1 = 0.001        pdiblc2 = 0.001        pdiblcb = -0.005       drout   = 0.5
+pvag    = 1e-020       delta   = 0.01         pscbe1  = 8.14e+008    pscbe2  = 1e-007
+fprout  = 0.2          pdits   = 0.01         pditsd  = 0.23         pditsl  = 2.3e+006
+rsh     = 5            rdsw    = 130          rsw     = 75           rdw     = 75
+rdswmin = 0            rdwmin  = 0            rswmin  = 0            prwg    = 0
+prwb    = 0            wr      = 1            alpha0  = 0.074        alpha1  = 0.005
+beta0   = 30           agidl   = 0.0002       bgidl   = 2.1e+009     cgidl   = 0.0002
+egidl   = 0.8

+aigbacc = 0.012        bigbacc = 0.0028       cigbacc = 0.002
+nigbacc = 1            aigbinv = 0.014        bigbinv = 0.004        cigbinv = 0.004
+eigbinv = 1.1          nigbinv = 3            aigc    = 0.012        bigc    = 0.0028
+cigc    = 0.002        aigsd   = 0.012        bigsd   = 0.0028       cigsd   = 0.002
+nigc    = 1            poxedge = 1            pigcd   = 1            ntox    = 1

+xrcrg1  = 12           xrcrg2  = 5
+cgso    = 0.65e-010    cgdo    = 0.65e-010    cgbo    = 2.56e-011    cgdl    = 2.653e-10
+cgsl    = 2.653e-10    ckappas = 0.03         ckappad = 0.03         acde    = 1
+moin    = 15           noff    = 0.9          voffcv  = 0.02

+kt1     = -0.11        kt1l    = 0            kt2     = 0.022        ute     = -1.5
+ua1     = 4.31e-009    ub1     = 7.61e-018    uc1     = -5.6e-011    prt     = 0
+at      = 33000

+fnoimod = 1            tnoimod = 0

+jss     = 0.0001       jsws    = 1e-011       jswgs   = 1e-010       njs     = 1
+ijthsfwd= 0.01         ijthsrev= 0.001        bvs     = 10           xjbvs   = 1
+jsd     = 0.0001       jswd    = 1e-011       jswgd   = 1e-010       njd     = 1
+ijthdfwd= 0.01         ijthdrev= 0.001        bvd     = 10           xjbvd   = 1
+pbs     = 1            cjs     = 0.0005       mjs     = 0.5          pbsws   = 1
+cjsws   = 5e-010       mjsws   = 0.33         pbswgs  = 1            cjswgs  = 3e-010
+mjswgs  = 0.33         pbd     = 1            cjd     = 0.0005       mjd     = 0.5
+pbswd   = 1            cjswd   = 5e-010       mjswd   = 0.33         pbswgd  = 1
+cjswgd  = 5e-010       mjswgd  = 0.33         tpb     = 0.005        tcj     = 0.001
+tpbsw   = 0.005        tcjsw   = 0.001        tpbswg  = 0.005        tcjswg  = 0.001
+xtis    = 3            xtid    = 3

+dmcg    = 0e-006       dmci    = 0e-006       dmdg    = 0e-006       dmcgt   = 0e-007
+dwj     = 0.0e-008     xgw     = 0e-007       xgl     = 0e-008
```

```
+rshg    = 0.4          gbmin    = 1e-010       rbpb     = 5            rbpd     = 15
+rbps    = 15           rbdb     = 15           rbsb     = 15           ngcon    = 1

* PTM 22nm PMOS

.model  pmos  pmos  level = 54

+version = 4.0          binunit = 1            paramchk= 1            mobmod   = 0
+capmod  = 2            igcmod   = 1           igbmod   = 1           geomod   = 1
+diomod  = 1            rdsmod   = 0           rbodymod= 1            rgatemod= 1
+permod  = 1            acnqsmod= 0            trnqsmod= 0

+tnom    = 27           toxe     = 1.2e-009     toxp     = 0.9e-009     toxm     = 1.2e-009
+dtox    = 0.3e-9       epsrox   = 3.9          wint     = 5e-009       lint     = 2.0e-009
+ll      = 0            wl       = 0            lln      = 1            wln      = 1
+lw      = 0            ww       = 0            lwn      = 1            wwn      = 1
+lwl     = 0            wwl      = 0            xpart    = 0            toxref   = 1.2e-009
+xl      = -9e-9
+vth0    = -0.372       k1       = 0.4          k2       = -0.01        k3       = 0
+k3b     = 0            w0       = 2.5e-006     dvt0     = 1            dvt1     = 2
+dvt2    = -0.032       dvt0w    = 0            dvt1w    = 0            dvt2w    = 0
+dsub    = 0.1          minv     = 0.05         voffl    = 0            dvtp0    = 1e-011
+dvtp1   = 0.05         lpe0     = 0            lpeb     = 0            xj       = 7.2e-009
+ngate   = 2e+020       ndep     = 4.40e+018    nsd      = 2e+020       phin     = 0
+cdsc    = 0.000        cdscb    = 0            cdscd    = 0            cit      = 0
+voff    = -0.13        nfactor  = 2.3          eta0     = 0.0037       etab     = 0
+vfb     = -1.058       u0       = 0.00230      ua       = -0.5e-009    ub       = 1.6e-018
+uc      = 0            vsat     = 78000        a0       = 1.0          ags      = 1e-020
+a1      = 0            a2       = 1            b0       = 0            b1       = 0
+keta    = -0.047       dwg      = 0            dwb      = 0            pclm     = 0.1
+pdiblc1 = 0.001        pdiblc2 = 0.001         pdiblcb = 3.4e-008      drout    = 0.6
+pvag    = 1e-020       delta    = 0.01         pscbe1   = 8.14e+008    pscbe2   = 9.58e-007
+fprout  = 0.2          pdits    = 0.08         pditsd   = 0.23         pditsl   = 2.3e+006
+rsh     = 5            rdsw     = 130          rsw      = 65           rdw      = 65
+rdswmin = 0            rdwmin   = 0            rswmin   = 0            prwg     = 0
+prwb    = 0            wr       = 1            alpha0   = 0.074        alpha1   = 0.005
+beta0   = 30           agidl    = 0.0002       bgidl    = 2.1e+009     cgidl    = 0.0002
+egidl   = 0.8

+aigbacc = 0.012        bigbacc = 0.0028        cigbacc = 0.002
+nigbacc = 1            aigbinv = 0.014         bigbinv = 0.004         cigbinv = 0.004
+eigbinv = 1.1          nigbinv = 3             aigc     = 0.69         bigc     = 0.0012
+cigc    = 0.0008       aigsd    = 0.0087       bigsd    = 0.0012       cigsd    = 0.0008
+nigc    = 1            poxedge = 1             pigcd    = 1            ntox     = 1

+xrcrg1  = 12           xrcrg2   = 5
+cgso    = 0.65e-010    cgdo     = 0.65e-010    cgbo     = 2.56e-011    cgdl     = 2.653e-10
+cgsl    = 2.653e-10    ckappas = 0.03          ckappad = 0.03          acde     = 1
+moin    = 15           noff     = 0.9          voffcv   = 0.02

+kt1     = -0.11        kt1l     = 0            kt2      = 0.022        ute      = -1.5
+ua1     = 4.31e-009    ub1      = 7.61e-018    uc1      = -5.6e-011    prt      = 0
+at      = 33000

+fnoimod = 1            tnoimod = 0

+jss     = 0.0001       jsws     = 1e-011       jswgs    = 1e-010       njs      = 1
+ijthsfwd= 0.01         ijthsrev= 0.001         bvs      = 10           xjbvs    = 1
+jsd     = 0.0001       jswd     = 1e-011       jswgd    = 1e-010       njd      = 1
+ijthdfwd= 0.01         ijthdrev= 0.001         bvd      = 10           xjbvd    = 1
+pbs     = 1            cjs      = 0.0005       mjs      = 0.5          pbsws    = 1
+cjsws   = 5e-010       mjsws    = 0.33         pbswgs   = 1            cjswgs   = 3e-010
+mjswgs  = 0.33         pbd      = 1            cjd      = 0.0005       mjd      = 0.5
+pbswd   = 1            cjswd    = 5e-010       mjswd    = 0.33         pbswgd   = 1
+cjswgd  = 5e-010       mjswgd   = 0.33         tpb      = 0.005        tcj      = 0.001
+tpbsw   = 0.005        tcjsw    = 0.001        tpbswg   = 0.005        tcjswg   = 0.001
+xtis    = 3            xtid     = 3

+dmcg    = 0e-006       dmci     = 0e-006       dmdg     = 0e-006       dmcgt    = 0e-007
+dwj     = 0.0e-008     xgw      = 0e-007       xgl      = 0e-008

+rshg    = 0.4          gbmin    = 1e-010       rbpb     = 5            rbpd     = 15
+rbps    = 15           rbdb     = 15           rbsb     = 15           ngcon    = 1
```

A2. SPICE script

FPGA interconnect model 0. Simple test path with no extra branches.

```
* FPGA interconnect model 0

* Enable post-processing by AvanWaves
.OPTIONS POST DCON=1

* Transistor dimensions:
.PARAM tran_length='22n'
.PARAM tran_width='130n'

* Transistor model file
.include ptm_mos_22nm_lev54.sp

* Supply voltage:
.PARAM supply_vol='0.8'
***************************************************************************
* Constant voltage source: Vname node1 node2 volatge
VDD VDD GND supply_vol

* Input waveform generator:
* Pulsed voltage source: Vname node1 node2 PULSE param
* param = vlow vhigh delay rise fall pulse_width period
VIN IN0 GND PULSE 0 supply_vol 0.5n 0.01n 0.01n 9.98n 20n
* Inverter:
* Mosfet: Mname drain gate source substrate model length width
MIN1 IN IN0 VDD VDD pmos L=tran_length W=tran_width
MIN2 IN IN0 GND GND nmos L=tran_length W=tran_width

***************************************************************************
* Transient analysis: step size = 50 psec, duration = 32 nsec
.TRAN 50P 32N

***************************************************************************
* Netlist declaration:
***************************************************************************
* Main branch:

* LB1 output buffer
* Mosfet: Mname drain gate source substrate model length width
M1 1 IN VDD VDD pmos L=tran_length W=tran_width
M2 1 IN GND GND nmos L=tran_length W=tran_width
M3 2 1 VDD VDD pmos L=tran_length W=tran_width
M4 2 1 GND GND nmos L=tran_length W=tran_width

* Pass transistor
* Mosfet: Mname drain gate source substrate model length width
M5 3 VDD 4 GND nmos L=tran_length W=tran_width

* LB2 input buffer
* Mosfet: Mname drain gate source substrate model length width
M6 6 5 VDD VDD pmos L=tran_length W=tran_width
M7 6 5 GND GND nmos L=tran_length W=tran_width
M8 OUT 6 VDD VDD pmos L=tran_length W=tran_width
M9 OUT 6 GND GND nmos L=tran_length W=tran_width

* Resistive open defect at the input of pass transistor:
* Resistor: Rname node1 node2 value
```

```
*R 2 3 16k
.connect 2 3

* Resistive open defect at the output of pass transistor:
* Resistor: Rname node1 node2 value
*R 4 5 4k
.connect 4 5

* Capacitance of wires:
* Capacitor: Cname node1 node2 value
C3 3 GND 0.5f
C5 5 GND 0.5f

****************************************************************************
*Delay measurement:

** Minimum and maximum values:
.MEASURE TRAN inmax AVG V(IN) FROM=12n TO=20n
.MEASURE TRAN inmin AVG V(IN) FROM=22n TO=30n
.MEASURE TRAN outmax AVG V(OUT) FROM=12n TO=20n
.MEASURE TRAN outmin AVG V(OUT) FROM=22n TO=30n

** Low-to high delay:
.MEASURE TRAN Td_low_high TRIG V(IN) val='(inmin+inmax)/2' TD=2n RISE=1
+ TARG V(OUT) val='(outmin+outmax)/2' RISE=1

** High-to-low delay:
.MEASURE TRAN Td_high_low TRIG V(IN) val='(inmin+inmax)/2' TD=2n FALL=1
+ TARG V(OUT) val='(outmin+outmax)/2' FALL=1

* Specify nodes to print (hspice will print them all, anyway)
.PRINT TRAN V(IN) V(OUT)
.END
```

FPGA interconnect model 1. Test path with 1 extra branch.

```
* FPGA interconnect model 1

* Enable post-processing by AvanWaves
.OPTIONS POST DCON=1

* Transistor dimensions:
.PARAM tran_length='22n'
.PARAM tran_width='130n'

* Transistor model file
.include ptm_mos_22nm_lev54.sp

* Supply voltage:
.PARAM supply_vol='0.8'
****************************************************************************
* Constant voltage source: Vname node1 node2 volatge
VDD VDD GND supply_vol

* Input waveform generator:
* Pulsed voltage source: Vname node1 node2 PULSE param
* param = vlow vhigh delay rise fall pulse_width period
VIN IN0 GND PULSE 0 supply_vol 0.5n 0.01n 0.01n 9.98n 20n
* Inverter:
* Mosfet: Mname drain gate source substrate model length width
```

```
MIN1 IN IN0 VDD VDD pmos L=tran_length W=tran_width
MIN2 IN IN0 GND GND nmos L=tran_length W=tran_width

***************************************************************************
* Transient analysis: step size = 50 psec, duration = 32 nsec
.TRAN 50P 32N

***************************************************************************
* Netlist declaration:
***************************************************************************
* Main branch:

* LB1 output buffer
* Mosfet: Mname drain gate source substrate model length width
M1 1 IN VDD VDD pmos L=tran_length W=tran_width
M2 1 IN GND GND nmos L=tran_length W=tran_width
M3 2 1 VDD VDD pmos L=tran_length W=tran_width
M4 2 1 GND GND nmos L=tran_length W=tran_width

* Pass transistor
* Mosfet: Mname drain gate source substrate model length width
M5 4 VDD 5 GND nmos L=tran_length W=tran_width

* LB2 input buffer
* Mosfet: Mname drain gate source substrate model length width
M6 7 6 VDD VDD pmos L=tran_length W=tran_width
M7 7 6 GND GND nmos L=tran_length W=tran_width
M8 OUT 7 VDD VDD pmos L=tran_length W=tran_width
M9 OUT 7 GND GND nmos L=tran_length W=tran_width

* Resistive open defect at the output of buffer:
* Resistor: Rname node1 node2 value
*R 2 3 16k
.connect 2 3

* Resistive open defect at the input to pass transistor:
* Resistor: Rname node1 node2 value
*R 3 4 32k
.connect 3 4

* Resistive open defect at the output of pass transistor:
* Resistor: Rname node1 node2 value
*R 5 6 15k
.connect 5 6

* Capacitance of wires:
* Capacitor: Cname node1 node2 value
C4 4 GND 0.5f
C6 6 GND 0.5f

***************************************************************************
* Branch 1:

* Branch 1 Pass transistor
* Mosfet: Mname drain gate source substrate model length width
M10 8 VDD 9 GND nmos L=tran_length W=tran_width

* Branch 1 LB input buffer
* Mosfet: Mname drain gate source substrate model length width
M11 11 10 VDD VDD pmos L=tran_length W=tran_width
M12 11 10 GND GND nmos L=tran_length W=tran_width
```

```
M13 OUT_B1 11 VDD VDD pmos L=tran_length W=tran_width
M14 OUT_B1 11 GND GND nmos L=tran_length W=tran_width

* Resistive open defect at the input to pass transistor:
* Resistor: Rname node1 node2 value
*R 3 8 0.5k
.connect 3 8

* Resistive open defect at the output of pass transistor:
* Resistor: Rname node1 node2 value
*R 9 10 0.5k
.connect 9 10

* Capacitance of wires:
* Capacitor: Cname node1 node2 valu
C8 8 GND 0.5f
C10 10 GND 0.5f

***************************************************************************
*Delay measurement:

** Minimum and maximum values:
.MEASURE TRAN inmax AVG V(IN) FROM=12n TO=20n
.MEASURE TRAN inmin AVG V(IN) FROM=22n TO=30n
.MEASURE TRAN outmax AVG V(OUT) FROM=12n TO=20n
.MEASURE TRAN outmin AVG V(OUT) FROM=22n TO=30n
.MEASURE TRAN outmax_b1 AVG V(OUT_B1) FROM=12n TO=20n
.MEASURE TRAN outmin_b1 AVG V(OUT_B1) FROM=22n TO=30n

** Low-to high delay:
.MEASURE TRAN Td_low_high TRIG V(IN) val='(inmin+inmax)/2' TD=2n RISE=1
+ TARG V(OUT) val='(outmin+outmax)/2' RISE=1
.MEASURE TRAN Td_b1_low_high TRIG V(IN) val='(inmin+inmax)/2' TD=2n RISE=1
+ TARG V(OUT_B1) val='(outmin_b1+outmax_b1)/2' RISE=1

** High-to-low delay:
.MEASURE TRAN Td_high_low TRIG V(IN) val='(inmin+inmax)/2' TD=2n FALL=1
+ TARG V(OUT) val='(outmin+outmax)/2' FALL=1
.MEASURE TRAN Td_b1_high_low TRIG V(IN) val='(inmin+inmax)/2' TD=2n FALL=1
+ TARG V(OUT_B1) val='(outmin_b1+outmax_b1)/2' FALL=1

* Specify nodes to print (hspice will print them all, anyway)
.PRINT TRAN V(IN) V(OUT) V(OUT_B1)
.END
```

FPGA interconnect model 12. Test path with 2 transistors and 1 extra branch.

```
* FPGA interconnect v=12

* Enable post-processing by AvanWaves
.OPTIONS POST DCON=1

* Transistor dimensions:
.PARAM tran_length='22n'
.PARAM tran_width='130n'

* Transistor model file
.include ptm_mos_22nm_lev54.sp
```

```
* Supply voltage:
.PARAM supply_vol='0.8'
***************************************************************************
* Constant voltage source: Vname node1 node2 volatge
VDD VDD GND supply_vol

* Input waveform generator:
* Pulsed voltage source: Vname node1 node2 PULSE param
* param = vlow vhigh delay rise fall pulse_width period
VIN IN0 GND PULSE 0 supply_vol 0.5n 0.01n 0.01n 9.98n 20n
* Inverter:
* Mosfet: Mname drain gate source substrate model length width
MIN1 IN IN0 VDD VDD pmos L=tran_length W=tran_width
MIN2 IN IN0 GND GND nmos L=tran_length W=tran_width

***************************************************************************
* Transient analysis: step size = 50 psec, duration = 32 nsec
.TRAN 50P 32N

***************************************************************************
* Netlist declaration:
***************************************************************************
* Main branch:

* LB1 output buffer
* Mosfet: Mname drain gate source substrate model length width
M1 1 IN VDD VDD pmos L=tran_length W=tran_width
M2 1 IN GND GND nmos L=tran_length W=tran_width
M3 2 1 VDD VDD pmos L=tran_length W=tran_width
M4 2 1 GND GND nmos L=tran_length W=tran_width

* Pass transistor
* Mosfet: Mname drain gate source substrate model length width
M5 4 VDD 5 GND nmos L=tran_length W=tran_width

* LB2 input buffer
* Mosfet: Mname drain gate source substrate model length width
M6 7 6 VDD VDD pmos L=tran_length W=tran_width
M7 7 6 GND GND nmos L=tran_length W=tran_width
M8 OUT 7 VDD VDD pmos L=tran_length W=tran_width
M9 OUT 7 GND GND nmos L=tran_length W=tran_width

* Resistive open defect at the output of buffer:
* Resistor: Rname node1 node2 value
*R 2 3 1k
.connect 2 3

* Resistive open defect at the input to pass transistor:
* Resistor: Rname node1 node2 value
*R 3 4 16k
.connect 3 4

* Resistive open defect at the output of pass transistor:
* Resistor: Rname node1 node2 value
*R 5 6 32k
.connect 5 6

* Capacitance of wires:
* Capacitor: Cname node1 node2 value
C4 4 GND 0.5f
```

```
C6 6 GND 0.5f

***************************************************************************
* Branch 1:

* Branch 1 Pass transistor 1
* Mosfet: Mname drain gate source substrate model length width
M10 8 VDD 9 GND nmos L=tran_length W=tran_width

* Branch 1 Pass transistor 2
* Mosfet: Mname drain gate source substrate model length width
M11 10 VDD 11 GND nmos L=tran_length W=tran_width

* Branch 1 LB input buffer
* Mosfet: Mname drain gate source substrate model length width
M12 13 12 VDD VDD pmos L=tran_length W=tran_width
M13 13 12 GND GND nmos L=tran_length W=tran_width
M14 OUT_B1 13 VDD VDD pmos L=tran_length W=tran_width
M15 OUT_B1 13 GND GND nmos L=tran_length W=tran_width

* Resistive open defect at the input to pass transistor:
* Resistor: Rname node1 node2 value
*R 3 8 15k
.connect 3 8

* Resistive open defect at the output of pass transistor 1:
* Resistor: Rname node1 node2 value
*R 9 10 16k
.connect 9 10

* Resistive open defect at the output of pass transistor 2:
* Resistor: Rname node1 node2 value
*R 11 12 14k
.connect 11 12

* Capacitance of wires:
* Capacitor: Cname node1 node2 valu
C8 8 GND 0.5f
C10 10 GND 0.5f
C12 12 GND 0.5f

***************************************************************************
*Delay measurement:

** Minimum and maximum values:
.MEASURE TRAN inmax AVG V(IN) FROM=16n TO=20n
.MEASURE TRAN inmin AVG V(IN) FROM=22n TO=30n
.MEASURE TRAN outmax AVG V(OUT) FROM=16n TO=20n
.MEASURE TRAN outmin AVG V(OUT) FROM=22n TO=30n
.MEASURE TRAN outmax_b1 AVG V(OUT_B1) AT=20n
.MEASURE TRAN outmin_b1 AVG V(OUT_B1) FROM=22n TO=30n

** Low-to high delay:
.MEASURE TRAN Td_low_high TRIG V(IN) val='(inmin+inmax)/2' TD=2n RISE=1
+ TARG V(OUT) val='(outmin+outmax)/2' RISE=1
.MEASURE TRAN Td_b1_low_high TRIG V(IN) val='(inmin+inmax)/2' TD=2n RISE=1
+ TARG V(OUT_B1) val='(outmin_b1+outmax_b1)/2' RISE=1

** High-to-low delay:
.MEASURE TRAN Td_high_low TRIG V(IN) val='(inmin+inmax)/2' TD=2n FALL=1
+ TARG V(OUT) val='(outmin+outmax)/2' FALL=1
```

```
.MEASURE TRAN Td_b1_high_low TRIG V(IN) val='(inmin+inmax)/2' TD=2n FALL=1
+ TARG V(OUT_B1) val='(outmin_b1+outmax_b1)/2' FALL=1

* Specify nodes to print (hspice will print them all, anyway)
.PRINT TRAN V(IN) V(OUT) V(OUT_B1)
.END
```

FPGA interconnect model 2. Test path with 2 extra branches.

```
* FPGA interconnect v=2

* Enable post-processing by AvanWaves
.OPTIONS POST DCON=1 *NODE

* Transistor dimensions:
.PARAM tran_length='22n'
.PARAM tran_width='130n'

* Transistor model file
.include ptm_mos_22nm_lev54.sp

* Supply voltage:
.PARAM supply_vol='0.8'
***************************************************************************
* Constant voltage source: Vname node1 node2 volatge
VDD VDD GND supply_vol

* Input waveform generator:
* Pulsed voltage source: Vname node1 node2 PULSE param
* param = vlow vhigh delay rise fall pulse_width period
VIN IN0 GND PULSE 0 supply_vol 0.5n 0.01n 0.01n 9.98n 20n
* Inverter:
* Mosfet: Mname drain gate source substrate model length width
MIN1 IN IN0 VDD VDD pmos L=tran_length W=tran_width
MIN2 IN IN0 GND GND nmos L=tran_length W=tran_width

***************************************************************************
* Transient analysis: step size = 50 psec, duration = 32 nsec
.TRAN 50P 32N

***************************************************************************
* Netlist declaration:
***************************************************************************
* Main branch:

* LB1 output buffer
* Mosfet: Mname drain gate source substrate model length width
M1 1 IN VDD VDD pmos L=tran_length W=tran_width
M2 1 IN GND GND nmos L=tran_length W=tran_width
M3 2 1 VDD VDD pmos L=tran_length W=tran_width
M4 2 1 GND GND nmos L=tran_length W=tran_width

* Pass transistor
* Mosfet: Mname drain gate source substrate model length width
M5 4 VDD 5 GND nmos L=tran_length W=tran_width

* LB2 input buffer
* Mosfet: Mname drain gate source substrate model length width
M6 7 6 VDD VDD pmos L=tran_length W=tran_width
M7 7 6 GND GND nmos L=tran_length W=tran_width
```

```
M8 OUT 7 VDD VDD pmos L=tran_length W=tran_width
M9 OUT 7 GND GND nmos L=tran_length W=tran_width

* Resistive open defect at the output of buffer:
* Resistor: Rname node1 node2 value
*R 2 4 16k
*.connect 2 3

* Resistive open defect at the input to pass transistor(NOTICE connection):
* Resistor: Rname node1 node2 value
*R 3 4 16k
*.connect 3 4

* Connection when no defects between LB1 and pass transistors:
.connect 2 4

* Resistive open defect at the output of pass transistor:
* Resistor: Rname node1 node2 value
*R 5 6 16k
.connect 5 6

* Capacitance of wires:
* Capacitor: Cname node1 node2 value
C4 4 GND 0.5f
C6 6 GND 0.5f

***************************************************************************
* Branch 1:

* Branch 1 Pass transistor
* Mosfet: Mname drain gate source substrate model length width
M10 8 VDD 9 GND nmos L=tran_length W=tran_width

* Branch 1 LB input buffer
* Mosfet: Mname drain gate source substrate model length width
M11 11 10 VDD VDD pmos L=tran_length W=tran_width
M12 11 10 GND GND nmos L=tran_length W=tran_width
M13 OUT_B1 11 VDD VDD pmos L=tran_length W=tran_width
M14 OUT_B1 11 GND GND nmos L=tran_length W=tran_width

* Resistive open defect at the input to pass transistor:
* Resistor: Rname node1 node2 value
*R 3 8 0.5k
*.connect 3 8

* Connection when no defects between LB1 and pass transistors:
*.connect 2 8

* Connection when defect is at the LB1 output buffer:
.connect 4 8

* Resistive open defect at the output of pass transistor:
* Resistor: Rname node1 node2 value
*R 9 10 0.5k
.connect 9 10

* Capacitance of wires:
* Capacitor: Cname node1 node2 valu
C8 8 GND 0.5f
C10 10 GND 0.5f
```

```
***************************************************************************
* Branch 2:

* Branch 2 Pass transistor
* Mosfet: Mname drain gate source substrate model length width
M15 12 VDD 13 GND nmos L=tran_length W=tran_width

* Branch 2 LB input buffer
* Mosfet: Mname drain gate source substrate model length width
M16 15 14 VDD VDD pmos L=tran_length W=tran_width
M17 15 14 GND GND nmos L=tran_length W=tran_width
M18 OUT_B2 15 VDD VDD pmos L=tran_length W=tran_width
M19 OUT_B2 15 GND GND nmos L=tran_length W=tran_width

* Resistive open defect at the input to pass transistor:
* Resistor: Rname node1 node2 value
*R 3 12 0.5k
*.connect 3 12

* Connection when no defects:
*.connect 2 12

* Connection when defect is at the LB1 output buffer:
.connect 4 12

* Resistive open defect at the output of pass transistor:
* Resistor: Rname node1 node2 value
*R 13 14 0.5k
.connect 13 14

* Capacitance of wires:
* Capacitor: Cname node1 node2 value
C12 12 GND 0.5f
C14 14 GND 0.5f

***************************************************************************
*Delay measurement:

** Minimum and maximum values:
.MEASURE TRAN inmax AVG V(IN) FROM=16n TO=20n
.MEASURE TRAN inmin AVG V(IN) FROM=22n TO=30n
.MEASURE TRAN outmax AVG V(OUT) FROM=16n TO=20n
.MEASURE TRAN outmin AVG V(OUT) FROM=22n TO=30n
.MEASURE TRAN outmax_b1 AVG V(OUT_B1) FROM=16n TO=20n
.MEASURE TRAN outmin_b1 AVG V(OUT_B1) FROM=22n TO=30n
.MEASURE TRAN outmax_b2 AVG V(OUT_B2) FROM=16n TO=20n
.MEASURE TRAN outmin_b2 AVG V(OUT_B2) FROM=22n TO=30n

** Low-to high delay:
.MEASURE TRAN Td_low_high TRIG V(IN) val='(inmin+inmax)/2' TD=2n RISE=1
+ TARG V(OUT) val='(outmin+outmax)/2' RISE=1
.MEASURE TRAN Td_b1_low_high TRIG V(IN) val='(inmin+inmax)/2' TD=2n RISE=1
+ TARG V(OUT_B1) val='(outmin_b1+outmax_b1)/2' RISE=1
.MEASURE TRAN Td_b2_low_high TRIG V(IN) val='(inmin+inmax)/2' TD=2n RISE=1
+ TARG V(OUT_B2) val='(outmin_b2+outmax_b2)/2' RISe=1

** High-to-low delay:
.MEASURE TRAN Td_high_low TRIG V(IN) val='(inmin+inmax)/2' TD=2n FALL=1
+ TARG V(OUT) val='(outmin+outmax)/2' FALL=1
.MEASURE TRAN Td_b1_high_low TRIG V(IN) val='(inmin+inmax)/2' TD=2n FALL=1
+ TARG V(OUT_B1) val='(outmin_b1+outmax_b1)/2' FALL=1
```

```
.MEASURE TRAN Td_b2_high_low TRIG V(IN) val='(inmin+inmax)/2' TD=2n FALL=1
+ TARG V(OUT_B2) val='(outmin_b2+outmax_b2)/2' FALL=1

* Specify nodes to print (hspice will print them all, anyway)
.PRINT TRAN V(IN) V(OUT) V(OUT_B1) V(OUT_B2)
.END
```

FPGA interconnect model 22. Test path with 2 transistors and 2 extra branches.

```
* FPGA interconnect v=2

* Enable post-processing by AvanWaves
.OPTIONS POST DCON=1 *NODE

* Transistor dimensions:
.PARAM tran_length='22n'
.PARAM tran_width='130n'

* Transistor model file
.include ptm_mos_22nm_lev54.sp

* Supply voltage:
.PARAM supply_vol='0.8'
***************************************************************************
* Constant voltage source: Vname node1 node2 volatge
VDD VDD GND supply_vol

* Input waveform generator:
* Pulsed voltage source: Vname node1 node2 PULSE param
* param = vlow vhigh delay rise fall pulse_width period
VIN IN0 GND PULSE 0 supply_vol 0.5n 0.01n 0.01n 9.98n 20n
* Inverter:
* Mosfet: Mname drain gate source substrate model length width
MIN1 IN IN0 VDD VDD pmos L=tran_length W=tran_width
MIN2 IN IN0 GND GND nmos L=tran_length W=tran_width

***************************************************************************
* Transient analysis: step size = 50 psec, duration = 32 nsec
.TRAN 50P 32N

***************************************************************************
* Netlist declaration:
***************************************************************************
* Main branch:

* L output buffer
* Mosfet: Mname drain gate source substrate model length width
M1 1 IN VDD VDD pmos L=tran_length W=tran_width
M2 1 IN GND GND nmos L=tran_length W=tran_width
M3 2 1 VDD VDD pmos L=tran_length W=tran_width
M4 2 1 GND GND nmos L=tran_length W=tran_width

* Pass transistor
* Mosfet: Mname drain gate source substrate model length width
M5 4 VDD 5 GND nmos L=tran_length W=tran_width

* LB2 input buffer
* Mosfet: Mname drain gate source substrate model length width
M6 7 6 VDD VDD pmos L=tran_length W=tran_width
```

```
M7 7 6 GND GND nmos L=tran_length W=tran_width
M8 OUT 7 VDD VDD pmos L=tran_length W=tran_width
M9 OUT 7 GND GND nmos L=tran_length W=tran_width

* Resistive open defect at the output of buffer:
* Resistor: Rname node1 node2 value
*R 2 4 16k

* Connection when defect is at the input to first pass transistor (main or
branch):
*.connect 2 3

* Resistive open defect at the input to pass transistor(NOTICE connection):
* Resistor: Rname node1 node2 value
*R 3 4 15k

* Connection when defect is at the input to first pass transistor in the
branch:
*.connect 3 4

* Connection when no defects between LB1 and pass transistors:
.connect 2 4

* Resistive open defect at the output of pass transistor:
* Resistor: Rname node1 node2 value
R 5 6 15k
*.connect 5 6

* Capacitance of wires:
* Capacitor: Cname node1 node2 value
C4 4 GND 0.5f
C6 6 GND 0.5f

***************************************************************************
* Branch 1:

* Branch 1 Pass transistor 1
* Mosfet: Mname drain gate source substrate model length width
M20 20 VDD 21 GND nmos L=tran_length W=tran_width

* Branch 1 Pass transistor 2
* Mosfet: Mname drain gate source substrate model length width
M21 22 VDD 23 GND nmos L=tran_length W=tran_width

* Branch 1 LB input buffer
* Mosfet: Mname drain gate source substrate model length width
M22 25 24 VDD VDD pmos L=tran_length W=tran_width
M23 25 24 GND GND nmos L=tran_length W=tran_width
M24 OUT_B1 25 VDD VDD pmos L=tran_length W=tran_width
M25 OUT_B1 25 GND GND nmos L=tran_length W=tran_width

* Resistive open defect at the input to pass transistor 1:
* Resistor: Rname node1 node2 value
*R 3 20 12k

* Connection when defect at the input to main pass transistor:
*.connect 3 20

* Connection when no defects between LB1 and pass transistor:
.connect 2 20
```

```
* Connection when defect at the output of LB1 buffer:
*.connect 4 20

* Resistive open defect at the input to pass transistor 2:
* Resistor: Rname node1 node2 value
*R 21 22 16k
.connect 21 22

* Resistive open defect at the output of pass transistor 2:
* Resistor: Rname node1 node2 value
*R 23 24 15k
.connect 23 24

* Capacitance of wires:
* Capacitor: Cname node1 node2 valu
C20 20 GND 0.5f
C22 22 GND 0.5f
C24 24 GND 0.5f

***************************************************************************
* Branch 2:

* Branch 2 Pass transistor 1
* Mosfet: Mname drain gate source substrate model length width
M30 30 VDD 31 GND nmos L=tran_length W=tran_width

* Branch 2 Pass transistor 2
* Mosfet: Mname drain gate source substrate model length width
M31 32 VDD 33 GND nmos L=tran_length W=tran_width

* Branch 2 LB input buffer
* Mosfet: Mname drain gate source substrate model length width
M32 35 34 VDD VDD pmos L=tran_length W=tran_width
M33 35 34 GND GND nmos L=tran_length W=tran_width
M34 OUT_B2 35 VDD VDD pmos L=tran_length W=tran_width
M35 OUT_B2 35 GND GND nmos L=tran_length W=tran_width

* Resistive open defect at the input to pass transistor 1:
* Resistor: Rname node1 node2 value
*R 3 30 0.5k

* Connection when defect at the input to main pass transistor:
*.connect 3 30

* Connection when no defects:
.connect 2 30

* Connection when defect at the output of LB1 buffer:
*.connect 4 30

* Resistive open defect at the input to pass transistor 2:
* Resistor: Rname node1 node2 value
*R 31 32 0.5k
.connect 31 32

* Resistive open defect at the output of pass transistor 2:
* Resistor: Rname node1 node2 value
*R 33 34 0.5k
.connect 33 34

* Capacitance of wires:
```

```
* Capacitor: Cname node1 node2 valu
C30 30 GND 0.5f
C32 32 GND 0.5f
C34 34 GND 0.5f
****************************************************************************
*Delay measurement:

** Minimum and maximum values:
.MEASURE TRAN inmax AVG V(IN) FROM=16n TO=20n
.MEASURE TRAN inmin AVG V(IN) FROM=22n TO=30n
.MEASURE TRAN outmax AVG V(OUT) FROM=16n TO=20n
.MEASURE TRAN outmin AVG V(OUT) FROM=22n TO=30n
.MEASURE TRAN outmax_b1 AVG V(OUT_B1) FROM=16n TO=20n
.MEASURE TRAN outmin_b1 AVG V(OUT_B1) FROM=22n TO=30n
.MEASURE TRAN outmax_b2 AVG V(OUT_B2) FROM=16n TO=20n
.MEASURE TRAN outmin_b2 AVG V(OUT_B2) FROM=22n TO=30n

** Low-to high delay:
.MEASURE TRAN Td_low_high TRIG V(IN) val='(inmin+inmax)/2' TD=2n RISE=1
+ TARG V(OUT) val='(outmin+outmax)/2' RISE=1
.MEASURE TRAN Td_b1_low_high TRIG V(IN) val='(inmin+inmax)/2' TD=2n RISE=1
+ TARG V(OUT_B1) val='(outmin_b1+outmax_b1)/2' RISE=1
.MEASURE TRAN Td_b2_low_high TRIG V(IN) val='(inmin+inmax)/2' TD=2n RISE=1
+ TARG V(OUT_B2) val='(outmin_b2+outmax_b2)/2' RISe=1

** High-to-low delay:
.MEASURE TRAN Td_high_low TRIG V(IN) val='(inmin+inmax)/2' TD=2n FALL=1
+ TARG V(OUT) val='(outmin+outmax)/2' FALL=1
.MEASURE TRAN Td_b1_high_low TRIG V(IN) val='(inmin+inmax)/2' TD=2n FALL=1
+ TARG V(OUT_B1) val='(outmin_b1+outmax_b1)/2' FALL=1
.MEASURE TRAN Td_b2_high_low TRIG V(IN) val='(inmin+inmax)/2' TD=2n FALL=1
+ TARG V(OUT_B2) val='(outmin_b2+outmax_b2)/2' FALL=1

* Specify nodes to print (hspice will print them all, anyway)
.PRINT TRAN V(IN) V(OUT) V(OUT_B1) V(OUT_B2)
.END
```