

FPGA Interconnect Delay Fault Testing

Erik Chmelar
Center for Reliable Computing
Stanford University
April 7, 2003

Outline

- ***Introduction***
- Previous Work
- Implementation
- Analysis
- Conclusion
- Future Work

Introduction

Motivation

- ~80% of die area is routing resources
 - Wire segments, vias, PIPs, PSMs
- Susceptible to resistive open defects
 - Resistive via or PIP, partially-open metal wire
- Resistive opens can lower reliability
 - Latent defect can 'worsen'
 - Metal migration




Introduction

Objective

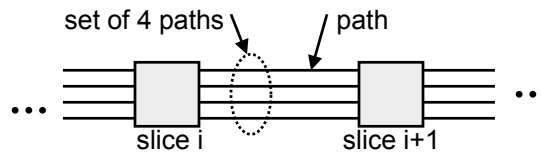
- Devise testing method to detect delay faults
 - Also detect stuck-at 0/1, stuck-open, bridging
 - Few configurations or use partial reconfig.
 - Partial reconfiguration = fast test time
 - Ability to localize fault
 - Diagnosis
 - Application-specific FPGA model [Xilinx 03]

Introduction

Definitions

- Delay faults: rising  falling 
- 

- Slice: smallest functional logic block
- Path: group of elements adjoining two slices
 - Wire segments, vias, PSMs, PIPs
- Set: group of paths simultaneously tested



04/07/03

(c) 2003 Center for Reliable Computing (CRC)

5

Outline

- Introduction
- ***Previous Work***
- Implementation
- Analysis
- Conclusion
- Future Work

04/07/03

(c) 2003 Center for Reliable Computing (CRC)

6

Previous Work

[Tahoori 02]

- Configure chains of paths and logic blocks
 - Add additional load to path
 - Activate one or more additional PIPs
- Shift 1 (0) synchronously through chain
 - Determine where 1 (0) is 'lost'
 - Delay $\propto [R_{tr}(V_{DD}) + R_{defect}] \cdot [C_{segment} + C_{load}]$
 $= [R_{segment} + R_{defect}] \cdot [C_{segment} + C_{load}]$
 - Additional load increases delay of path

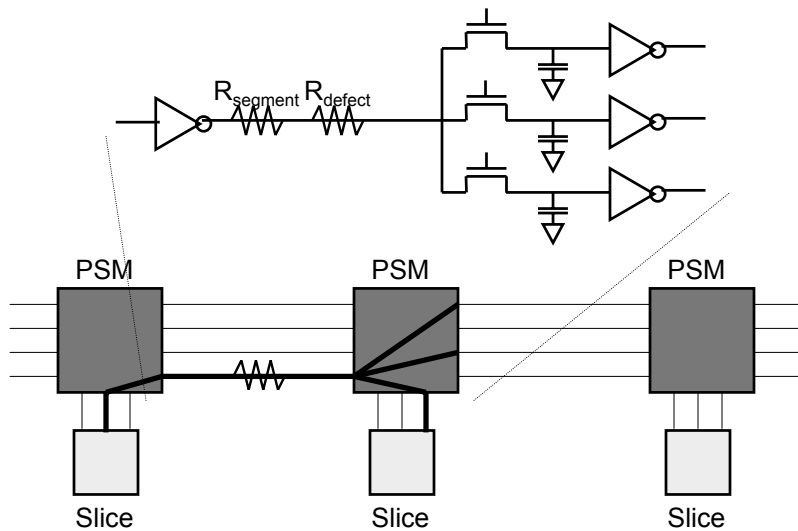
04/07/03

(c) 2003 Center for Reliable Computing (CRC)

7

Previous Work

[Tahoori 02]



04/07/03

(c) 2003 Center for Reliable Computing (CRC)

8

Previous Work

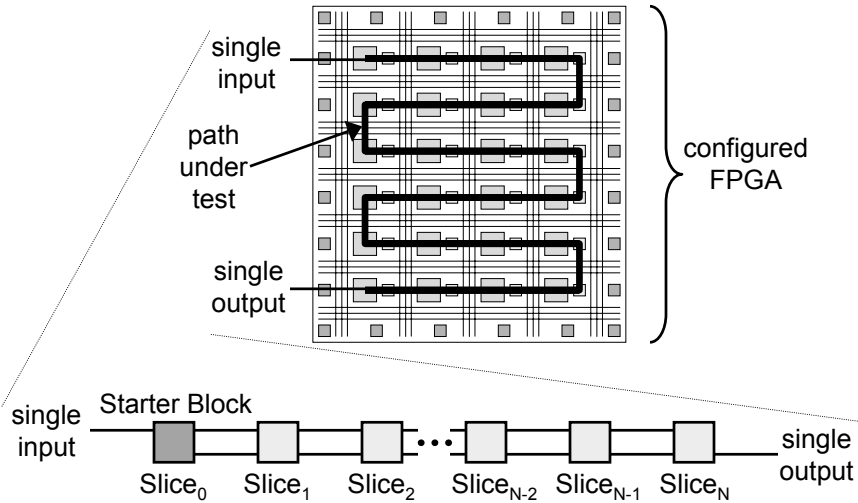
[Abr. & Str. 02]

- Configure chains of paths and logic blocks
 - Identical sequence of LUTs, latches, wire segments, etc. (all transparent, asynch)
- Apply input transition
- Measure phase difference at output by an oscillator loop and counter
 - Determine threshold number of oscillations
 - Less oscillations = pass
 - More oscillations = fail

Outline

- Introduction
- Previous Work
- ***Implementation***
- Analysis
- Conclusion
- Future Work

Implementation Overview



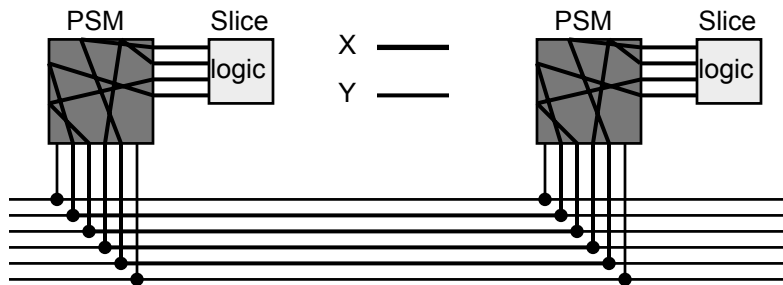
04/07/03

(c) 2003 Center for Reliable Computing (CRC)

11

Implementation Overview

- Configure FPGA into set/slice chains
 - Logic configured in slices
 - Sets partitioned into X and Y paths
 - X and Y paths driven to opposite polarity



04/07/03

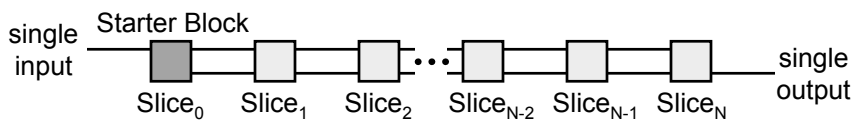
(c) 2003 Center for Reliable Computing (CRC)

12

Implementation

Overview

- 2 slice configs. needed to test both \uparrow and \downarrow
 - Config. A (Test Phase A): X \uparrow , Y \downarrow
 - Config. B (Test Phase B): X \downarrow , Y \uparrow
- Apply single input transition to each chain
 - Test Phase A: \uparrow Test Phase B: \downarrow
- Observe single output transition
 - Test Phase A: \downarrow Test Phase B: \uparrow



04/07/03

(c) 2003 Center for Reliable Computing (CRC)

13

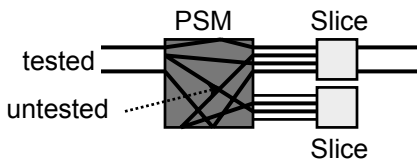
Implementation

Overview

- Methodology bounded by current architectures
 - Constrain logic to smallest functional block
 - Xilinx = *slice*, Alterra = *LE (logic element)*
 - ‘Extra’ routing is needed if logic requires multiple slices to connect logic together

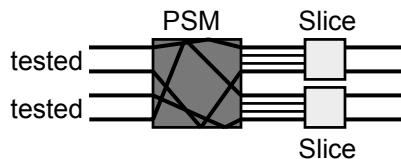
Bad:

- 1 set of routes tested per 2 slices
- many untested routes



Good:

- 1 set of routes tested per slice
- few untested routes (feedback loop)



04/07/03

(c) 2003 Center for Reliable Computing (CRC)

14

Implementation

Overview

- Slice functionality (configured logic)
 - Detects if delay defect present in previous set of paths (slice inputs)
 - Logic creates race condition between paths
 - Signals arrive close in time = pass
 - Up to N-1 of N signals arrive late = fail
 - Controls next set of paths (slice outputs)
 - Pass = prop. *Pass Signal Transition (PST)*¹
 - Fail = prop. *Fail Signal Transition (FST)*²

^{1,2}Defined in example on slides 21, 22

04/07/03

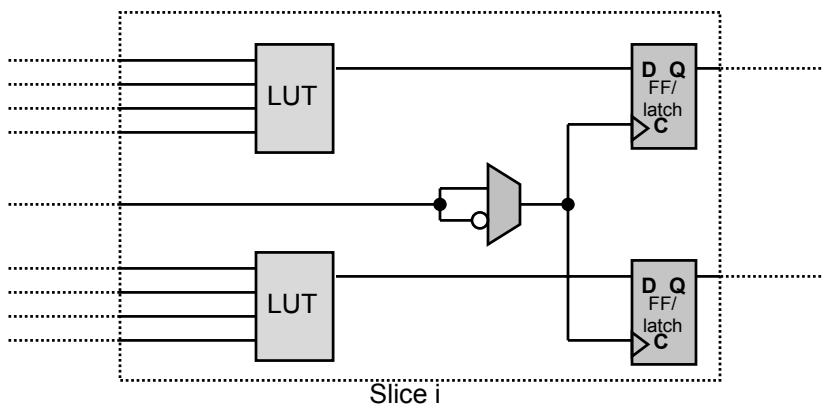
(c) 2003 Center for Reliable Computing (CRC)

15

Implementation

Simplified View of a Slice or LE

- 2 logic elements (LUT/RAM/SR)
- 2 bistable elements (latch/flip flop)



04/07/03

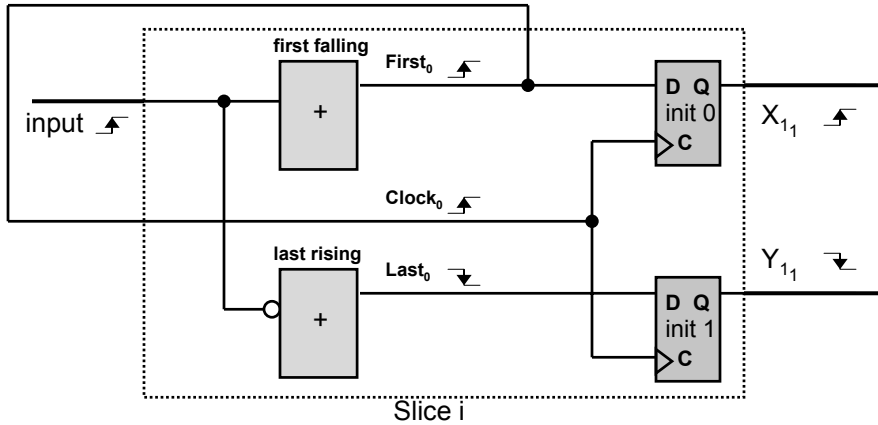
(c) 2003 Center for Reliable Computing (CRC)

16

Implementation

Starter Block (Test Phase A): $X \uparrow$, $Y \downarrow$

- 1 tester input signal to avoid skew problems
 - Test Phase B starter block similar



04/07/03

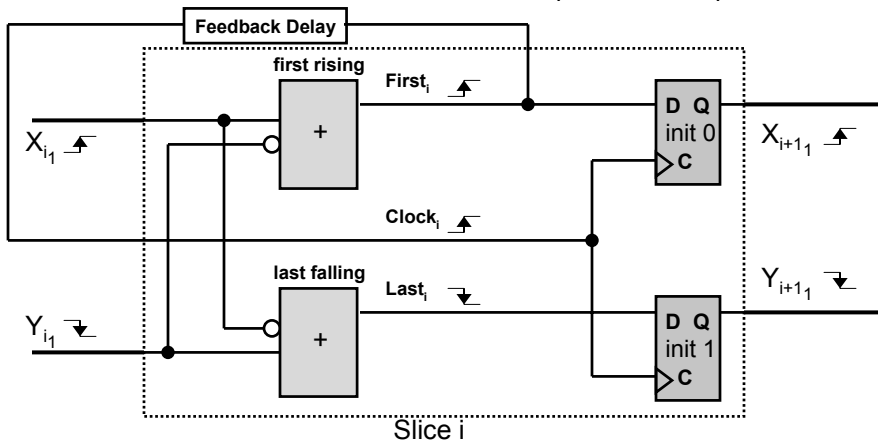
(c) 2003 Center for Reliable Computing (CRC)

17

Implementation

Configuration A (Test Phase A): $X \uparrow$, $Y \downarrow$

- $Clock_i$ is a delayed version of $First_i$
- Race condition between $Clock_i$ and $Last_i$



04/07/03

(c) 2003 Center for Reliable Computing (CRC)

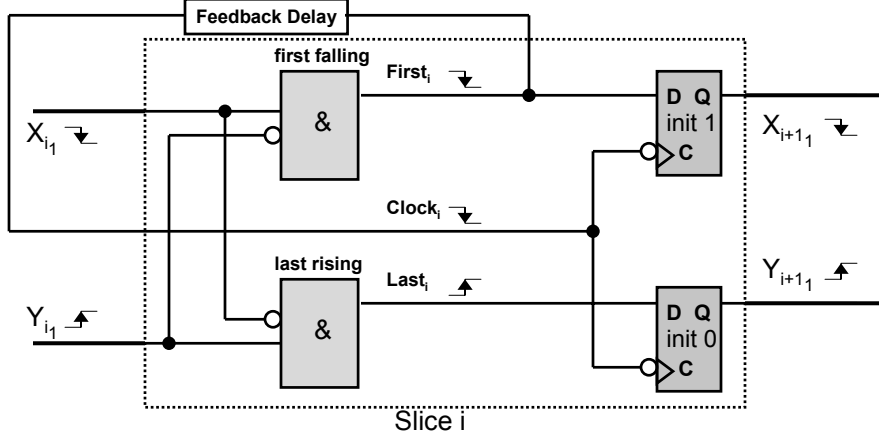
18

Implementation

Configuration B (Test Phase B): $X \downarrow$, $Y \uparrow$

- Partial reconfiguration of Configuration A

– $Clock_i \rightarrow Clock_i'$, $+ \rightarrow \&$, $init : 0 \rightarrow 1$, $1 \rightarrow 0$



04/07/03

(c) 2003 Center for Reliable Computing (CRC)

19

Outline

- Introduction
- Previous Work
- Implementation
- **Analysis**
- Conclusion
- Future Work

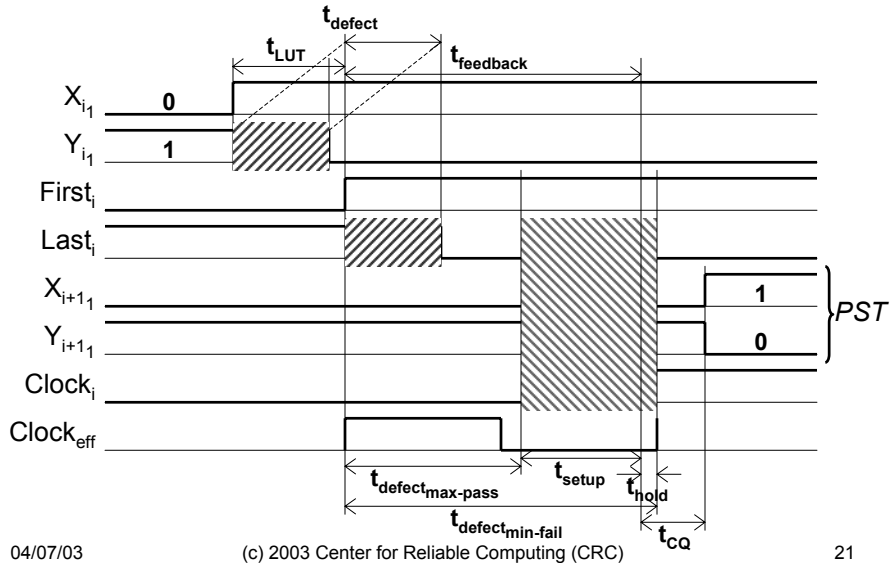
04/07/03

(c) 2003 Center for Reliable Computing (CRC)

20

Analysis

Below-threshold Delay Fault (Cfg. A)



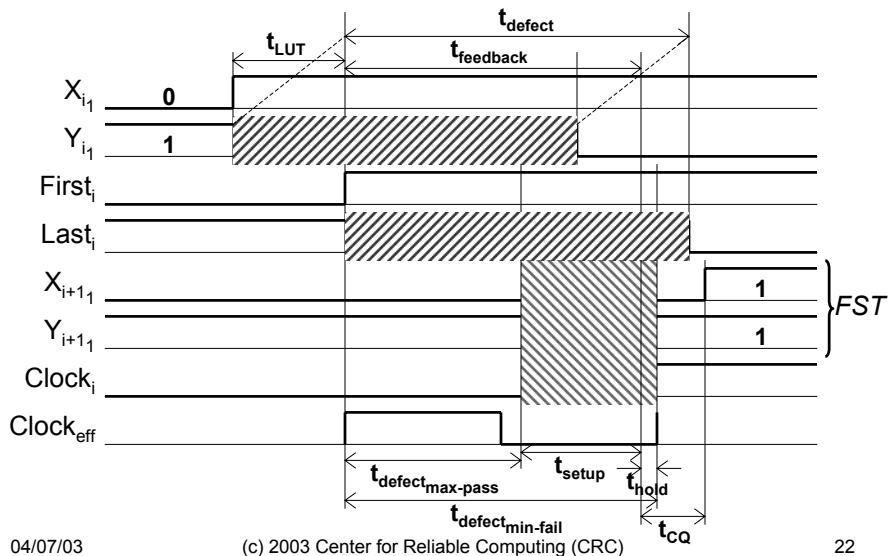
04/07/03

(c) 2003 Center for Reliable Computing (CRC)

21

Analysis

Above-threshold Delay Fault (Cfg. A)



04/07/03

(c) 2003 Center for Reliable Computing (CRC)

22

Analysis

- Timing parameters
 - Average minimum feedback delay
 - $t_{\text{feedback}_{\text{avg-min}}}$
 - Max path delay difference guaranteed to pass
 - $t_{\text{defect}_{\text{max-pass}}} = t_{\text{feedback}} - t_{\text{setup}}$
 - Min path delay difference guaranteed to fail
 - $t_{\text{defect}_{\text{min-fail}}} = t_{\text{feedback}} + t_{\text{hold}}$

Analysis

Maximum Sensitivity Bounds

Device	Speed	$t_{\text{feedback}_{\text{avg-min}}}$ (ps)	$t_{\text{defect}_{\text{max-pass}}}$ (ps)	$t_{\text{defect}_{\text{min-fail}}}$ (ps)
Spartan-II	fast	840	40	840
	slow		40	
Spartan-IIE	fast	360	-340	360
	slow		-440	
Virtex	fast	1020	420	1020
	slow		220	
Virtex-II	fast	970	670	900
	slow		600	
Virtex-IIE	fast	370	-90	370
	slow		-230	
Virtex-IIPro	fast	790	780	880
	slow		780	

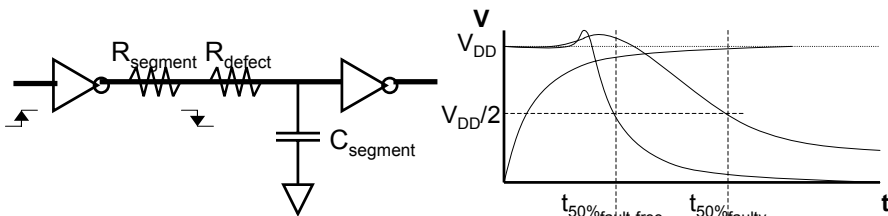
Analysis

Architecture-independent

- Min feedback delay is architecture-dependent
 - 360 ps (Spartan-IIIE) – 1020 ps (Virtex)
 - But dependence doesn't matter
 - Usually too small anyway (sometimes < 0)
 - Too sensitive => false failures
 - Allow margin for path delay mismatches
 - Use longer feedback path => less sensitive
- Slice/LE always has 2 LUTs, 2 FF/latches

Analysis

Resistive Open Defects



$$V(t) = V_{DD} (1 - e^{-t / (R_{segment} + R_{defect}) C_{segment}})$$

$$t_{50\%} = -(R_{segment} + R_{defect}) C_{segment} \ln(.5)$$

$$t_{defect} = t_{50\% \text{ faulty}} - t_{50\% \text{ fault-free}} = -R_{defect} C_{segment} \ln(.5)$$

$$R_{defect_{\max-pass}} = \frac{-t_{defect_{\max-pass}}}{C_{segment} \ln(.5)} ; R_{defect_{\min-fail}} = \frac{-t_{defect_{\min-fail}}}{C_{segment} \ln(.5)}$$

Analysis

Maximum Sensitivity Bounds, Guaranteed Failure

Device	Speed	Clock _{eff} (GHz)	R _{defect_{min-fail}} (kΩ) (C _{segment} = .5 pF)
Spartan-II	fast	1.19	2.4
	slow	1.19	2.4
Spartan-IIE	fast	2.78	1.0
	slow	2.78	1.0
Virtex	fast	0.98	2.9
	slow	0.98	2.9
Virtex-II	fast	1.11	2.6
	slow	1.14	2.5
Virtex-IIE	fast	2.70	1.1
	slow	2.70	1.1
Virtex-IIPro	fast	1.14	2.5
	slow	1.10	2.6

04/07/03

(c) 2003 Center for Reliable Computing (CRC)

27

Analysis

Bridging Faults

- X, Y always driven with opposite polarity
 - Test Phase A: **X=0→1, Y=1→0**
 - Test Phase B: **X=1→0, Y=0→1**
 - Bridge fault (any model, ex. wired AND/OR) causes X, Y lines to have same value
 - Causes *First_i* to transition, *Last_i* does not
 - » *Clock_i* transitions
 - » Slice propagates *FST* down chain

04/07/03

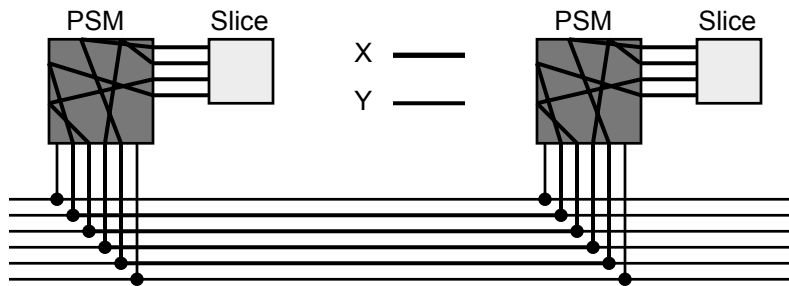
(c) 2003 Center for Reliable Computing (CRC)

28

Analysis

Bridging Fault Bus Line Interleaving

- If set consists of more than 2 paths
 - Interleave X and Y lines on shared common bus: \equiv , not \equiv



04/07/03

(c) 2003 Center for Reliable Computing (CRC)

29

Analysis

Fault Localization

- No search iterations necessary
 - Configure chains of paths and logic blocks
 - Test
 - Incremental test result saved in each slice
 - Flip-flops save *PST* (pass) or *FST* (fail)
 - Scan out (*readback*) flip-flop values
 - XOR readback with expected value
 - First non-zero entry determines failing set
 - Quickly identifies set between 2 slices

04/07/03

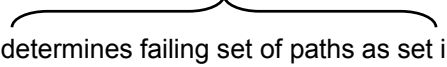
(c) 2003 Center for Reliable Computing (CRC)

30

Analysis

Fault Localization Example: Cfg. A

readback file:	$Q_{X_0}Q_{Y_0}$	$Q_{X_1}Q_{Y_1}$...	$Q_{X_{i-1}}Q_{Y_{i-1}}$	$Q_{X_i}Q_{Y_i}$	$Q_{X_{i+1}}Q_{Y_{i+1}}$...	$Q_{X_N}Q_{Y_N}$
before test:	01	01	...	01	01	01	...	01
test passed:	10	10	...	10	10	10	...	10
test failed: \oplus	10	10	...	10	11	11	...	11
	00	00	...	00	01	01	...	01
	set ₀	set ₁	...	set _{i-1}	set _i	set _{i+1}	...	set _N



determines failing set of paths as set i

Outline

- Introduction
- Previous Work
- Implementation
- Analysis
- **Conclusion**
- Future Work

Conclusion

- FPGA interconnect delay fault testing method
 - Detects multiple small delay defects ($\sim 1\text{-}2\text{ k}\Omega$)
 - Also detects multiple
 - Stuck-at 0/1 faults
 - Stuck open faults
 - Bridging faults
 - Able to quickly localize faulty set of paths
 - Architecture-independent

Outline

- Introduction
- Previous Work
- Implementation
- Analysis
- Conclusion
- ***Future Work***

Future Work

- Try out method on actual devices
 - Determine appropriate feedback delay values
 - Set detection sensitivity
- Will setup time violations cause problems? (no)
 - Defects between $t_{\text{defect}_{\text{max-pass}}}$ and $t_{\text{defect}_{\text{min-fail}}}$

References (1)

- [Abr. & Str. 02] Abramovici, M and Stroud, C. "BIST-Based Delay-Fault Testing in FPGAs," Proc. of the Eighth International On-Line Testing Workshop, 131-134, 2000.
- [Chmelar 03] Chmelar, E. "Path Phase Difference' Delay Fault Testing of Routing Resources in FPGAs with Diagnosis," Reliability and Testability Seminar, Stanford University, January 2003.
- [Tahoori 02] Tahoori, M. B. "Testing for Resistive Opens in FPGA," Center for Reliable Computing Technical Report, 2002.
- [Xilinx 03] Xilinx, Inc., "Xilinx EasyPath Solutions," http://www.xilinx.com/xlnx/xil_prodcat_product.jsp?title=v2_easypath, 2003.