



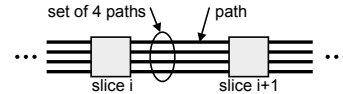


'Path Phase Difference' Delay Fault Testing of Routing Resources in FPGAs with Diagnosis

Erik Chmelar
Center for Reliable Computing
Stanford University
January 27, 2003

Introduction Definitions

- Delay faults: rising , falling 
- Path: group of elements adjoining two slices
– wire segments, vias, PSMs, PIPs
- Set: group of paths simultaneously tested
- Slice: self-contained subdivision of a CLB



01/27/03

(c) 2003 Center for Reliable Computing

4

Outline

- Introduction
- Objective
- Implementation
- Analysis
- Fault Coverage
- Diagnosis
- Conclusion
- Future Work

01/27/03

(c) 2003 Center for Reliable Computing

2

Introduction Basic Idea

- Path delay $\alpha (R^*_{\text{path}} + R^*_{\text{def}}) \times (C^*_{\text{path}} + C^*_{\text{def}})$
- Test for delay defects
 - Excess resistance (resistive open)
 - Excess capacitance (capacitive coupling)

01/27/03

(c) 2003 Center for Reliable Computing

5

Introduction Motivation

- ~80% of die area is routing resources
 - wire segments, PSMs, PIPs, etc.
- VLSI scaling causing more delay defects [Abr. & Str. 02]
- Delay defects can cause reliability issues
 - Resistive opens can become 'worse'
 - metal migration

01/27/03

(c) 2003 Center for Reliable Computing

3

Introduction

Regularity of Routing Resources

- Routing resources are bus oriented
 - Usually 4-8 wires per bus
 - Allows creation of signal paths with nearly identical propagation delays
- Exploit regularity to test for delay faults

01/27/03

(c) 2003 Center for Reliable Computing

6

Introduction

Previous Work [Abr. & Str. 02]

- Configure chains of paths and logic blocks
 - Same sequence of LUTs, latches, wire segments, etc. (all transparent, no clock)
- Apply input transition
- Measure phase difference at output by an oscillator loop and counter
 - Determine some threshold number of oscillations

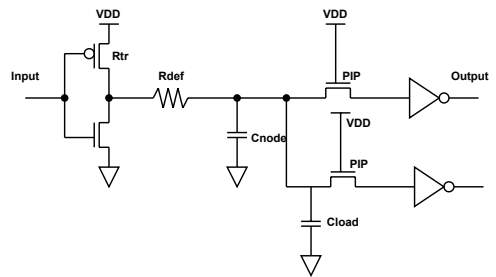
01/27/03

(c) 2003 Center for Reliable Computing

7

Introduction

Previous Work [Tahoori 02]



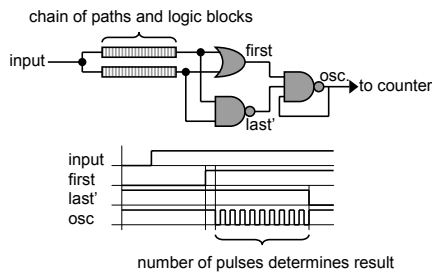
01/27/03

(c) 2003 Center for Reliable Computing

10

Introduction

Previous Work [Abr. & Str. 02]



01/27/03

(c) 2003 Center for Reliable Computing

8

Outline

- Introduction
- **Objective**
- Implementation
- Analysis
- Fault Coverage
- Diagnosis
- Conclusion
- Future Work

01/27/03

(c) 2003 Center for Reliable Computing

11

Introduction

Previous Work [Tahoori 02]

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

01/27/03

(c) 2003 Center for Reliable Computing

9

Objective

Delay Fault Test Criteria

- Detect delay defects in routing resources
 - Few configurations
 - fast test time
 - Diagnosable
 - ability to quickly localize failure location
 - Ability to detect other faults
 - stuck-at 0/1
 - stuck-open
 - bridging (wire AND/OR)

01/27/03

(c) 2003 Center for Reliable Computing

12

Outline

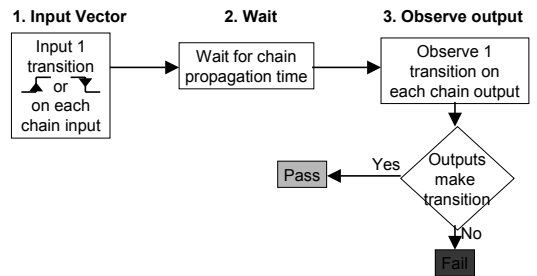
- Introduction
- Objective
- **Implementation**
- Analysis
- Fault Coverage
- Diagnosis
- Conclusion
- Future Work

01/27/03

(c) 2003 Center for Reliable Computing

13

Implementation Test Flow Chart



01/27/03

(c) 2003 Center for Reliable Computing

16

Implementation Overview

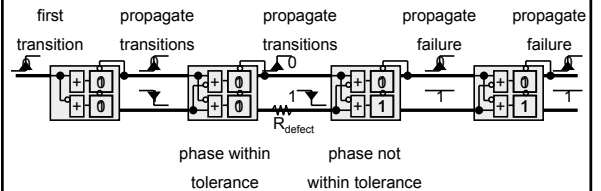
- Configure FPGA into set/slice chains
 - Slice output tests (controls) next set
 - Slice input detects if delay defect present in previous set of paths
 - measures paths' signal phase difference
 - tests next set if previous passed
 - propagates failure if previous failed
- Apply single input transition (no clock)
- Observe single output transition

01/27/03

(c) 2003 Center for Reliable Computing

14

Implementation Chain Propagation Example (Cfg. A)

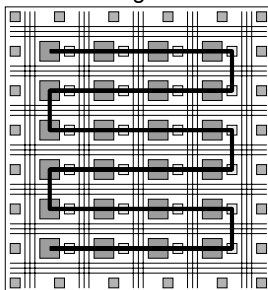


01/27/03

(c) 2003 Center for Reliable Computing

17

Implementation Chain Signal Flow

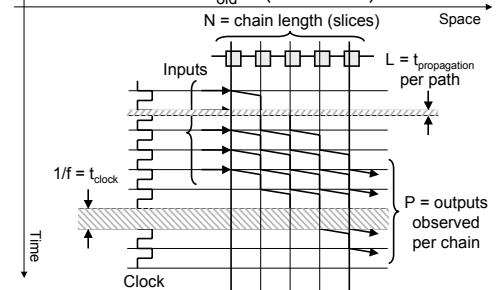


01/27/03

(c) 2003 Center for Reliable Computing

15

Implementation tester time_{old} = (N + P - 1)/f



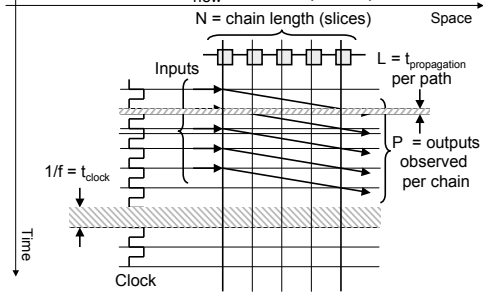
01/27/03

(c) 2003 Center for Reliable Computing

18

Implementation

$$\text{tester time}_{\text{new}} = N * L + (P - 1) / f$$



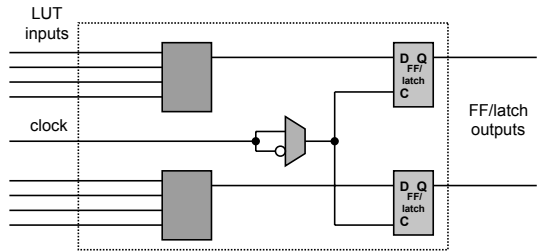
01/27/03

(c) 2003 Center for Reliable Computing

19

Implementation

Simple View of a Slice



01/27/03

(c) 2003 Center for Reliable Computing

22

Implementation

Pattern Application Speedup Example

- speedup = test time_{old} / test time_{new} ≈ 1 / (f * L)
 - f = 60 MHz (ATE clock)
 - P = 1 (observe 1 output cycle)
 - N = 10000 (chain length (slices))
 - L = 2 ns (path length (delay))
 - test time_{old} = (N + P - 1) / f = 167 μs
 - test time_{new} = N * L + (P - 1) / f = 20 μs
 - speedup ≈ 8

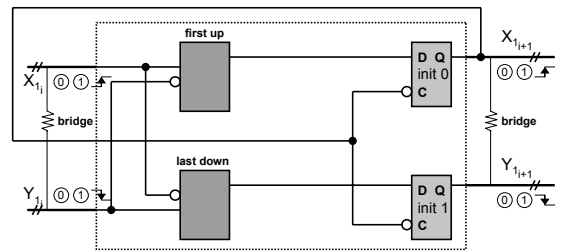
01/27/03

(c) 2003 Center for Reliable Computing

20

Implementation

Configuration A: X \downarrow , Y \uparrow



01/27/03

(c) 2003 Center for Reliable Computing

23

Implementation

Configurations

- Two configurations to test each set
 - One for \downarrow delay, one for \uparrow delay
 - 2 ≤ paths in set ≤ maximum inputs to LUT
 - at least 2 paths, each races the other
 - typically 4 inputs per LUT, 2 LUTs per slice
 - half the set is input to each LUT

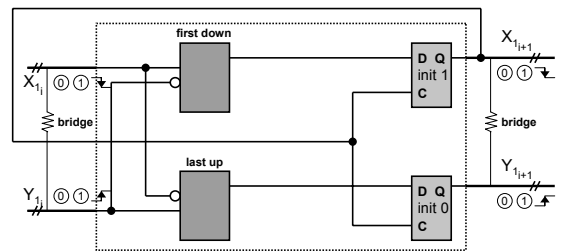
01/27/03

(c) 2003 Center for Reliable Computing

21

Implementation

Configuration B: X \uparrow , Y \downarrow



01/27/03

(c) 2003 Center for Reliable Computing

24

Outline

- Introduction
- Objective
- Implementation
- **Analysis**
- Fault Coverage
- Diagnosis
- Conclusion
- Future Work

01/27/03

(c) 2003 Center for Reliable Computing

25

Analysis

Xilinx Virtex-II Timing Values¹

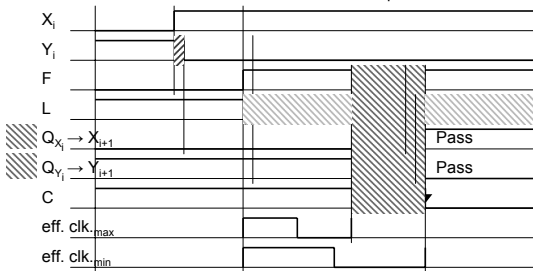
$$t_{\text{fault-max-pass}} = t_{\text{DQ}} + t_{\text{feedback}} - t_{\text{setup}}; \quad t_{\text{fault-min-fail}} = t_{\text{DQ}} + t_{\text{feedback}} + t_{\text{hold}}$$

Symbol	Speed Grade			Units
	-6 (fastest)	-5	-4 (slowest)	
² t _{DQ-latch}	.54	.59	.68	ns, max
t _{setup}	.30	.33	.37	ns, max
t _{feedback}	.1	.1	.1	ns, approx
t _{hold}	-.07	-.08	-.09	ns, min
t _{fault-max-pass}	.34	.36	.41	ns, max
t _{fault-min-fail}	.57	.61	.69	ns, min
eff. clk. _{max}	2.94	2.78	2.44	GHz, max

[Xilinx 01] ¹Other device families/vendors are similar; ²t_{DQ} = t_{co}₂₈
01/27/03 (c) 2003 Center for Reliable Computing

Analysis

$$0 \leq t_{\text{fault}} \leq t_{\text{DQ}} + t_{\text{feedback}} - t_{\text{setup}} \text{ (Cfg. A)}$$



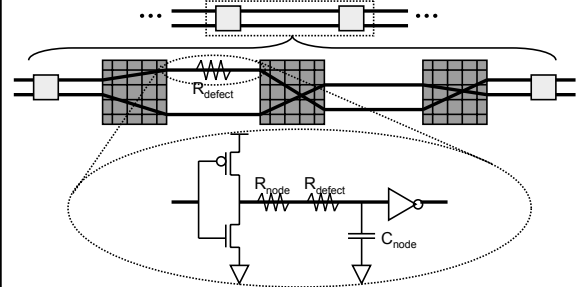
01/27/03

(c) 2003 Center for Reliable Computing

26

Analysis

Intermediate Nodes



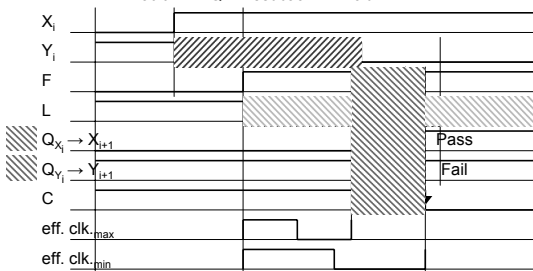
01/27/03

(c) 2003 Center for Reliable Computing

29

Analysis

$$t_{\text{fault}} \geq t_{\text{DQ}} + t_{\text{feedback}} + t_{\text{hold}} \text{ (Cfg. A)}$$



01/27/03

(c) 2003 Center for Reliable Computing

27

Analysis

$$V(t) = V_{DD}(1 - e^{-t/R^*C^*}); \quad \text{let } R^*, C^* = R, C$$

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

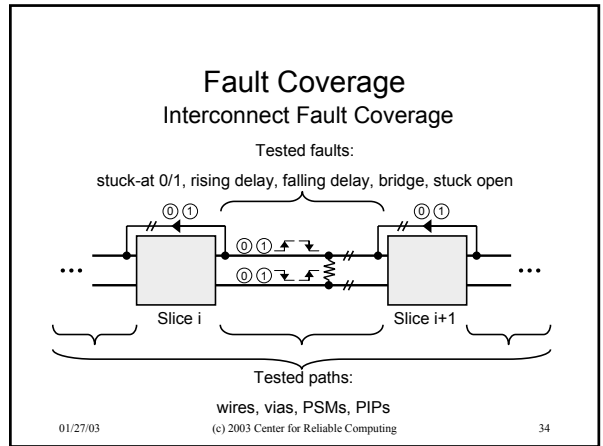
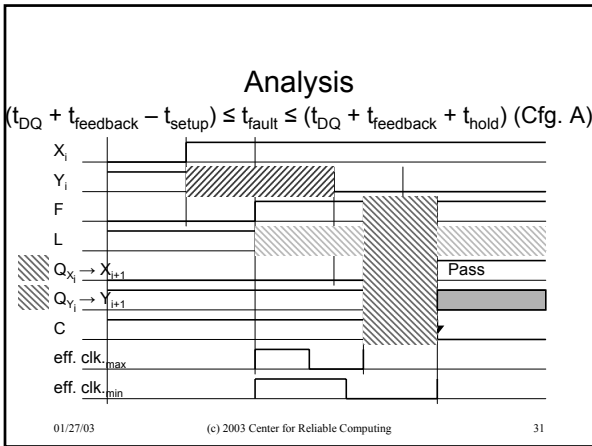
$$t_{\text{fault}} = t_{50\%_{\text{faulty}}} - t_{50\%_{\text{fault-free}}} = -R_{\text{defect}}C_{\text{node}} \ln(.5)$$

$$R_{\text{defect-max-pass}} = \frac{-t_{\text{fault-max-pass}}}{C_{\text{node}} \ln(.5)}; \quad R_{\text{defect-min-fail}} = \frac{-t_{\text{fault-min-fail}}}{C_{\text{node}} \ln(.5)}$$

Symbol	Speed Grade			Units
	-6 (fastest)	-5	-4 (slowest)	
¹ R _{defect-pass-max}	1.0	1.1	1.2	kΩ, max
¹ R _{defect-fail-min}	1.6	1.8	2.0	kΩ, min
² R _{defect-fail-approx}	48			kΩ, approx

¹C_{node} approximated as .5pF; ²Assume f = 60 MHz tester clock
01/27/03 (c) 2003 Center for Reliable Computing

30



Analysis

Chain Propagation

Cfg.	Delay Fault Presence	Test Result	Test Detection	Diagnosis Readback File		
				Q _{X_k} Q _{Y_k}		
				k < i	k = i	k > i
A: init 01	none	pass	n/a	10	10	10
	marginal	pass	no	10	10	10
		fail	yes	10	11	11
gross	fail	yes	10	11	11	
B: init 10	none	pass	n/a	01	01	01
	marginal	pass	no	01	01	01
		fail	yes	01	00	00
	gross	fail	yes	01	00	00

01/27/03 (c) 2003 Center for Reliable Computing 32

- ### Fault Coverage
- #### Interconnect Fault Coverage
- Rising/falling delay fault
 - Stuck-at 0/1 (infinite delay)
 - Never any transition on path
 - Stuck open
 - Probably never any transition on path
 - if coupling to adjacent path, then behaves similar to bridge
 - Bridge
 - Opposite polarity on paths detect bridge
- 01/27/03 (c) 2003 Center for Reliable Computing 35

- ### Outline
- Introduction
 - Objective
 - Implementation
 - Analysis
 - **Fault Coverage**
 - Diagnosis
 - Conclusion
 - Future Work
- 01/27/03 (c) 2003 Center for Reliable Computing 33

Fault Coverage

Bridging Faults

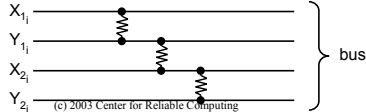
Cfg.	Bridging Fault	X _i (k) Y _i (k)	X _i (k+1) Y _i (k+1)	X _{i+1} (k+1) Y _{i+1} (k+1)	X _{i+1} (k+2) Y _{i+1} (k+2)	Test Result
A:	none	01	10	01	10	pass
	wire AND	0·1 = 00	1·0 = 00	11	11	fail
01	wire OR	0+1 = 11	1+0 = 11	11	11	fail
B:	none	10	01	10	01	pass
	wire AND	1·0 = 00	0·1 = 00	00	00	fail
10	wire OR	1+0 = 11	0+1 = 11	00	00	fail

01/27/03 (c) 2003 Center for Reliable Computing 36

Fault Coverage

Bridging Fault Bus Line Staggering

- If set consists of more than two paths
 - Stagger X and Y lines
 - X, Y always driven with opposite polarity
 - bridge fault (wire AND/OR) causes X, Y lines to always have same value
 - bridge fault will be detected



01/27/03

(c) 2003 Center for Reliable Computing

37

Diagnosis

Single Oscillator Loop [Abr. & Str. 02]

- No reasonable diagnosis method
 - All paths grouped together in long chains
 - no intermediate path test results stored
 - Iterate:
 - configure
 - test
 - count oscillations to determine if faulty
 - re-route chains (eliminate LUTs, wire segments, etc.)

01/27/03

(c) 2003 Center for Reliable Computing

40

Outline

- Introduction
- Objective
- Implementation
- Analysis
- Fault Coverage
- **Diagnosis**
- Conclusion
- Future Work

01/27/03

(c) 2003 Center for Reliable Computing

38

Diagnosis

Path Phase Difference

- No search iterations necessary
 - Incremental test result saved in each slice
 - Configure
 - Test
 - Observe output transition
 - XOR readback with expected value
 - first non-zero entry determines failing set
 - identifies set between 2 slices

01/27/03

(c) 2003 Center for Reliable Computing

41

Diagnosis

Standard Shift Chain Testing

- Shift single 1 (0) through chain of 0's (1's)
 - Determine where 1 (0) is 'lost'
 - Iterate:
 - configure
 - test
 - readback
 - determine if 1 (0) is 'lost'
 - modify test length

01/27/03

(c) 2003 Center for Reliable Computing

39

Diagnosis

Path Phase Difference 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

01/27/03

(c) 2003 Center for Reliable Computing

42

Outline

- Introduction
- Objective
- Implementation
- Analysis
- Fault Coverage
- Diagnosis
- **Conclusion**
- Future Work

01/27/03

(c) 2003 Center for Reliable Computing

43

Future Work

- Try out method on actual devices
 - Are the timing thresholds appropriate?
 - $t_{\text{fault}_{\text{pass-max}}} = t_{\text{DQ}} + t_{\text{feedback}} - t_{\text{setup}}$
 - $t_{\text{fault}_{\text{fail-min}}} = t_{\text{DQ}} + t_{\text{feedback}} + t_{\text{hold}}$
 - don't want threshold too low
 - false failures
 - Will setup time violation cause problems?
 - metastable oscillations

01/27/03

(c) 2003 Center for Reliable Computing

46

Conclusion

- Able to detect small delay defects (~1-2 k Ω)
- Also detect stuck-at 0/1, stuck open, bridge
- Able to quickly localize faulty set of paths
- Test pattern application time reduced

01/27/03

(c) 2003 Center for Reliable Computing

44

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.
- [Tahoori 02] Tahoori, M. B. "Testing for Resistive Opens in FPGA," Center for Reliable Computing Technical Report, 2002.
- [Xilinx 01] Xilinx, Inc., San Jose, Ca. "Virtex-II Platform FPGA Handbook," December 2001.

01/27/03

(c) 2003 Center for Reliable Computing

47

Outline

- Introduction
- Objective
- Implementation
- Analysis
- Fault Coverage
- Diagnosis
- Conclusion
- **Future Work**

01/27/03

(c) 2003 Center for Reliable Computing

45