


Testing Digital Systems I

Lecture 7: Boolean Testing Using Fault Models

Instructor: M. Tahoori


Copyright 2010, M. Tahoori TDS I: Lecture 7 1



This Lecture

- Specific Fault Objective — Target Fault
 - Boolean
 - Algebraic and Boolean Difference
 - Path Tracing
 - D Algorithm, PODEM, Fan


Copyright 2010, M. Tahoori TDS I: Lecture 7 2



Fault Model-based Test Sets

- Good or Fault-Free Circuit
 - Circuit with No Faults Present
- Faulty Circuit
 - Circuit with Fault Present
- Detection vs Diagnosis

Copyright 2010, M. Tahoori TDS I: Lecture 7 3



Specific-Fault Oriented Test Generation

- Two fundamental test generation steps
 - **ACTIVATE**, Excite, Provoke or Setup the Fault
 - Make Fault OBSERVABLE, Fault Sensitization
 - Find Primary Input Values that Cause
 - Error Signal in Faulty Circuit
 - For Single-Stuck-at-v Fault
 - Place v' at Fault Site
 - **PROPAGATE** the Resulting Error to a Primary Output
 - Path Sensitization
 - Find Primary Input Values that Sensitize
 - Error Signal to Primary Output

Copyright 2010, M. Tahoori TDS I: Lecture 7 4

Specific-Fault Oriented Test Generation

- Example: Test for $c/0$ is $w,x,y = 0,1,1$
 - ACTIVATE Fault $c/0$
 - Set $x = y = 1$ to make $c=1$
 - in Fault-free Circuit
 - PROPAGATE Value on c to f
 - Set $w = 0$ to sensitize c to f

```


graph LR
    x --> AND[&]
    y --> AND
    AND -- c --> OR[+]
    w --> OR
    OR -- f --> f
    
```

Copyright 2010, M. Tahoori TDS I: Lecture 7 5

Line Justification


- Find Input Assignment to Place Value v on Line g
- Algebraic Approach
 - Find Boolean Function Realized on line $g = G(X)$
 - Use Prime Implicant of $G(X)$ to Place 1 on g
 - Use Prime Implicate of $G(X)$ to Place 0 on g
- PROPAGATE Error (Fault Effect)
 - Algebraic Approach
 - Use Boolean Difference

Copyright 2010, M. Tahoori TDS I: Lecture 7 6



Boolean Difference

Copyright 2010, M. Tahoori TDS I: Lecture 7 7




Boolean Difference

- Shannon expansion
 - A Boolean function $f(X_1, X_2, \dots, X_n)$ can be expanded about any variable X_i
 - $f(X_1, X_2, \dots, X_n) = X_i'f(X_1, \dots, X_i = 0, \dots, X_n) + X_i f(X_1, \dots, X_i = 1, \dots, X_n)$
- Boolean Difference of $f(X_1, X_2, \dots, X_n)$ with respect to X_i
- Symbol is (partial derivation)

$$\frac{d f(X_1, X_i, \dots, X_n)}{d X_i}$$
- Definition is:
- $\frac{d f}{d X_i} = f_{X_i'} \oplus f_{X_i} = f(X_1, \dots, X_i = 0, \dots, X_n) \oplus f(X_1, \dots, X_i = 1, \dots, X_n)$

Copyright 2010, M. Tahoori TDS I: Lecture 7 8




Boolean Difference

- Example
 - $f = w + xy,$
 - $f_y = w$
 - $f_y = w + x$
 - $\frac{df}{dy} = (w) \oplus (w + x) = w'x$

Copyright 2010, M. Tahoori

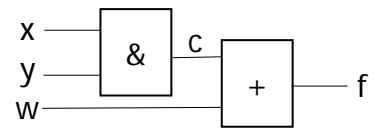
TDS I: Lecture 7

9



Boolean Difference

- $\frac{df(x,y,w)}{dy} = 1$
- for values of w and x for which f depends on y
- $\frac{df(x,y,w)}{dy} = 0$
- for values of w and x for which f is independent of y
- $\frac{df(w+xy)}{dy} = w'x$
- $w'x = 1, \text{ for } w=0, x=1$
 - When $w = 0, x = 1, w + xy = y$



```

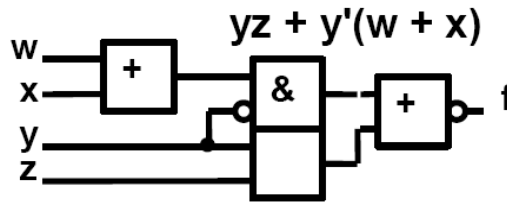
graph LR
    x --> AND[&]
    y --> AND
    AND -- c --> OR[+]
    w --> OR
    OR -- f --> f
            
```

Copyright 2010, M. Tahoori

TDS I: Lecture 7

10

Boolean Difference



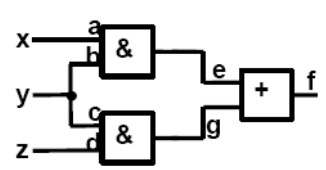
$yz + y'(w + x)$

- Example
 - $df/dy = fy' \oplus fy$
 - $= (w + x) \oplus z$
 - $= WZ' + XZ' + W'X'Z$

Copyright 2010, M. Tahoori TDS I: Lecture 7 11

Boolean Difference

- Test pattern generation
 - $df/dx = d(xy+yz)/dx = yz \oplus (y + yz) = yz'$
- Test for a/0 is $xyz = (110)$
 - Set $x = 1$ to *Provoke* Fault
 - Set $y = 1, z = 0$ to *Sensitize* Fault Site to Output



Copyright 2010, M. Tahoori TDS I: Lecture 7 12

Boolean Difference

- Test pattern generation
 - C/1
 - $df/dc = d(cx+yz)/dc = yz \oplus (x + yz) = x(y' + z) = x(y' + z')$
 - To Propagate Fault, Set $x = 1, y$ or $z = 0$


- $c = v' + w'$
- For c/1, must set $c = 0$,
 - so $v = w = 1$

Copyright 2010, M. Tahoori
TDS I: Lecture 7
13

Boolean Difference

- Algebraic Technique to Determine
 - Path Sensitization from Fault Site to Output, or
 - Fault Observability Conditions
- Used Mainly for Theoretical Studies

Copyright 2010, M. Tahoori
TDS I: Lecture 7
14




Path Tracing

Copyright 2010, M. Tahoori

TDS I: Lecture 7

15




Test Generation Using Path Tracing

- Notation
 - D Signal Value
 - 1 in Fault-free Circuit, 0 in Faulty Circuit
 - D' or \overline{D} Signal Value
 - 0 in Fault-free Circuit, 1 in Faulty Circuit
 - X
 - Signal Value is Unspecified

Copyright 2010, M. Tahoori

TDS I: Lecture 7

16




Notation

- Truth Table for AND

$a \backslash b$	0	1	X	D	\overline{D}
0	0	0	0	0	0
1	0	1	X	D	\overline{D}
X	0	X	X	X	X
D	0	D	X	D	0
\overline{D}	0	\overline{D}	X	0	\overline{D}

Copyright 2010, M. Tahoori TDS I: Lecture 7 17



Path Sensitization Method

- Fault Sensitization
 - Force tested node to opposite of fault value
- Fault Propagation (path sensitization)
 - Propagate the effect to one or more POs
- Line Justification
 - Justify internal signal assignments made to activate and sensitize fault
- These three steps may result in conflict
 - Different values are assigned to the same signal
 - Require **backtracking**

Copyright 2010, M. Tahoori TDS I: Lecture 7 18

Path Sensitization Method

- Example (B stuck-at 0)
- Fault activation
 - Requires $B = 1, f = D, g = D$
- Fault propagation
 - Three scenarios are possible
 - paths $f - h - k - L, g - i - j - k - L$, or both

Copyright 2010, M. Tahoori TDS I: Lecture 7 19

Path Sensitization Method

- Try path $f - h - k - L$
 - Requires $A = 1, j = 0, E = 1$
- Blocked at j
 - Since there is no way to justify 1 on i

Copyright 2010, M. Tahoori TDS I: Lecture 7 20

Path Sensitization Method

- Try simultaneous
 - paths $f - h - k - L$ and $g - i - j - k - L$
- Blocked at k because
 - D-frontier (chain of D or \overline{D}) disappears

Copyright 2010, M. Tahoori TDS I: Lecture 7 21

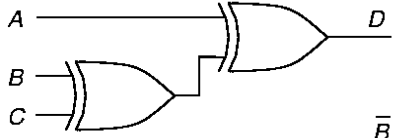
Path Sensitization Method

- Final try: path $g - i - j - k - L$
 - test found!

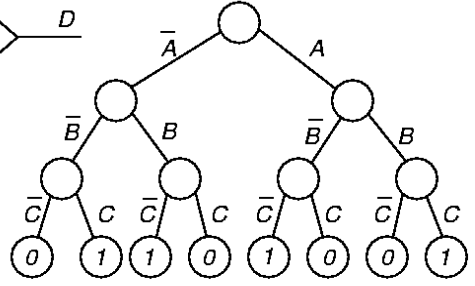
Copyright 2010, M. Tahoori TDS I: Lecture 7 22

Search Space Abstraction

- Binary Decision Tree (BDT)
 - The leaves represent the output of the good machine



(a) Circuit.



(b) Binary decision tree.

Copyright 2010, M. Tahoori TDS I: Lecture 7 23

Algorithm Completeness

- All ATPG programs implicitly search BDT
- Definition:
 - Algorithm is complete if it ultimately can search entire binary decision tree, as needed, to generate a test
- Untestable fault
 - No test for it even after entire tree searched
- Combinational circuits only
 - Untestable faults are redundant, showing the presence of unnecessary hardware

Copyright 2010, M. Tahoori TDS I: Lecture 7 24

ATPG Problem

- Ibarra and Sahni in 1975 showed that ATPG is NP_Complete
 - No polynomial-time algorithm is known
 - Presumed to be exponential
- These ATPG algorithms employ heuristics that
 - Find all necessary signal assignments for a test
 - As early as possible
 - Search as little of the decision space as possible

Copyright 2010, M. Tahoori TDS I: Lecture 7 25

Forward Implication

- Results in logic gate inputs that are significantly labeled so that output can be uniquely determined
- Example
 - AND gate forward implication table:

	<i>b</i>	0	1	X	D	\bar{D}
<i>a</i>	0	0	0	0	0	0
1	0	1	X	D	\bar{D}	
X	0	X	X	X	X	
D	0	D	X	D	0	
\bar{D}	0	\bar{D}	X	0	\bar{D}	

Copyright 2010, M. Tahoori TDS I: Lecture 7 26

Backward Implication

- Unique determination of all gate inputs when the gate output and some of the inputs are given
- Backward implication is implemented procedurally
 - Since tables are cumbersome for gates with more than 2 inputs

Copyright 2010, M. Tahoori
TDS I: Lecture 7
27

Implication Stack

- Push-down stack. Records:
 - Each signal set in circuit by ATPG
 - Whether alternate signal value already tried
 - Portion of binary search tree already searched
- Example
 - PIs were set in order A, C, E, and B
 - B was set to 1 but failed

Stack ptr. ●

Signal	Value	Alternative tried
A	1	NO
C	1	NO
E	1	NO
B	0	YES

Copyright 2010, M. Tahoori
TDS I: Lecture 7
28

Implication Stack after Backtrack

Stack ptr. ●

Signal	Value	Alternative tried
E	1	NO
B	0	YES
F	0	YES

..... Unexplored

—— Present Assignment

—— Searched and Infeasible

Copyright 2010, M. Tahoori

TDS I: Lecture 7

29


Objectives and Backtracing of ATPG

- Objective: desired signal value goal for ATPG
 - Guides it away from infeasible/hard solutions
 - Intermediate signal assignments may make it impossible to achieve it
- Backtrace: Determines which primary input and value to set to achieve objective
 - Use testability measures

Copyright 2010, M. Tahoori

TDS I: Lecture 7


30



Branch-and-Bound Search

- An efficiently search method of binary search tree
- Branching
 - At each tree level, selects which input variable to set to what value (0 or 1)
- Bounding
 - Avoids exploring large tree portions by restricting search decision choices
 - Complete exploration is impractical
 - Decision about bounding made with limited information
 - Uses heuristics

Copyright 2010, M. Tahoori TDS I: Lecture 7 31



Specific-Fault Oriented Test Generation

- Three Approaches
 - **D Algorithm**: Internal Line Values Assigned (Roth-1966)
 - D-cubes
 - Bridging faults
 - Logic gate function change faults
 - **PODEM**: Input Values Assigned (Goel – 1981)
 - X-Path-Check
 - Backtracing
 - **FAN**: Input and Internal Values Assigned (1983)

Copyright 2010, M. Tahoori TDS I: Lecture 7 32