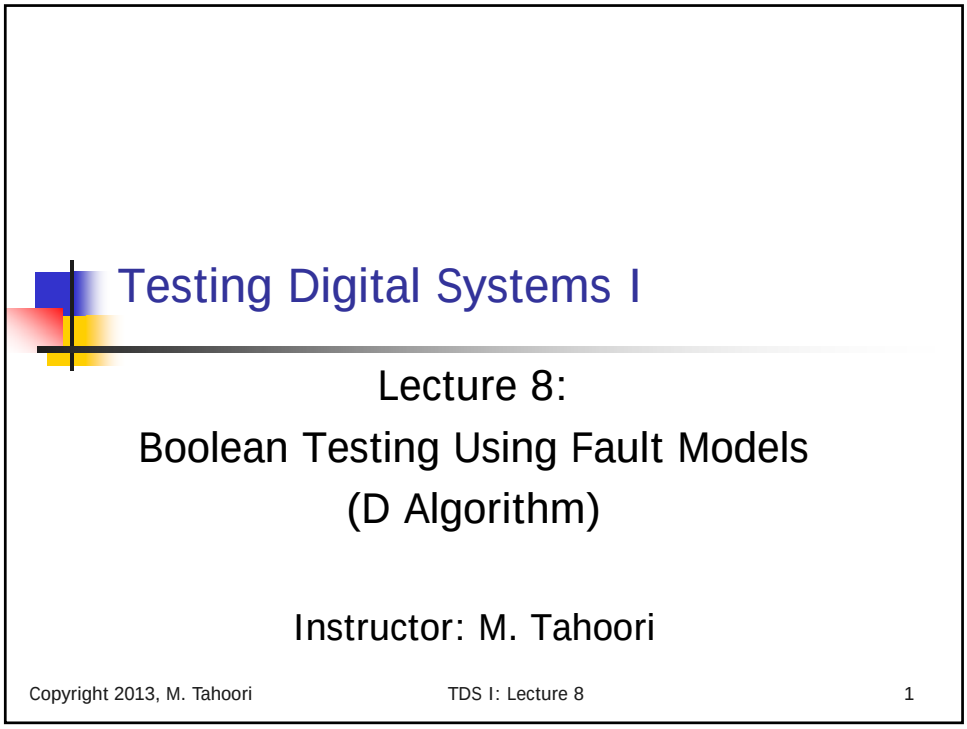
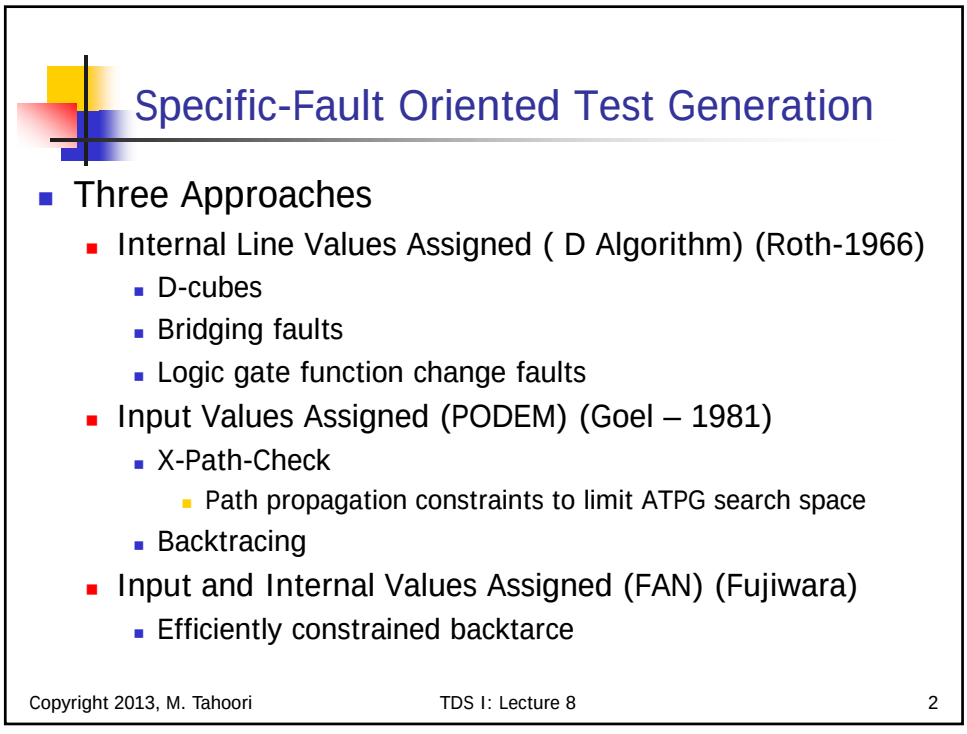
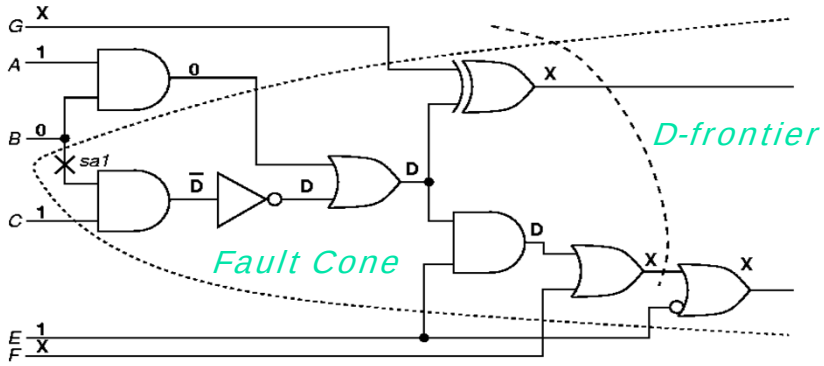

Testing Digital Systems I
Lecture 8:
Boolean Testing Using Fault Models
(D Algorithm)
Instructor: M. Tahoori
Copyright 2013, M. Tahoori
TDS I: Lecture 8
1


Specific-Fault Oriented Test Generation
Three Approaches
Internal Line Values Assigned ( D Algorithm) (Roth-1966)
D-cubes
Bridging faults
Logic gate function change faults
Input Values Assigned (PODEM) (Goel – 1981)
X-Path-Check
Path propagation constraints to limit ATPG search space
Backtracing
Input and Internal Values Assigned (FAN) (Fujiwara)
Efficiently constrained backtarce
Copyright 2013, M. Tahoori
TDS I: Lecture 8
2

## Fault Cone and D-frontier

- Fault Cone
  - Set of hardware affected by fault
- D-frontier
  - Set of gates closest to POs with fault effect(s) at input(s)



## D Algorithm

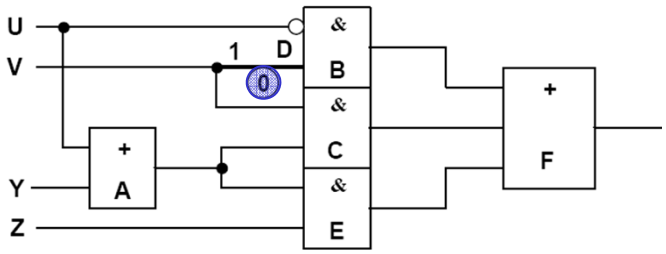
## D-Algorithm -- Roth IBM (1966)

- Fundamental concepts invented:
  - First complete ATPG algorithm
  - D-Cube
  - D-Calculus
  - Implications – forward and backward
  - Implication stack
  - Backtrack
  - Test Search Space

## D Algorithm

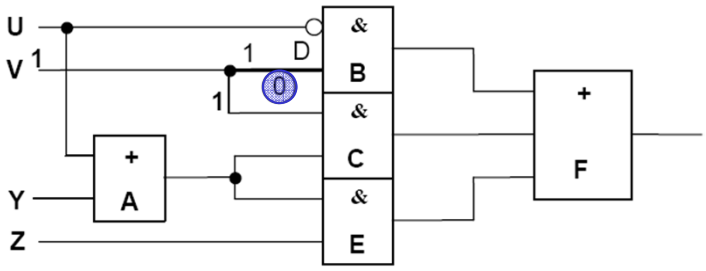
- Assigning internal line values
- Example:
  - Test for Stuck-at-0 on Lower Input to Gate B
  - Activate Fault — Put 1 on Faulty Lead



## D Algorithm

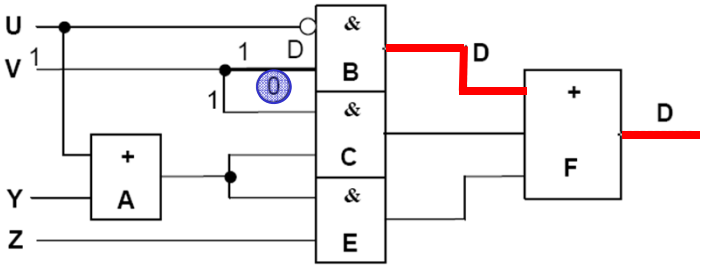
- Example
  - Test for Stuck-at-0 on Lower Input to Gate B
- Implication
  - Record Effects of Previous Assignments





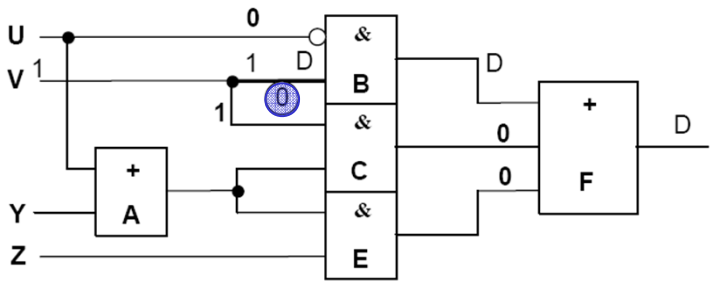

## D Algorithm

- Propagation
  - Select Path to Propagate D to Output
  - Single versus Multiple Path Propagation



## D Algorithm

- Propagation
  - Assign Required Gate Input Values
  - 0s on other inputs of OR, NOR Gates with D or $\overline{D}$ Input
  - 1s on other inputs of AND, NAND Gates with D or $\overline{D}$ Input






## Line Justification

- Find Input Assignment to Place Value v on Line g
- Path Tracing Approach
  - Propagate Signals using Element Functions
  - Must Choose Element Input Values and Paths
- Primitive cube of an element (gate) with output Z
  - List of prime implicants of Z and Z'

**AND**

| A | B | Z |
|---|---|---|
| 1 | 1 | 1 |
| 0 | — | 0 |
| — | 0 | 0 |

**NAND**

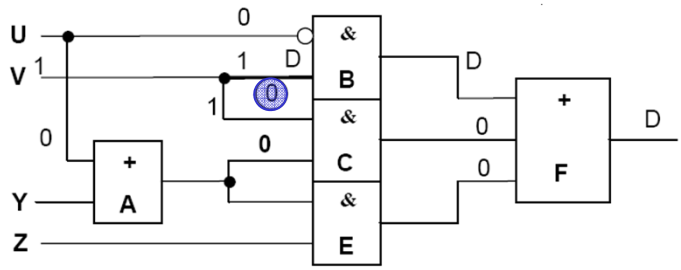
| A | B | Z |
|---|---|---|
| 1 | 1 | 0 |
| 0 | — | 1 |
| — | 0 | 1 |

Implication (no choices)

Decision (choices)

## D Algorithm

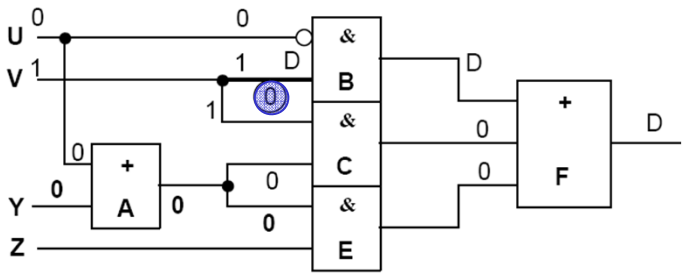
- Line Justification
  - Assign Required Gate Input Values
  - 0 on lower input of C to give 0 on output





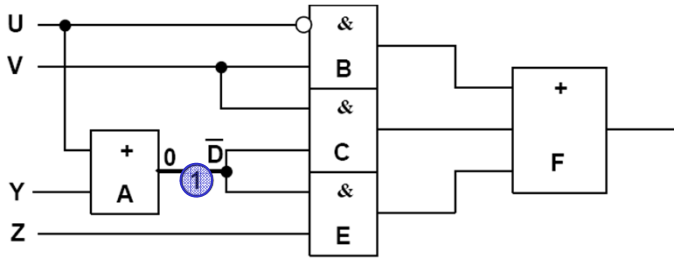

## D Algorithm

- Implication
  - Record Effects of Previous Assignments
- Test is U,V,Y,Z = 0,1,0,d
- DIFFICULTY
  - Internal Line Values May be Inconsistent



## D Algorithm

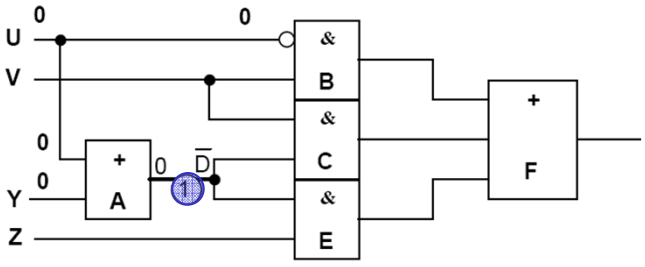
- Example
  - Test for Stuck-at-1 on Gate A Output
- Activate Fault
  - Put 0 on Faulty Lead





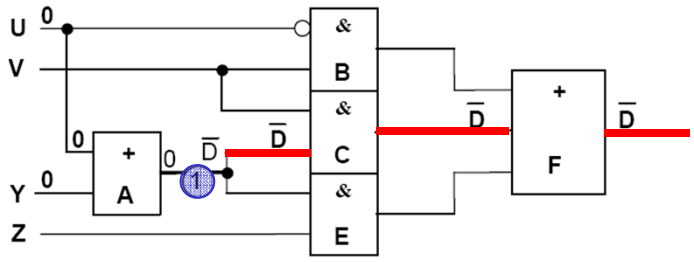

## D Algorithm

- Example
  - Test for Stuck-at-1 on Gate A Output
- Implication
  - Record Effects of Previous Assignments



## D Algorithm

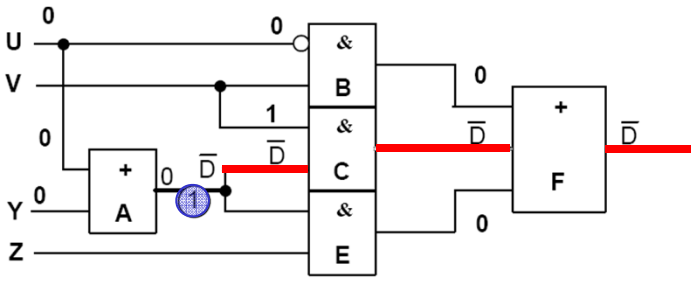
- Example
  - Test for Stuck-at-1 on Gate A Output
- Propagation
  - Select Path to Propagate D to Output





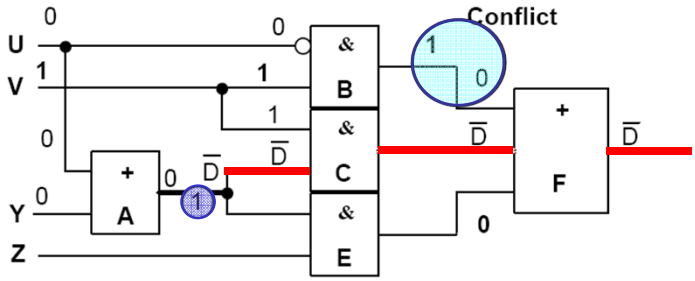

## D Algorithm

- Propagation
  - Assign Required Gate Input Values
  - 0s on other inputs of OR, NOR Gates with D or $\overline{D}$ Input
  - 1s on other inputs of AND, NAND Gates with D or $\overline{D}$ Input



## D Algorithm

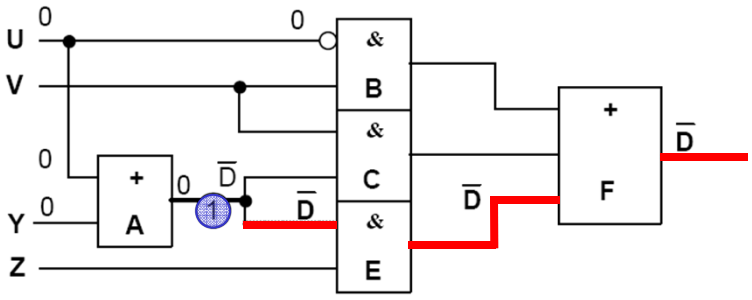
- Example
  - Test for Stuck-at-1 on Gate A Output
- Implication
  - Record Effects of Previous Assignments





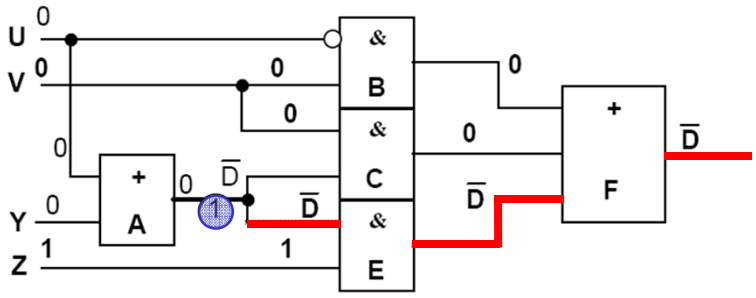

## D Algorithm

- Example
  - Test for Stuck-at-1 on Gate A Output
- Propagation
  - Select Alternate Path to Propagate D to Output



## D Algorithm

- Example
  - Test for Stuck-at-1 on Gate A Output
- Implication
  - Record Effects of Previous Assignments
- Test is U,V,Y,Z = 0,0,0,1

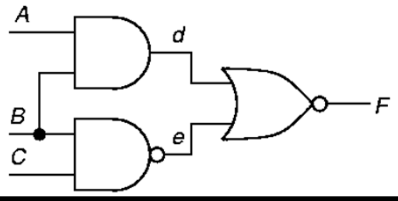





## D Calculus

## Singular Cover

- Singular Cover
  - Minimal set of input signal assignments to show essential prime implicants of Karnaugh map




| Gate | Inputs | | Output | Gate | Inputs | | Output |
|---|---|---|---|---|---|---|---|
| AND | *A* | *B* | *d* | NOR | *d* | *e* | *F* |
| 1 | 0 | X | 0 | 1 | 1 | X | 0 |
| 2 | X | 0 | 0 | 2 | X | 1 | 0 |
| 3 | 1 | 1 | 1 | 3 | 0 | 0 | 1 |



## D Algorithm

- D-Cube
  - A collapsed truth table entry
- Example
  - AND gate

| | *A* | *B* | *d* |
|---|---|---|---|
| Rows 1 & 3 | D | 1 | D |
| Reverse inputs | 1 | D | D |
| And two cubes | D | D | D |
| Interchange D and $\bar{D}$ | D | D | D |
| | 1 | D | D |
| | D | 1 | D |

## D Algorithm

- D Intersection
  - Defines how different D-cubes can coexist for different gates in logic circuit
  - If one cube assigns a specific signal value, the other cubes must assign either that same value or X
  - $1 \cap 1 = 1 \cap X = X \cap 1 = 1$
  - $0 \cap 0 = 0 \cap X = X \cap 0 = 0$
  - $X \cap X = X$

**ψ, ϕ represent incompatible assignments**
**μ, λ represent incompatibility if both present**

| ∩ | 0 | 1 | X | D | $\overline{D}$ |
|---|---|---|---|---|---|
| 0 | 0 | ϕ | 0 | ψ | ψ |
| 1 | ϕ | 1 | 1 | ψ | ψ |
| X | 0 | 1 | X | D | $\overline{D}$ |
| D | ψ | ψ | D | μ | λ |
| $\overline{D}$ | ψ | ψ | $\overline{D}$ | λ | μ |



## D Algorithm

- Primitive D-cube of Failure (PDF)
  - Models fault including
    - SA1: represented by $\overline{D}$
    - SA0: represented by D
  - Example: AND gate
    - PDF for output SA0 is 1 1 D
    - PDFs for output SA1 are 0 X $\overline{D}$, X 0 $\overline{D}$
- Propagation D-cube
  - Models conditions under which fault effect propagates through gate

## Implication Procedure

1. Model fault with appropriate PDF
2. Select propagation D-cubes to propagate fault effect to a PO (**D-drive** procedure)
3. Select singular cover cubes to justify internal circuit signals (**Consistency** procedure)

- D Algorithm's main problem
  - Selects cubes and singular covers arbitrarily



## D Algorithm