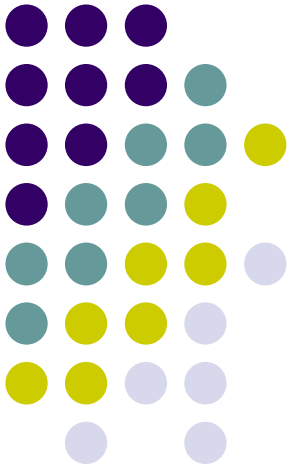
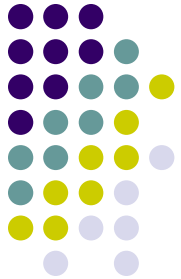
Chapter 3 Fault Modeling

錯誤模型

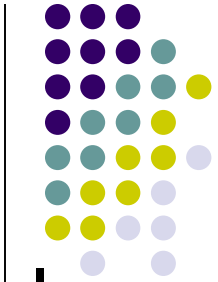


Outlines



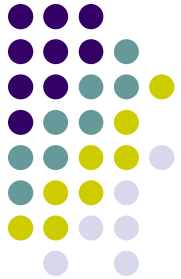
- Introduction
- Fault Models
- Properties of Stuck-at Faults
- Stuck-at Fault Collapsing

Fault Model and Structural Tests



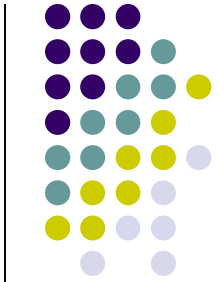
- Fault model is the foundation of structural testing methods
- Structural tests
  - Use the information of interconnected components (e.g., gates) to derived test regardless of the functions
  - Define faults
    - → fault coverage (quality evaluation)
    - → ATPG to generate tests for faults
    - → DfT for enhancing fault detection.

Characteristics of Fault Models



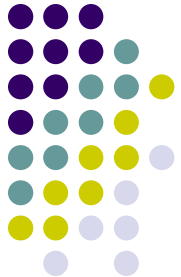
- Model the effects of physical defects on the logic function and timing
- Identifies target faults
  - Model faults (defects) most likely to occur
  - Depends on the process, design platform, design style, design level, etc.
- Limits the scope of test generation
  - Create tests only for the modeled faults

Levels of Fault Models

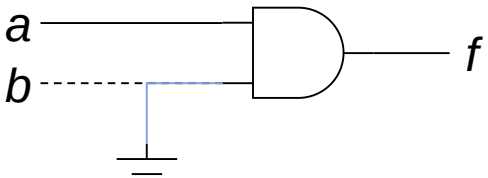


- Physical Defects
  - Silicon Defects
  - Photolithographic Defects
  - Mask Contamination
  - Process Variation
  - Defective Oxides
- Electrical Effects
  - Shorts (Bridging Faults)
  - Opens
  - Transistor Stuck-On/Open
  - Resistive Shorts/Opens
  - Change in Threshold Voltages
- Logical Effects
  - Logical Stuck-at 0/1
  - Slower Transition (Delay Faults)
  - AND-bridging, OR-bridging

Defect, Fault, and Error

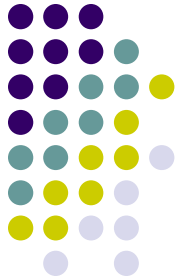


- Defect
  - Physical imperfection
  - The unintended difference between the manufactured hardware and its intended design
- Error
  - A wrong output signal produced by a defective system
- Fault
  - A representation of a defect at the abstracted function level



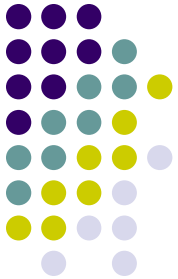
- One of the gate input terminal was mistakenly connected to ground
- Defect: short to ground
- Error: $f = 0$ when $a = b = 1$
- Fault: b stuck at 0

Common Fault Models

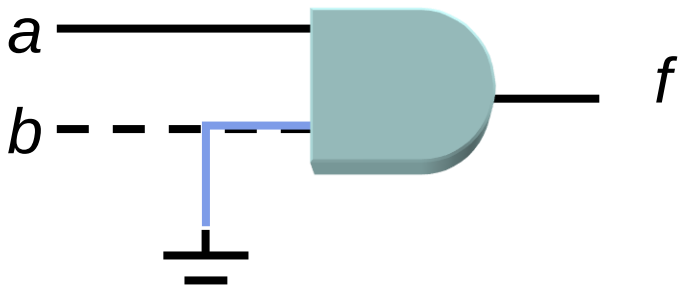


- Stuck-at faults
- Interconnect short and opens
- Transistor stuck-on/open faults
- Memory faults
- Delay faults

Single Stuck-At Fault Model

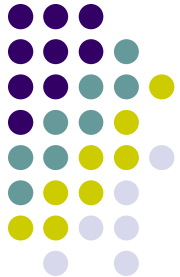


- Assumptions:
  - Only One line is faulty
  - Faulty line permanently set to 0 or 1
  - Fault can be at an input or output of a gate



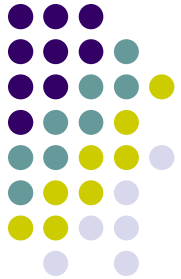
- One of the gate input terminal was mistakenly connected to ground
- Fault: *b* stuck at 0
- signal b will always be “0”

The Popularity of Single Stuck-At Faults



- Complexity is greatly reduced
- Technology independent
  - Can be applied to TTL, ECL, CMOS, BiCMOS etc.
- Design style independent
  - Gate array, standard cell, custom VLSI
- Detection capability of un-modeled defects
  - Empirically many defects accidentally detected by test derived based on single stuck-at fault

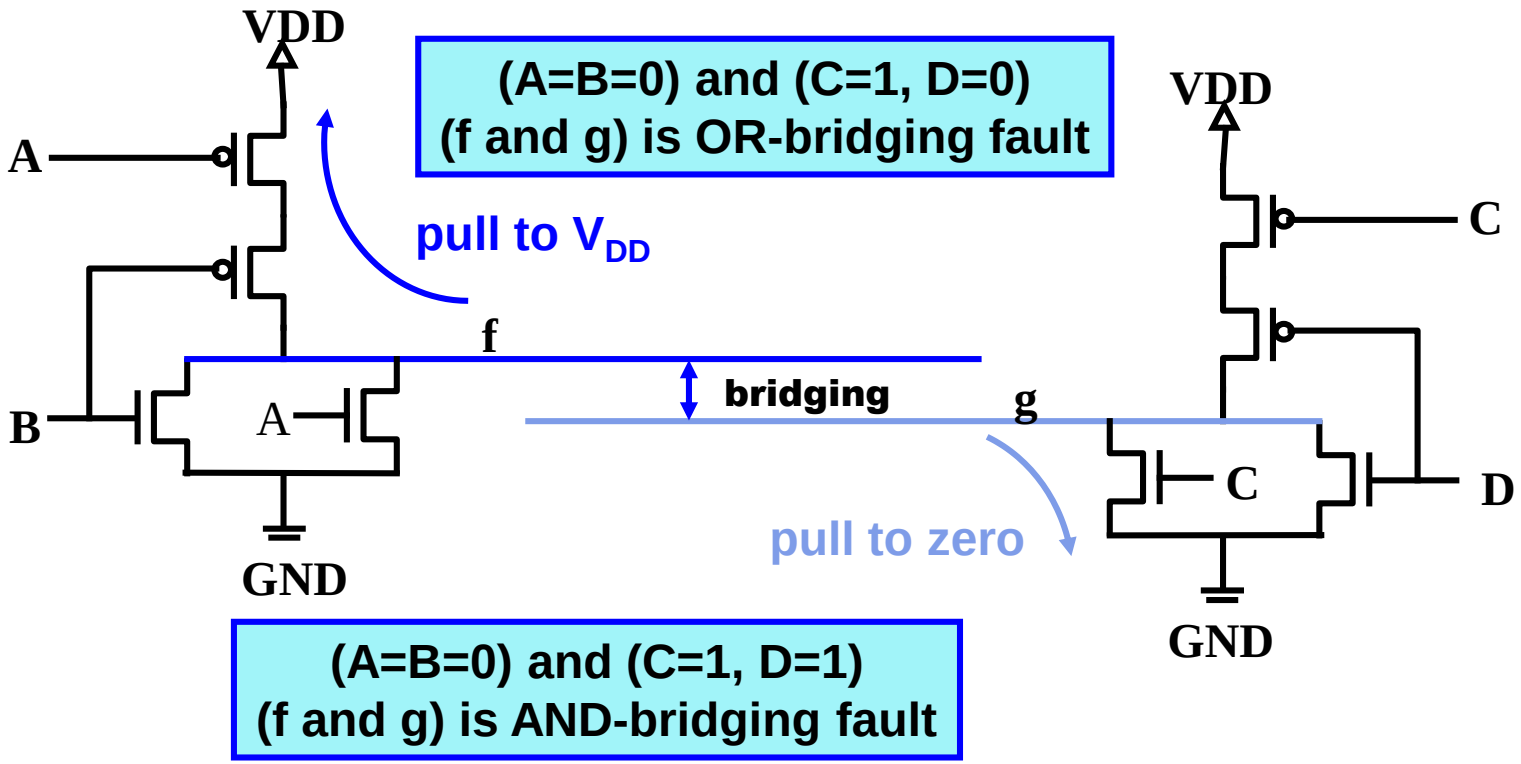
Multiple Stuck-At Faults



- Several stuck-at faults occur at the same time
- For a circuit with k lines
  - there are 2k single stuck-at faults
  - there are 3^k-1 multiple stuck-at faults
    - A line could be stuck-at-0, stuck-at-1, or fault-free
    - One out of 3^k resulting circuits is fault-free
- Most Multiple faults are covered by single-fault tests of combinational circuit:
  - 4-bit ALU (Hughes & McCluskey, ITC-84)
    All double and most triple-faults covered.
  - Large circuits (Jacob & Biswas, ITC-87)
    Almost 100% multiple faults covered for circuits with 3 or more outputs.

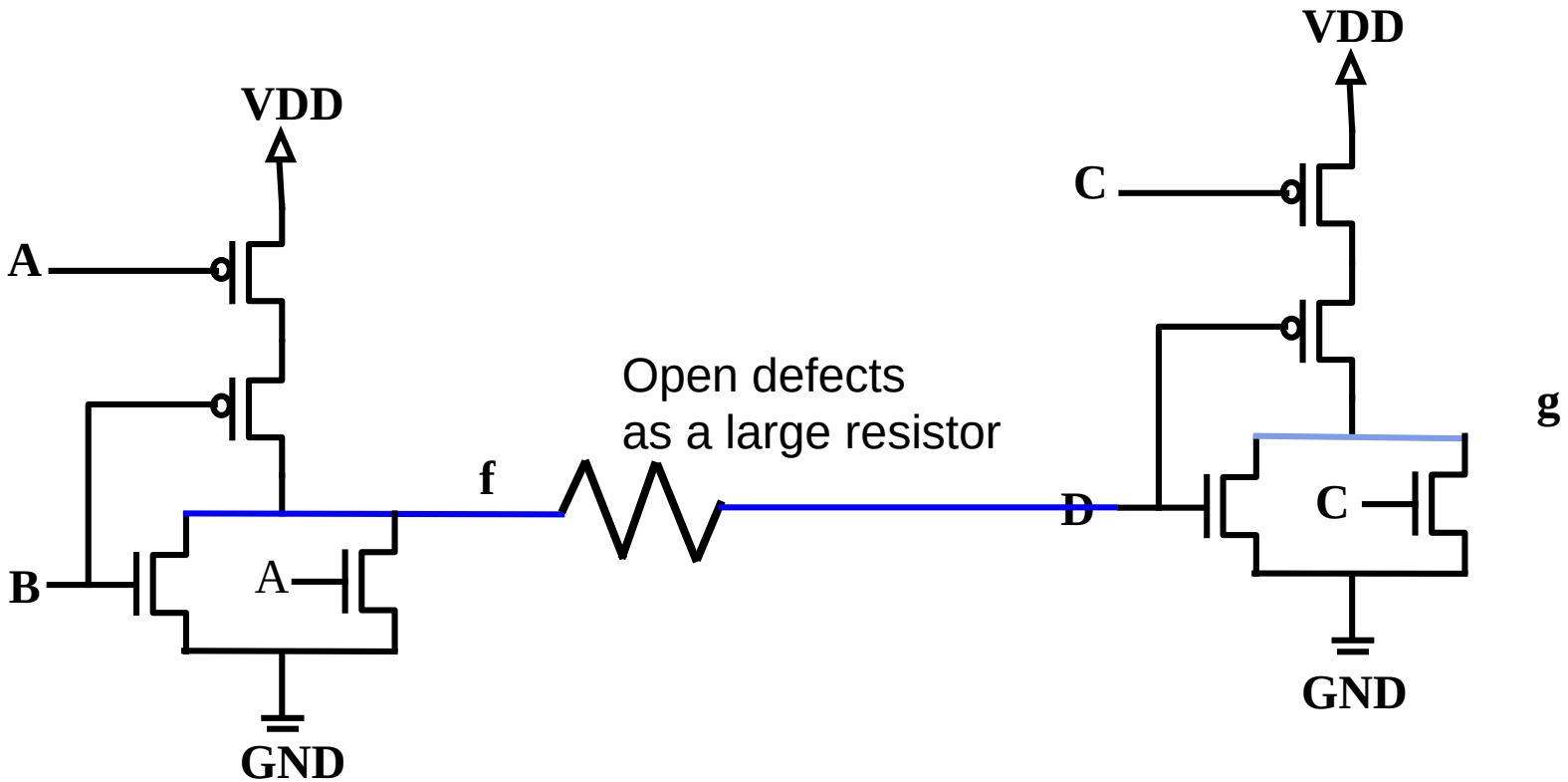
Bridging Faults For CMOS Logic

- Two or more normally distinct points (lines) are shorted together
- Could be AND-bridging or OR-bridging
  - depends on the inputs

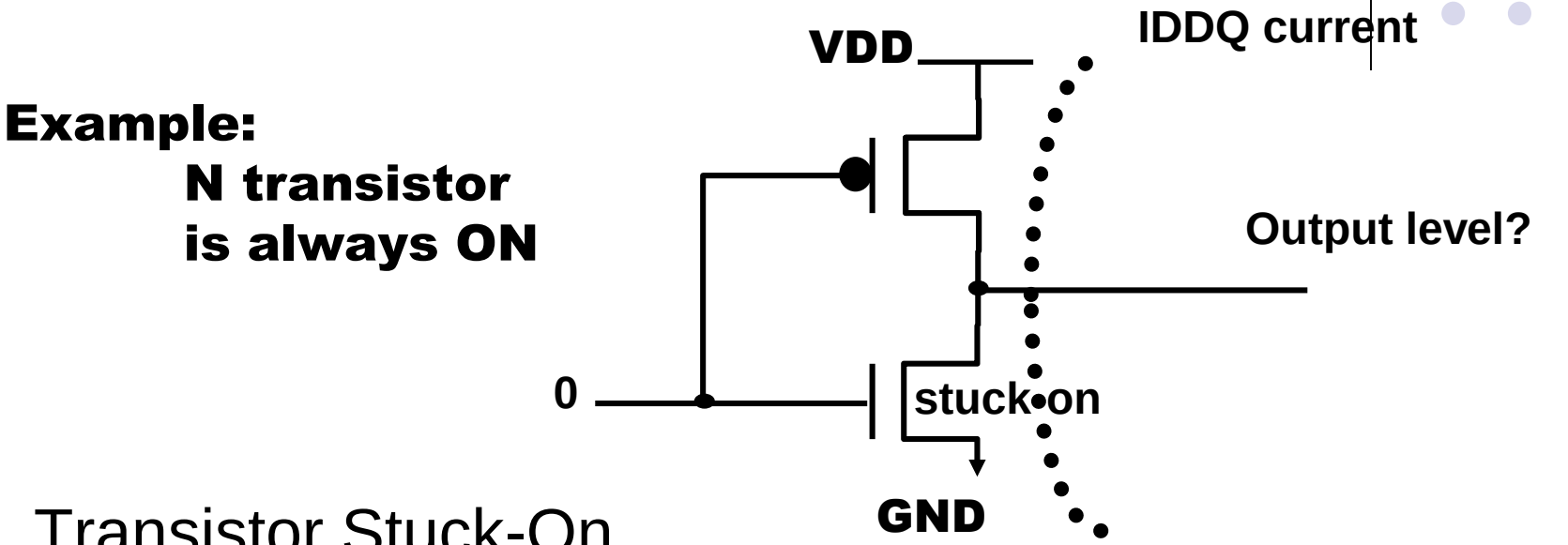


Interconnet Opens

- Open defects are due to a defect which splits up a node into two or more distinct nodes.
  - Large open (break): No current can go between the two ends of the open when a voltage is applied across it.
  - Narrow open: The opening is usually < 100 nm. In this case a small leakage current (by tunnel effect) go across the open.

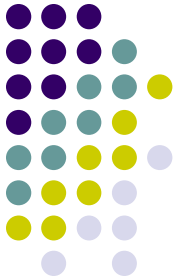


CMOS Transistor Stuck-On

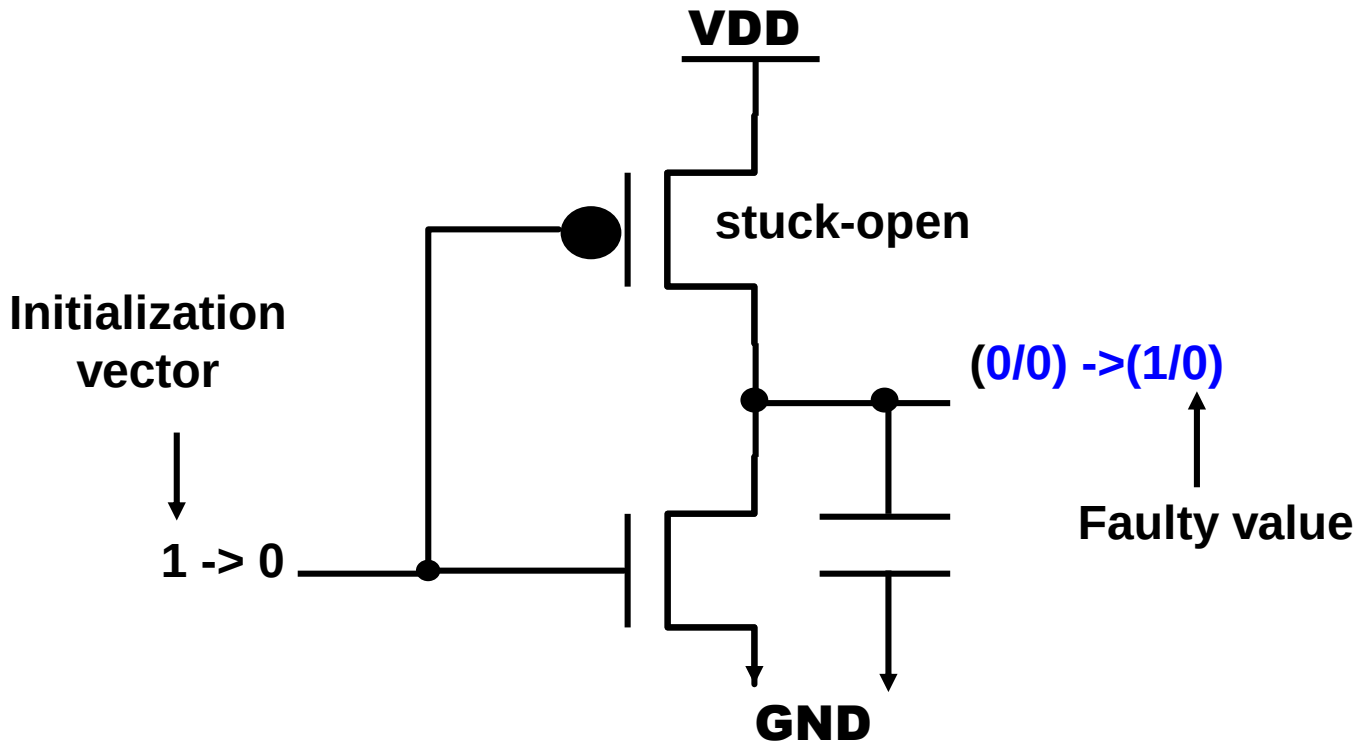


- Transistor Stuck-On
  - May cause ambiguous logic level
  - Depends on the relative impedances of the pull-up and pull-down networks
- When Input Is Low in the example
  - Both P and N transistors are conducting, causing increased quiescent current, called IDDQ fault

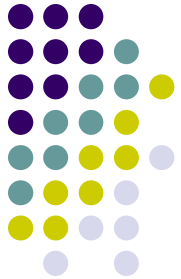
CMOS Transistor Stuck-Open



- Transistor stuck-open
  - May cause the output to be floating
  - The faulty cell has sequential behavior
- Stuck-open might require two vector tests

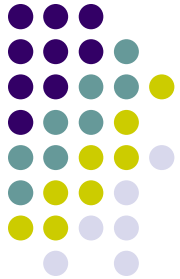


Delay Fault Models



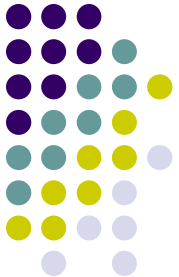
- Assumption
  - A “logical” model for decoupling delays from ATPG
  - Merge interconnect delays into pin-pin gate delays
  - All delay faults require two-cycle to detect
  - Usually apply with scan design and an internal clock (and some other DfT modification for aligning the ATE clock and the internal clock)
- Types
  - Transition fault
  - Path delay fault
  - Small delay defect (transition fault through the longest path)

Transition Fault Model (1/3)

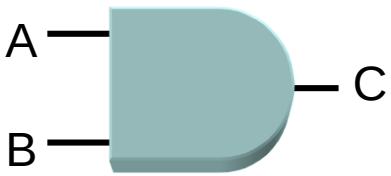


- Assumption
  - Delay fault affects only one signal in the circuit
  - The extra delay will prevent transitions to ANY primary output or flip-flop
    - Observation can be made at any PO or FF
- Two transition faults for each pin-to-pin signal
  - slow-to-rise fault
  - slow-to-fall fault

Transition Fault Model (2/3)



- Fault activation
  - Create a rising (or falling) signal transition at the fault site for a slow-to-rise fault (or slow-to-fall fault)
  - Need to set the values for two cycles
- Fault propagation
  - Propagate the faulty value at the second cycle
- Example
  - slow-to-rise on signal A on the following AND gate



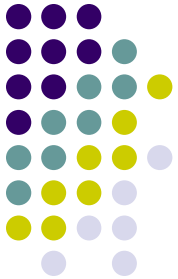
fault activation

signal	cycle 1	cycle 2
A	0	1
B	X	X
C	0	X

fault propagation

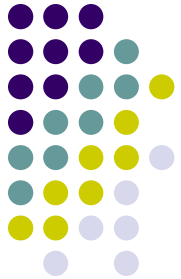
signal	cycle 1	cycle 2
A	0	1/0
B	X	1
C	0	1/0

Transition Fault Model (3/3)



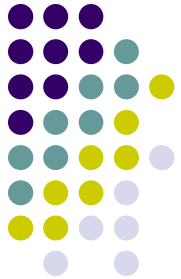
- Advantage
  - # of faults is linear w.r.t # of gates (signals)
  - ATPG tools for stuck-at faults can be used (why??)
- Problems
  - Fault size large enough to propagate through any paths (usually short paths)
  - Fault only affects one signal
- Despite the problems, transition fault model is good starting “base” set of delay tests

Path Delay Fault



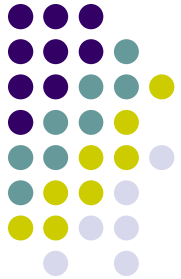
- Any path in a circuit can be a target for tests
  - A physical path is composed of a sequence of gates
  - A logical path is a physical path with a transition
- Advantage: model small variations on paths
  - All possible faults are captured.
- Problems:
  - number of paths grows exponentially with the circuit's size
  - Only a small subset of paths have good test patterns

Classification of Path Delay Faults



- Path sensitization
  - Single-path sensitizable
  - Robust
  - Non-robust
  - Functional sensitizable
  - Functional unsensitizable
- Functional (performance) redundant paths
  - Some path delay faults can never independently affect the performance
- Some path delay faults can result in better test quality but require higher complexity to generate the tests (or even cannot be generated)

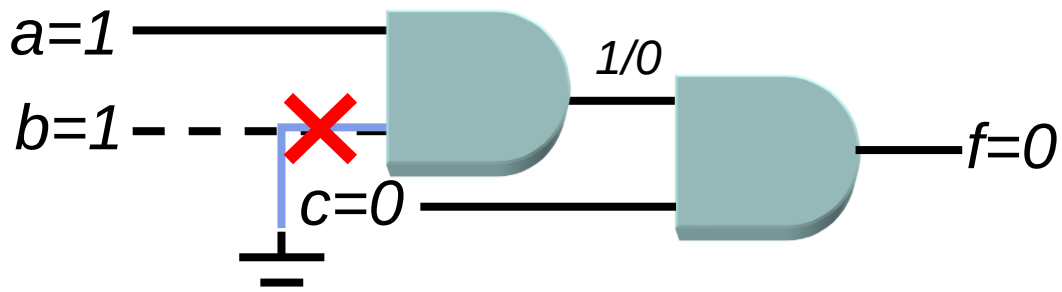
Small Delay Defect



- Idea: transition fault through the longest path
- Using STA result as a guidance
- Avoid of the drawback of transition faults whose corresponding test usually sensitize a short path
- The total number of faults are the same as the transition faults
- Require a more complicated ATPG

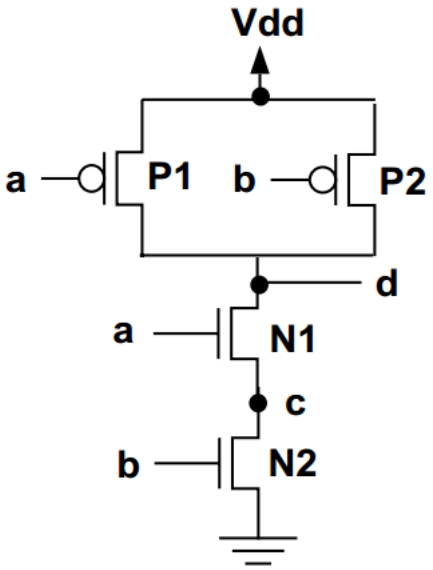
Pseudo-Stuck-At Fault Model for IDDQ testing

- It is similar to single stuck-at fault model, except that every cell output is considered observable by IDDQ testing.
  - The fault site at a gate input requires sensitization and propagation to an output of the same gate (but not to an output of the circuit) in order to be given credit for IDDQ fault detection.



Pseudo-Stuck-At Fault Model for IDDQ testing

- The exhaustive pseudo-stuck-at patterns for IDDQ testing cover all leakage faults in fully complementary combinational CMOS circuits.

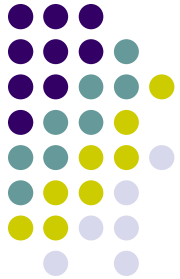


2-input NAND2

ab	cd	Detected Pseudo Stuck-At Faults	Detected leakage Faults (total $_6C_2$ = 15 leakage faults)
01	01	a-sa1 d-sa0	<b, 0> <c, 1> <d, 0> <a, 1> <a, b> <a, d> <b, c> <c, d> <0, 1>
10	11	b-sa1 d-sa0	<a, 0> <c, 0> <d, 0> <b, 1> <a, b> <b, c> <b, d> <0, 1>
11	00	a-sa0 b-sa0 d-sa1	<a, 0> <c, 0> <c, 1> <d, 1> <a, c> <a, d> <b, c> <b, d> <0, 1>

S.T., Zachariah, “A comparative study of pseudo stuck-at and leakage fault model”, IEEE Int’l Conf. on VLSI Design 1999

Memory Faults



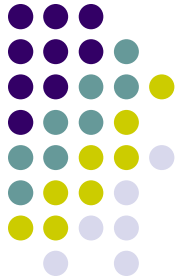
- Parametric Faults
  - Speed
  - Power Consumption
  - Noise Margin
  - Data Retention Time
- Functional Faults
  - Stuck Faults in Address Register, Data Register, and Address Decoder
  - Cell Stuck Faults
  - Adjacent Cell Coupling Faults
    - the presence of a faulty signal depends on the signal values of the neighboring cells
  - Pattern-Sensitive Faults
    - Pattern sensitivity between cells

0	**0**	**0**
0	**d**	**b**
0	**a**	**0**

a=b=0 => d=0

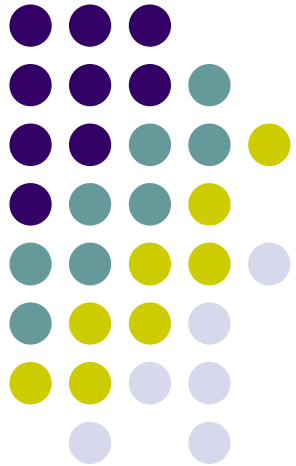
a=b=1 => d=1

Fault Model Summary



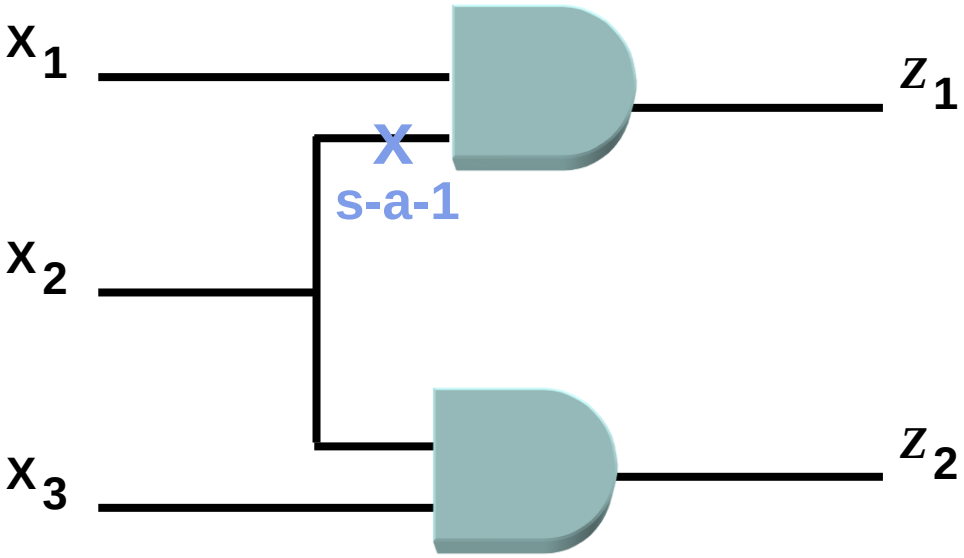
- Current practice
  - Single stuck-at faults
  - Transition faults
  - Small delay defect
- Other models to enhance defect coverage
  - N-detect faults
  - Interconnect open faults and bridging faults
  - Stuck-open (CMOS)
  - Path delay faults (high performance circuits)
  - Cell aware faults (defect-based faults)

Properties of Stuck-at Faults



Definition Of Fault Detection

- A test (vector) *t* detects a fault *f* iff
  - *t* detects *f* $\Leftrightarrow$ $\mathbf{z(t) \neq z_f(t)}$
  - Note that there can be multiple outputs for z(t)
- **A fault *f* is said to be detectable**
  - If there exists a test *t* that detects *f*; otherwise, *f* is undetectable.
- Example

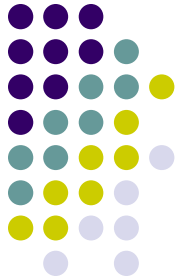


$z_1 = x_1 x_2$ $\quad z_2 = x_2 x_3$

$z_{1f} = x_1$ $\quad z_{2f} = x_2 x_3$

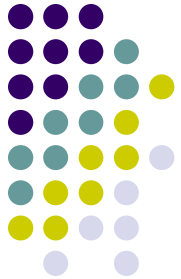
The test (x1,x2,x3) = (100) detects *f* because $z_1(100)=0$ while $z_{1f}(100)=1$

Redundancy of Faults

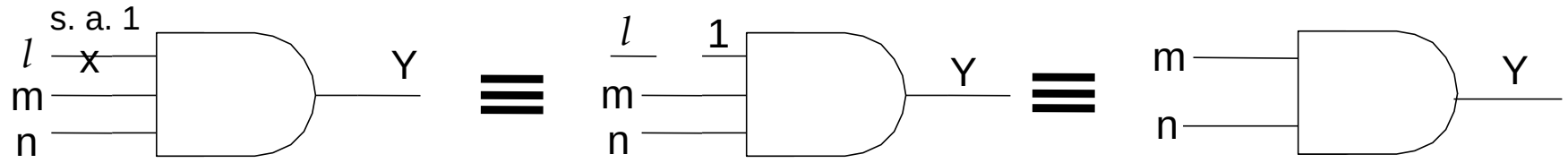


- **For combinational circuits, an undetectable fault is corresponding to a redundant wire.**
  - Undetectable faults do not change the function of the circuit
  - The wire can be connected to a constant value without changing circuit's function.

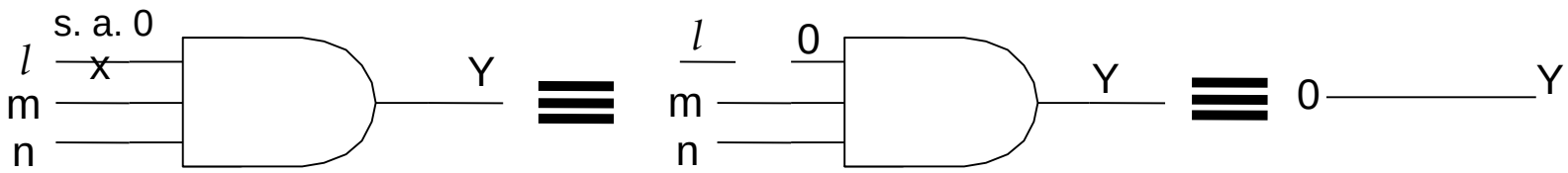
Examples of Simplifying Circuits with Redundancy



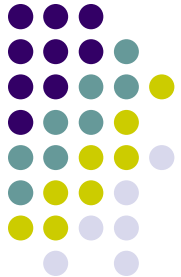
- If *l* s.a.1 is undetectable, the gate can be simplified as



- If *l* s.a.0 is undetectable, the entire gate can be removed & replaced by a constant 0 wire



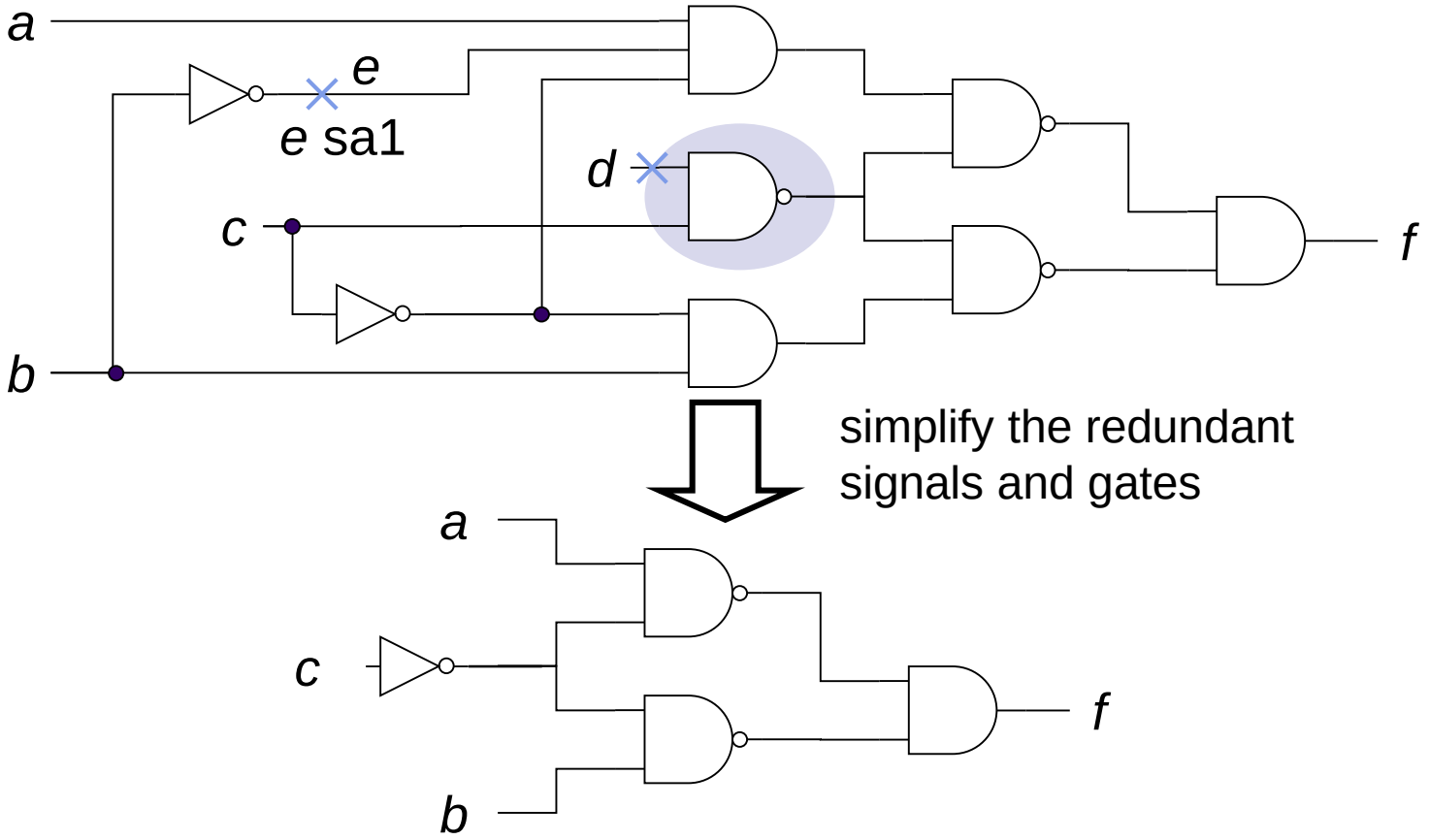
Cause of Circuit Redundancy



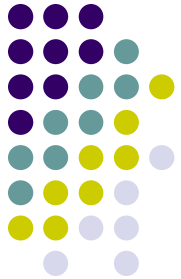
- Design to satisfy certain physical characteristics
  - Speed: carry look-ahead adders
  - Fault tolerant circuits: triple module redundant (TMR)
- Un-intentional redundancy by design partitioning.
  - Under specified requirements.

Example: Redundancy Removal

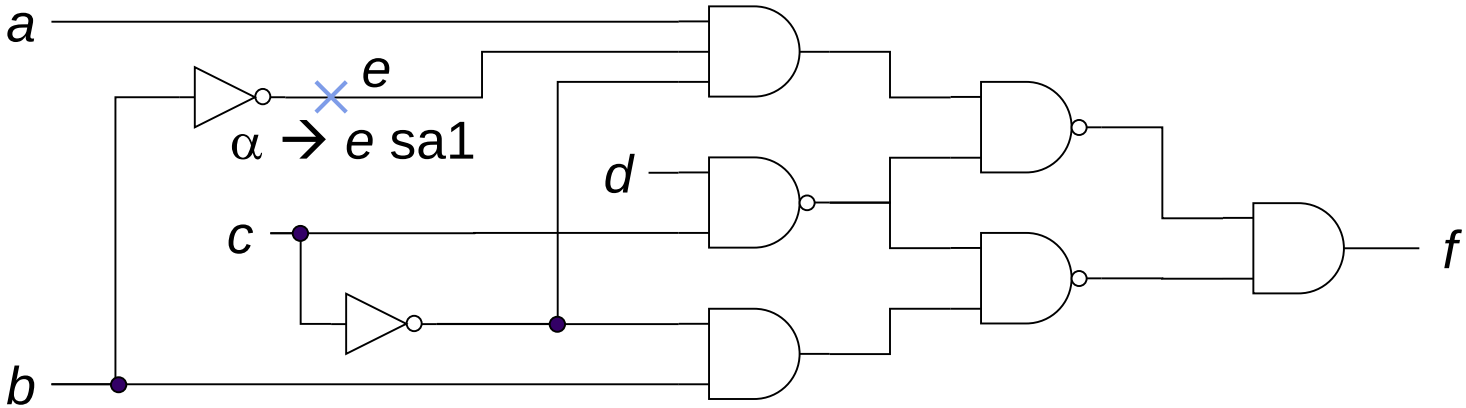
- Redundant faults: e s-a-1, d s-a-0 and d s-a-1
- The NAND gate with d is redundant



Fault Masking

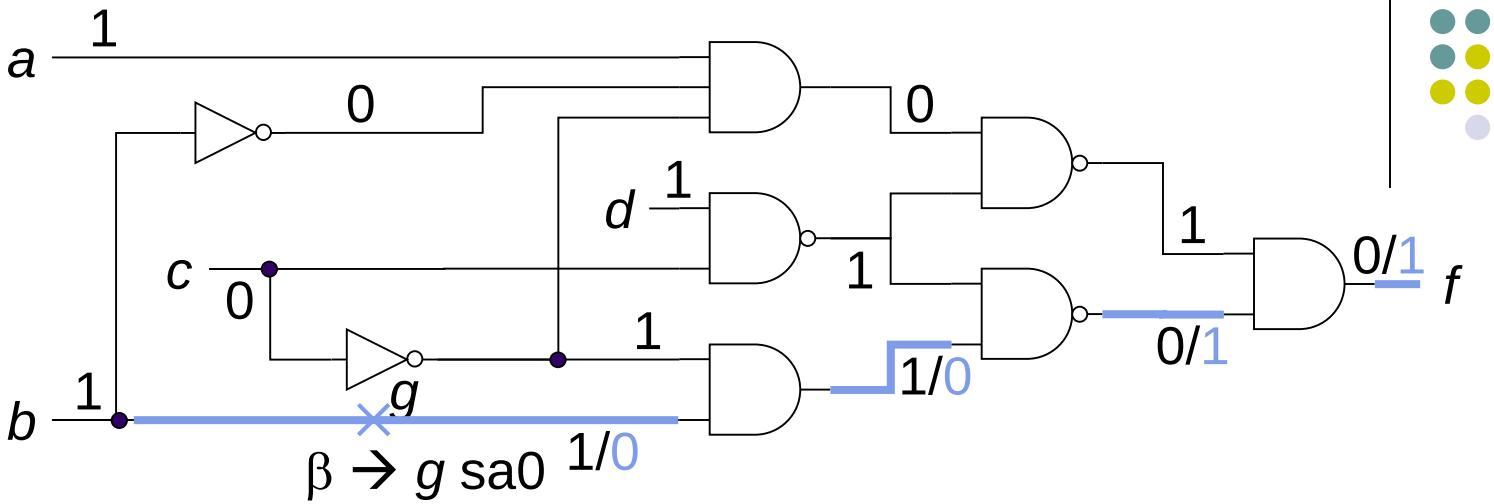


- **The existence of an undetectable fault may invalidates the test for another fault.**
  - **Test pattern masking by undetectable faults**

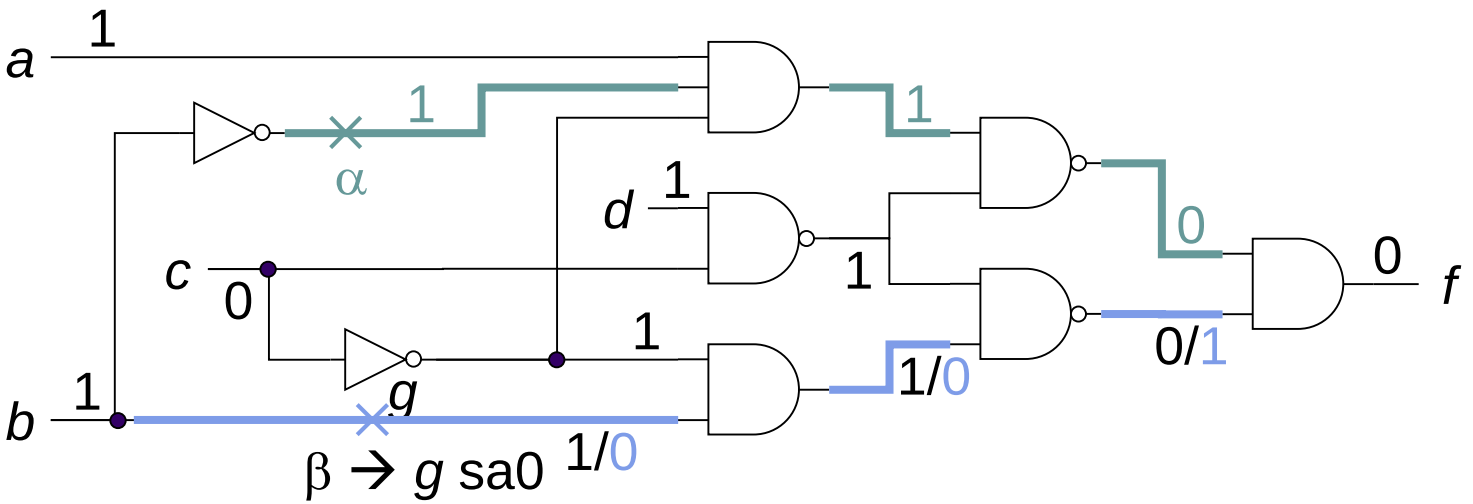


Fault α is undetectable, i.e., $f(t) = f\alpha(t)$ for all t.

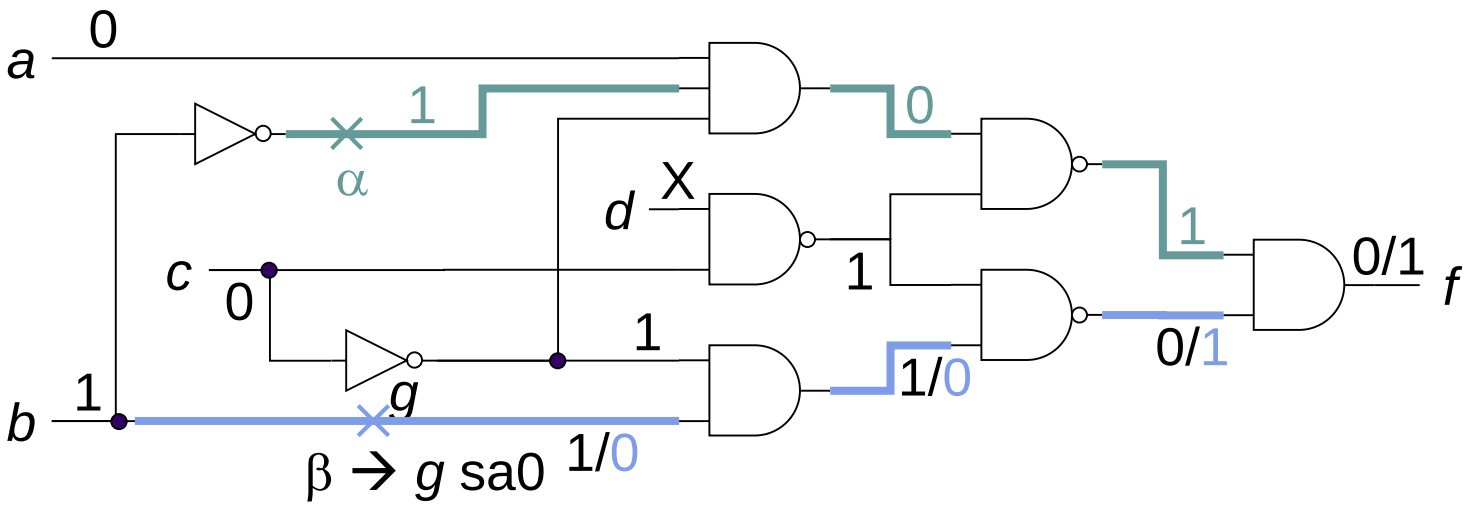
- Fault β can be detected by test vector 1101. (X10X)



- In the presence of the undetectabe fault α, the test vector 1101 becomes invalidate for fault β.



- But fault β can also be detected by test vector 010X with fault α.

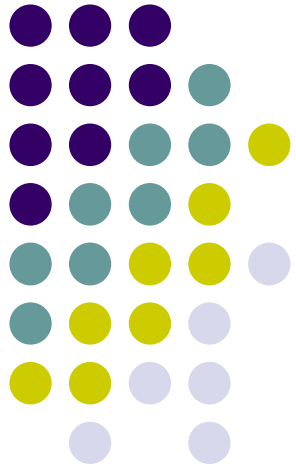


Stuck-at Fault Collapsing

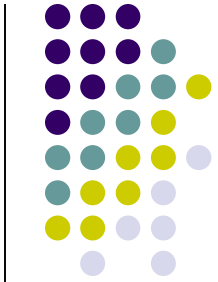
Fault Equivalence

Fault Dominance

Checkpoint Theorem



Fault Equivalence



- Distinguishing test
  - A test t distinguishes faults α and β if

$$Z_{\alpha}(t) \oplus Z_{\beta}(t) = 1$$

- Equivalent Faults
  - Two faults, α & β are said to be equivalent in a circuit , iff the function under α is equal to the function under β for any input combination (sequence) of the circuit.
  - No test can distinguish between α and β
  - In other words, test-set(α) = test-set(β)

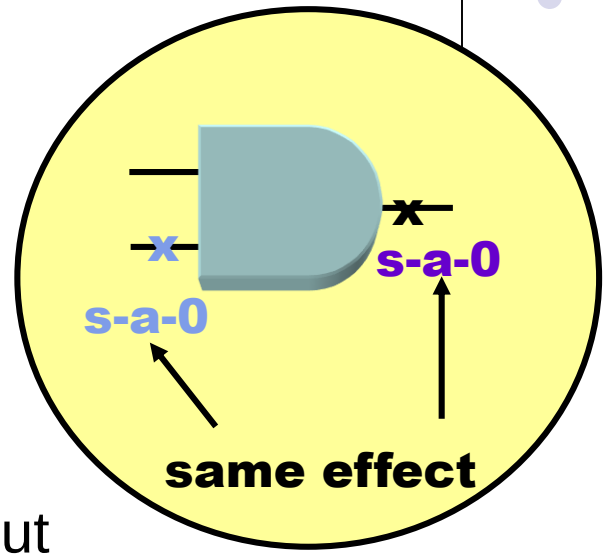
Equivalence Analysis of a Single Gate

AB	C	A sa1	B sa1	C sa1	A sa0	B sa0	C sa0
00	0			1			
01	0	1		1			
10	0		1	1			
11	1				0	0	0

- Fault Equivalence Class
  - (A s-a-0, B s-a-0, C s-a-0)
- Faults that can be ignored:
  - A s-a-0, B s-a-0, or C s-a-0

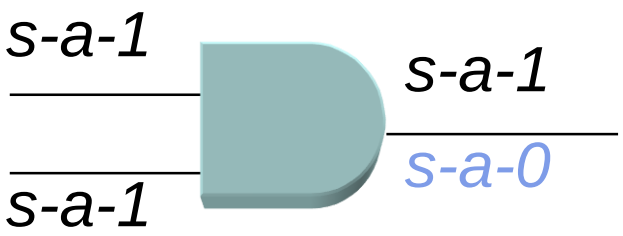
Fault Equivalence of Faults on a Gate

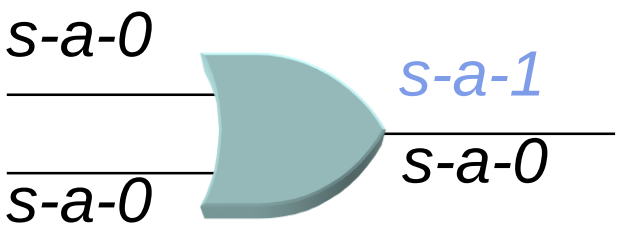
- AND gate:
  - all s-a-0 faults are equivalent
- OR gate:
  - all s-a-1 faults are equivalent
- NAND gate:
  - all the input s-a-0 faults and the output s-a-1 faults are equivalent
- NOR gate:
  - all input s-a-1 faults and the output s-a-0 faults are equivalent
- Inverter:
  - input s-a-1 and output s-a-0 are equivalent
    input s-a-0 and output s-a-1 are equivalent

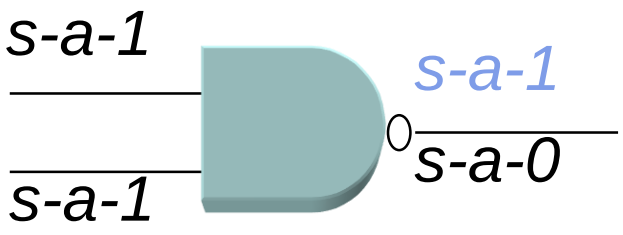


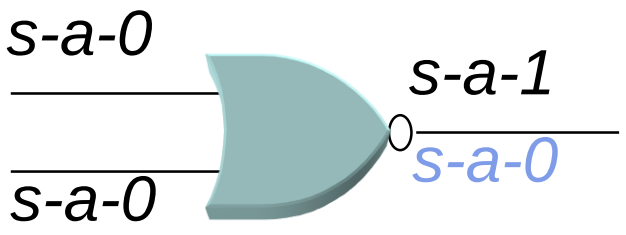
Equivalence Fault Collapsing of a Single Gate

- *n+2* instead of *2(n+1)* faults need to be considered for *n*-input gates



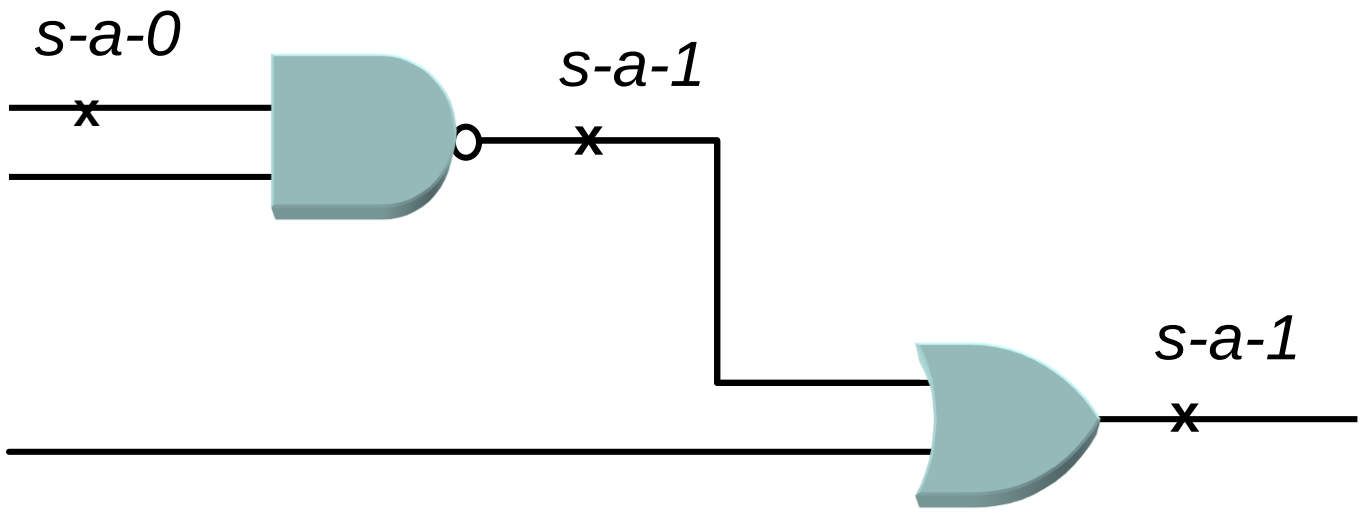






Equivalent Fault Group

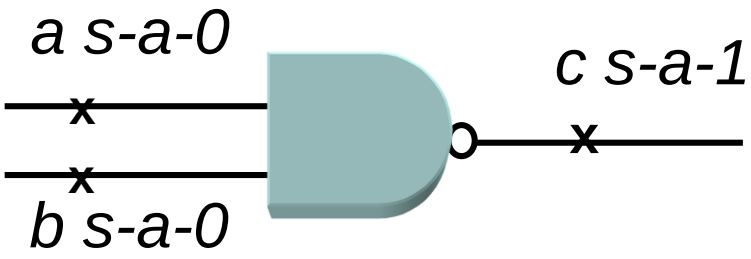
- In a combinational circuit, many faults may form an equivalent group
  - These equivalent faults can be found by sweeping the circuit from the primary outputs to the primary inputs
  - Transitive Rule: When $\alpha == \beta$ and $\beta == \gamma$, then $\alpha == \gamma$

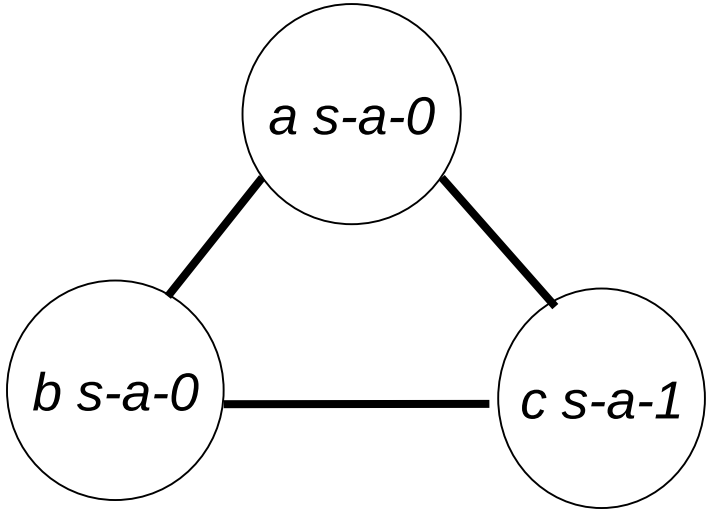


Three faults shown are equivalent !

Finding Equivalent Fault Groups

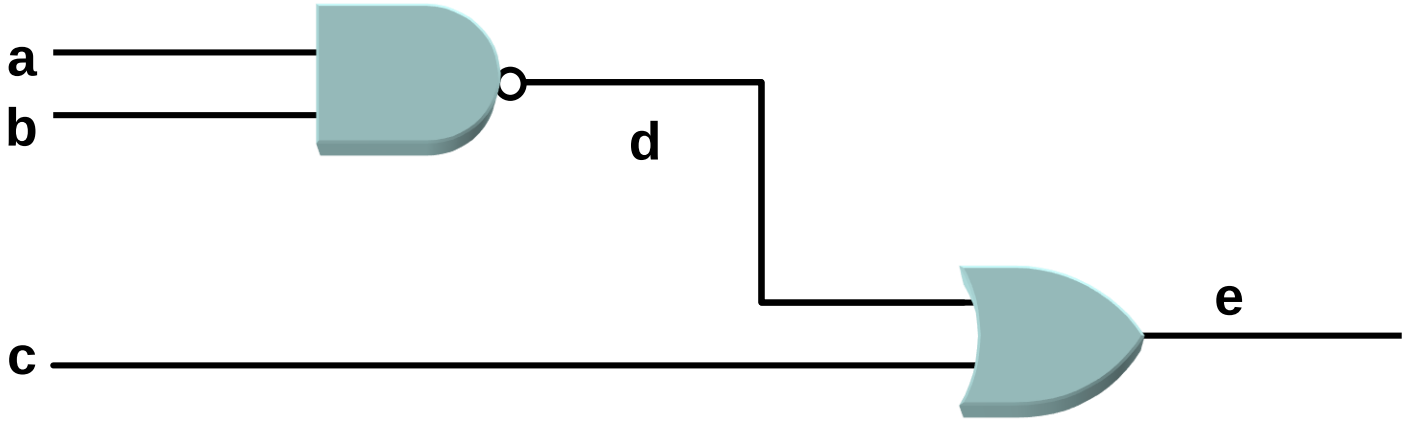
- Construct a Graph
  - A node is a fault
  - When there is a link between two node, the two faults are equivalent

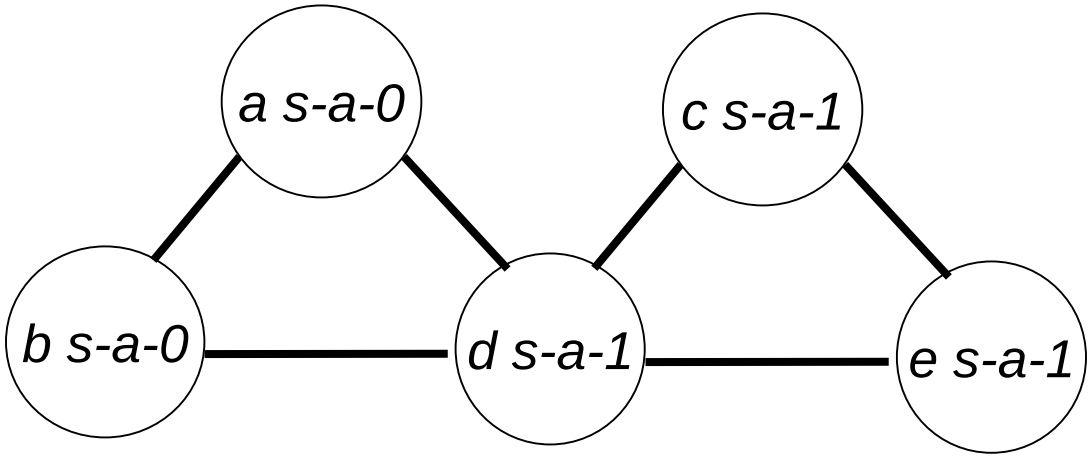




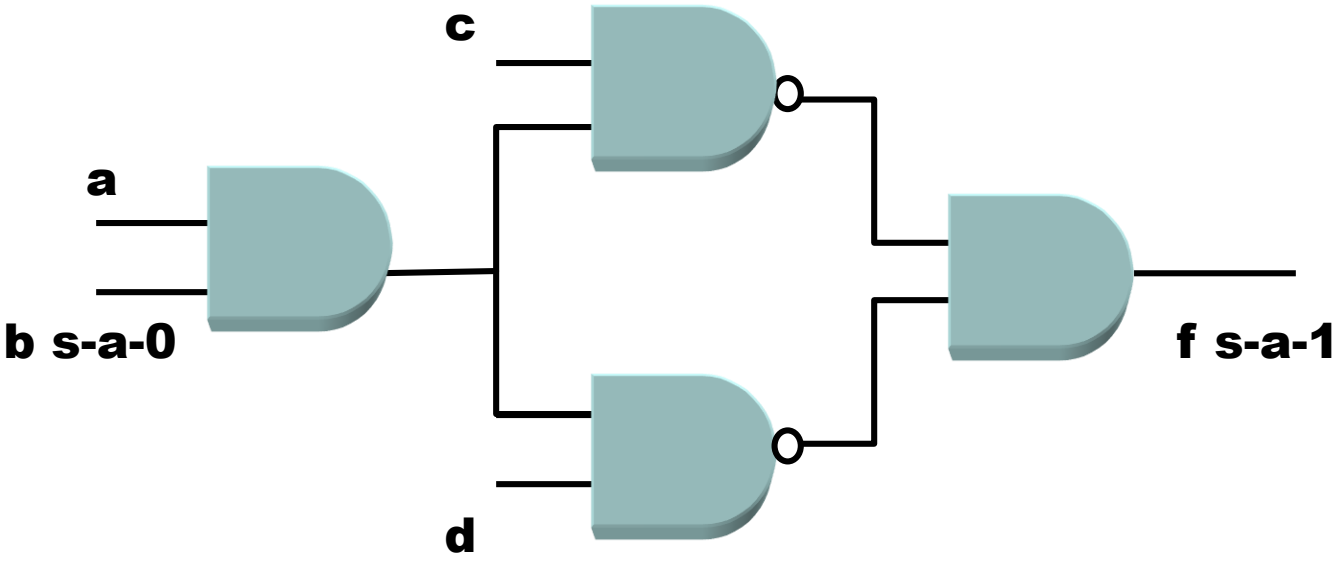
Example of Finding Equivalent Fault Groups

- Construct a bigger fault equivalent graph by sweeping the netlist from PO's to PI's





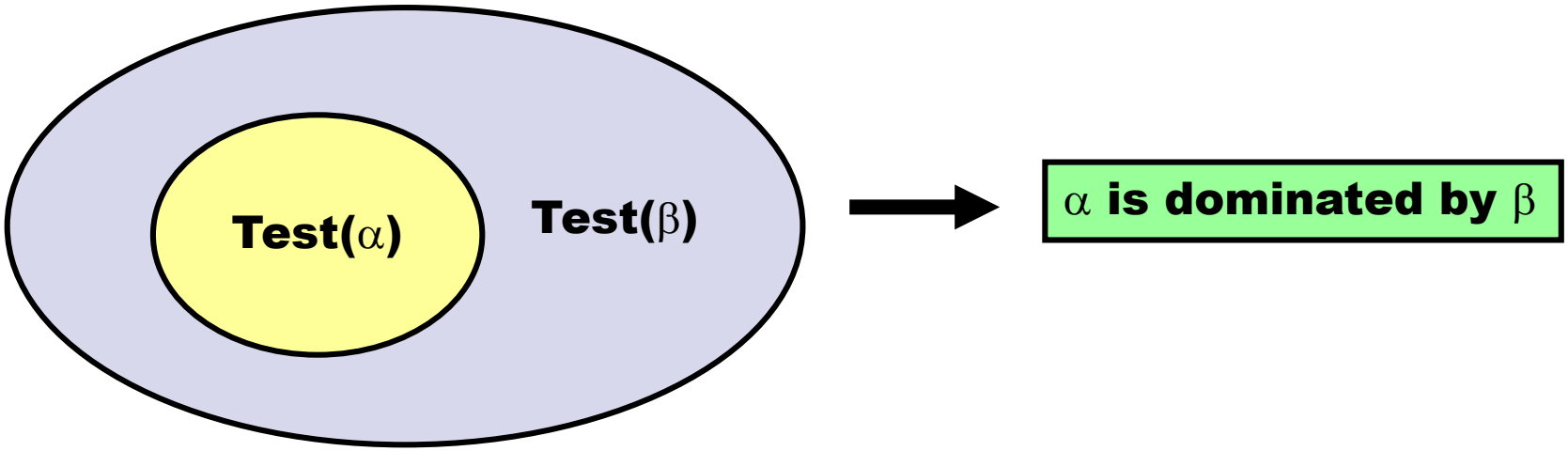
Limitation of Structural Analysis



- There is no rule to guarantee equivalence between faults of branches.
- It is a special case here b s.a.0 == f s.a.1
  - In general, they cannot be identified by simple structure analysis.

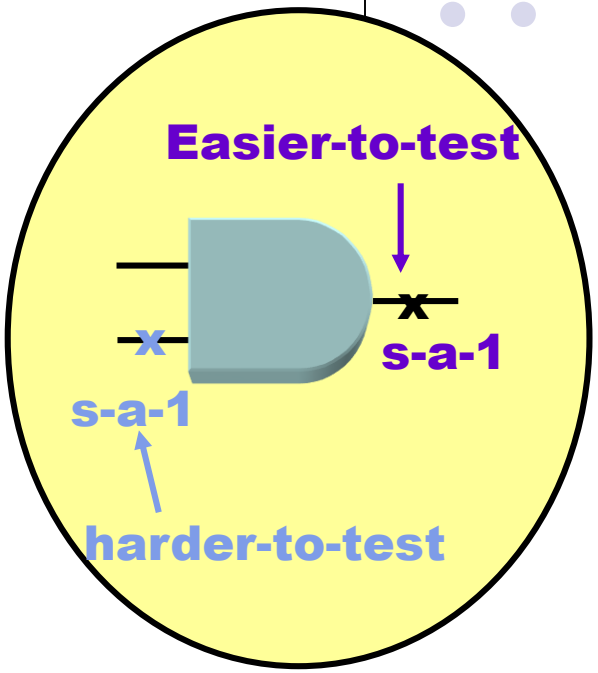
Fault Dominance

- Dominance Relation
  - A fault β is said to *dominate* another fault α in an irredundant circuit, iff every test (sequence) for α is also a test (sequence) for β.
  - I.e., test-set(β) > test-set(α)
  - No need to consider fault β for fault detection once α is detected



Fault Dominance

- AND gate:
  - Output *s-a-1* dominates any input *s-a-1*
- NAND gate:
  - Output *s-a-0* dominates any input *s-a-1*
- OR gate:
  - Output *s-a-0* dominates any input *s-a-0*
- NOR gate:
  - Output *s-a-1* dominates any input *s-a-0*
- Dominance fault collapsing:
  - The reduction of the set of faults to be analyzed based on dominance relation



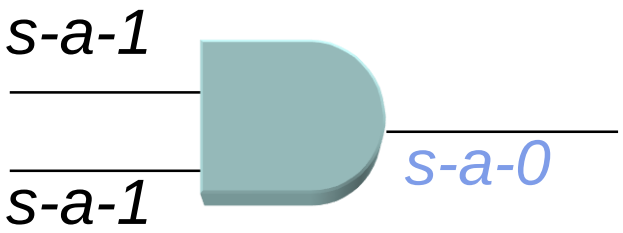
Dominance Analysis of a Single Gate

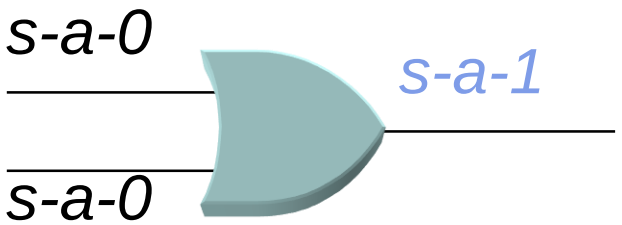
AB	C	A sa1	B sa1	C sa1	A sa0	B sa0	C sa0
00	0			1			
01	0	1		1			
10	0		1	1			
11	1				0	0	0

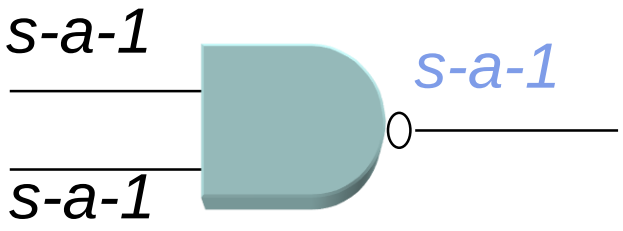
- Fault Dominance Relations
  - (C s-a-1 > A s-a-1) and (C s-a-1 > B s-a-1)
- Faults that can be ignored for test generation:
  - C s-a-1

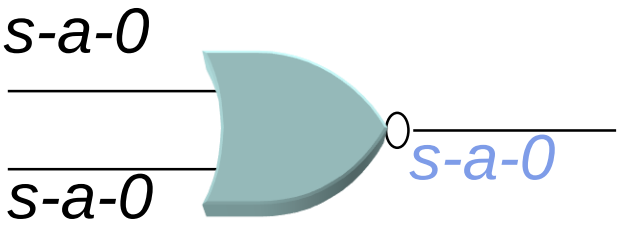
Complete Fault Collapsing of a Single Gate

- Equivalence + Dominance
  - For each *n*-input gate, we only need to consider *n+1* faults during test generation



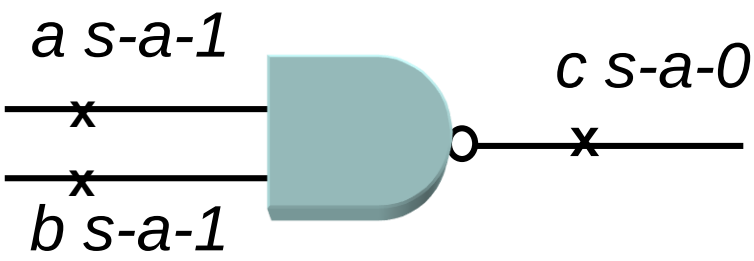


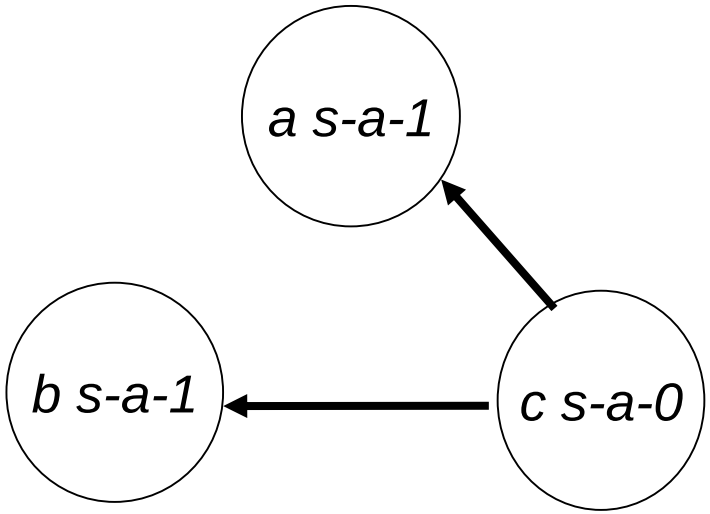




Dominance Analysis for a Group of Faults

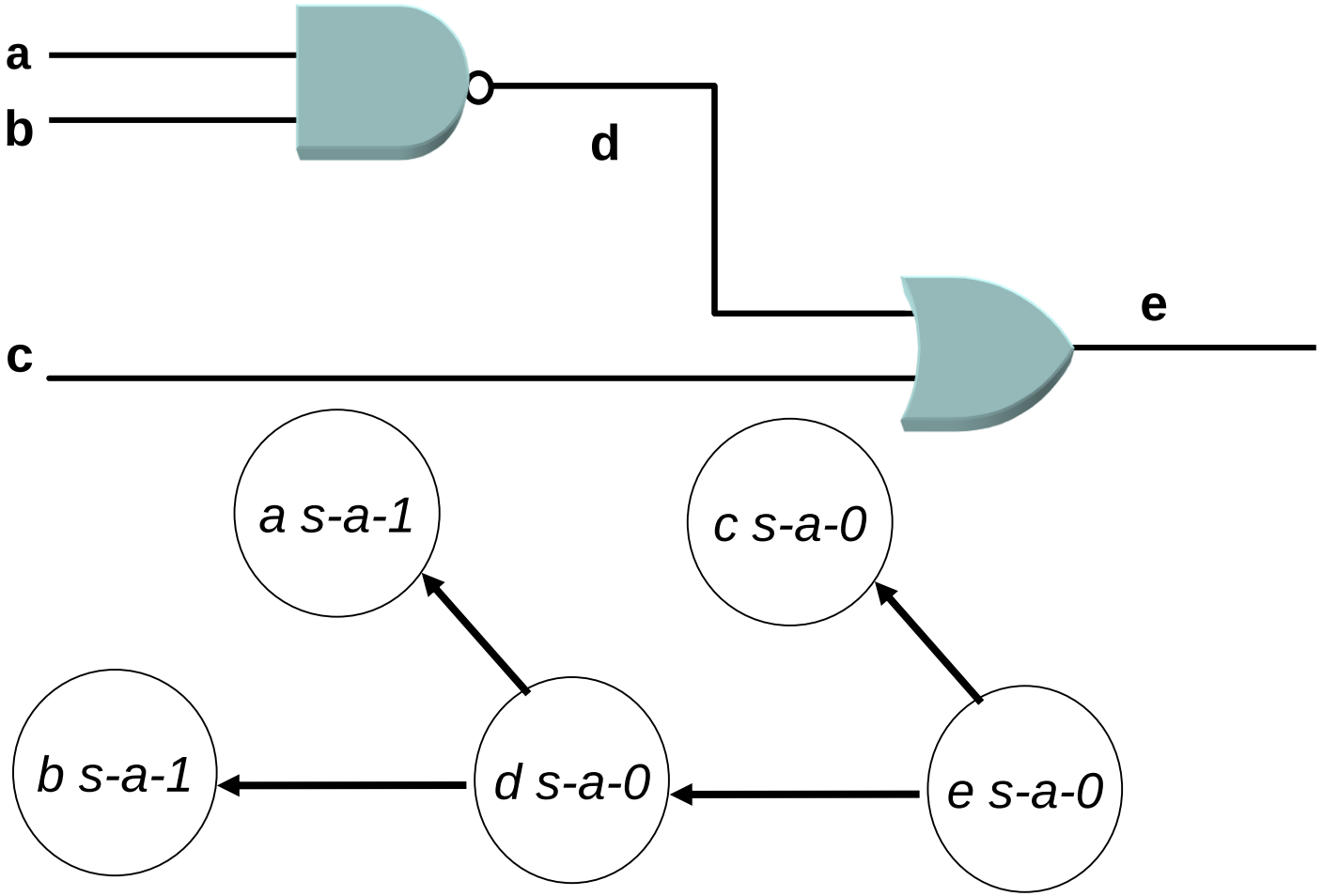
- Construct a dominance graph
  - A node is a fault
  - When fault α dominates fault β, then an arrow is pointing from α to β



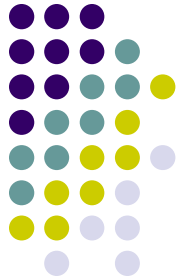


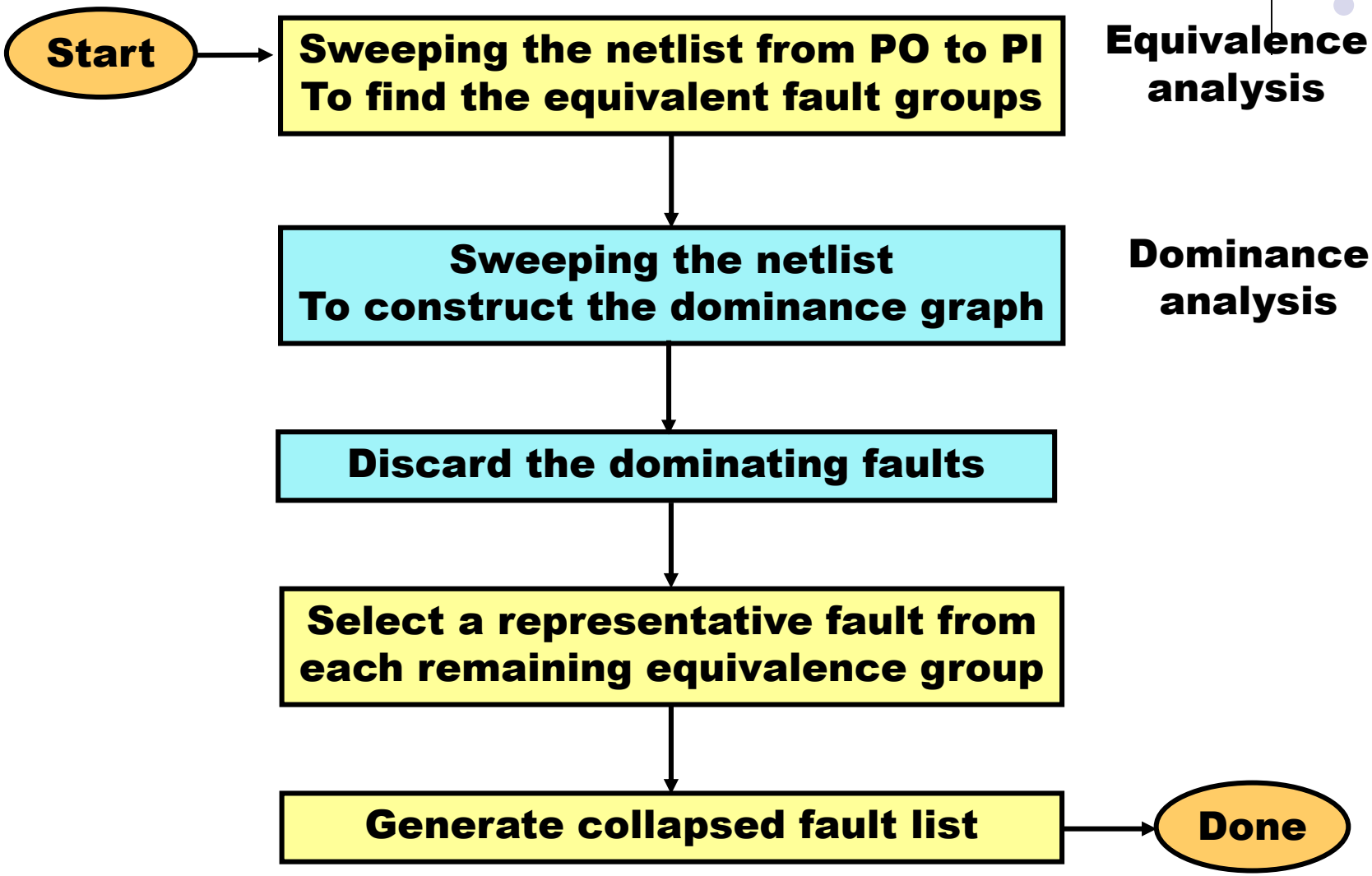
Example of Finding Dominant Fault Groups

- Construct a bigger fault equivalent graph by sweeping the netlist from PO's to PI's

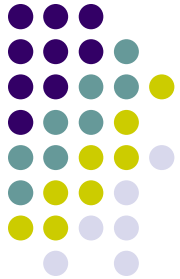


Fault Collapsing Flow

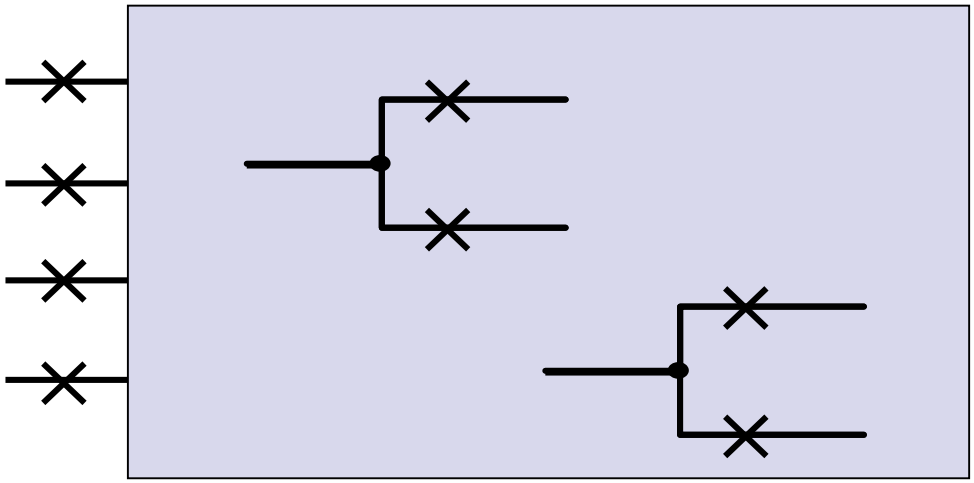




Checkpoint Theorem

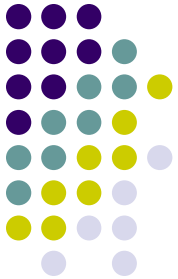


- Checkpoints Theorem
  - Checkpoints = primary inputs + fanout branches
  - In a combinational circuit, any test which detects all single stuck faults on all checkpoints detects all single faults in the circuit.

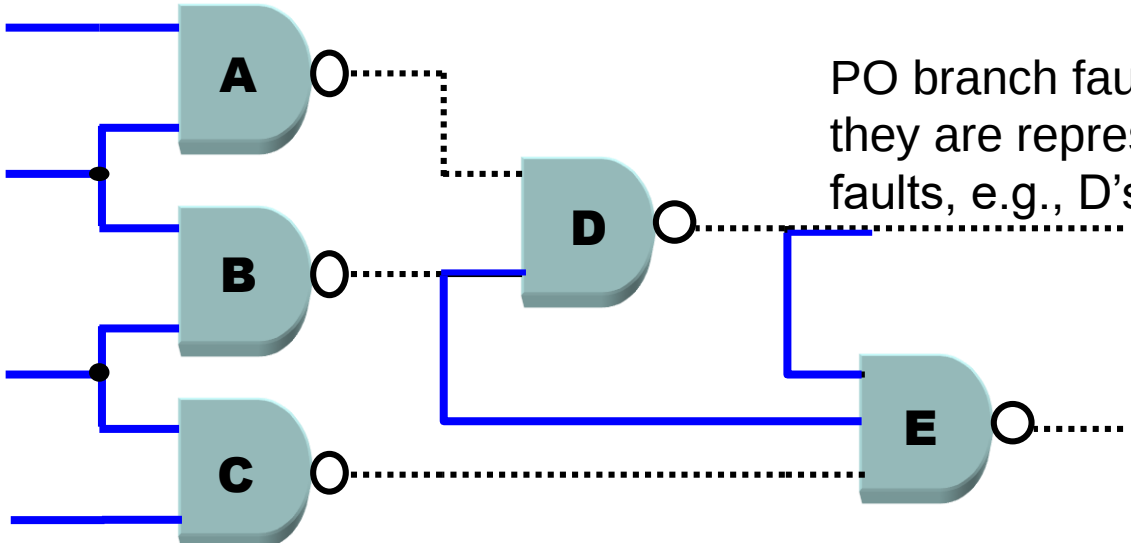


The faults marked by X are checkpoint faults.

Why Inputs + Branches Are Enough ?



- Sweeping the circuit from PO to PI to examine every gate and replace output faults by input faults until a PI or branch is met.
  - Based on an reversed levelized order: E→D→A→B→C
  - If a branch is met, start the above process from the stem again.
- In this example, checkpoints are marked in solid blue.



PO branch faults are ignored, since they are represented by gate output faults, e.g., D's output

An Example of Fault Collapsing + Checkpoint

- 10 checkpoint faults
  - a s-a-0 == d s-a-0, c s-a-0 == e s-a-0
    b s-a-0 > d s-a-0, b s-a-1 > d s-a-1
  - Note that the branch dominance is not obvious.
  - 6 out of 10 faults are enough.