CIS 4930. Digital System Testing
Fault Modeling

Dr. Hao Zheng
Comp. Sci & Eng
U of South Florida

Overview

- → Logic Fault Models
- → Fault Detection & Redundancy
- → Fault Equivalence & Fault Location
- → Fault Dominance
- → Single Stuck-Fault (SSF) Model
- → Multiple Stuck-Fault (MSF) Model
- → Summary

Recap: Testing Big Picture

- → A circuit defect leads to a fault.
- → A fault can cause a circuit error.
- → A circuit error can result in a circuit failure.
- → Testing a circuit:
  - → Apply test vectors to the circuit inputs.
  - → Compare circuit output responses to correct ones.
- → Exhaustive testing
  - → 2^n test vectors required for a *n*-input comb. circuit.
  - → Difficult for comb. circuit, impossible for seq. circuit.

Recap: Testing Big Picture

- Goal: find a small set of test vectors that target specific faults.
  - Ideally, no redundant test vectors for the same fault.
  - The set contains enough test vectors to uncover all target faults.
- Impossible to achieve 100% fault coverage.
  - Due to undetectable faults.

Recap: Physical Faults

Recall that we have 4 types of errors

- Design Errors
- Fabrication Errors
- Fabrication Defects
- Physical Failures

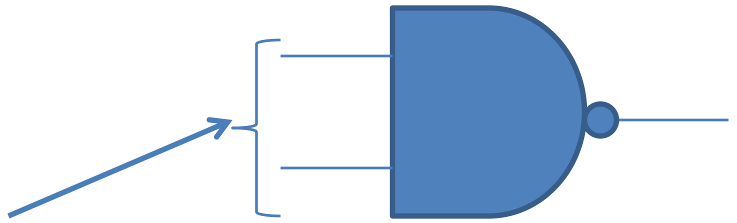
$$\left.\begin{array}{l}\text{Fabrication Errors}\\ \text{Fabrication Defects}\\ \text{Physical Failures}\end{array}\right\}\ \textbf{Physical Faults}$$

Physical Faults can be:

- Permanent
- Intermittent
- Transient

Recap: Logical Faults

- Physical faults are difficult to handle mathematically
- Therefore, for a physical fault we identify equivalent logical fault
  - Example:



Physical fault
- Inputs are shorted

$\Rightarrow$

Logical fault
- Gate now behaves as an inverter

4.1 Logic Fault Models

- Fault model: representation of physical faults and their natures at logic level.
- Recall that

  Behavior = Function + Timing
- Therefore, we can talk about two types of faults:
  - Logical Faults – modify circuit logic function.
  - Delay Faults - modify circuit operating speed.
- Our focus will be on Logical Faults

Logical Fault Modeling - Advantages

- → Fault analysis becomes a logical problem.
  - → Test can start before silicon is available.
- → Fault analysis become less complex.
  - → *Many physical faults can be modeled by the same logical fault*
- → Technology independent
- → Tests derived for logical faults may be used for physical faults whose effect
  - → not completely understood
  - → or too complex to analyze

Structural and Functional Faults

- Structural faults
  - Faults defined on a structural circuit model.
  - Effect: modify interconnections
- Functional faults
  - Faults defines on a functional circuit model
  - Effect: modify truth table etc.
- Intermittent & Permanent Faults
  - Statistical data on probability of occurrence of transient/intermittent faults are difficult to have.
  - Our focus in this discussion is on structural and permanent faults

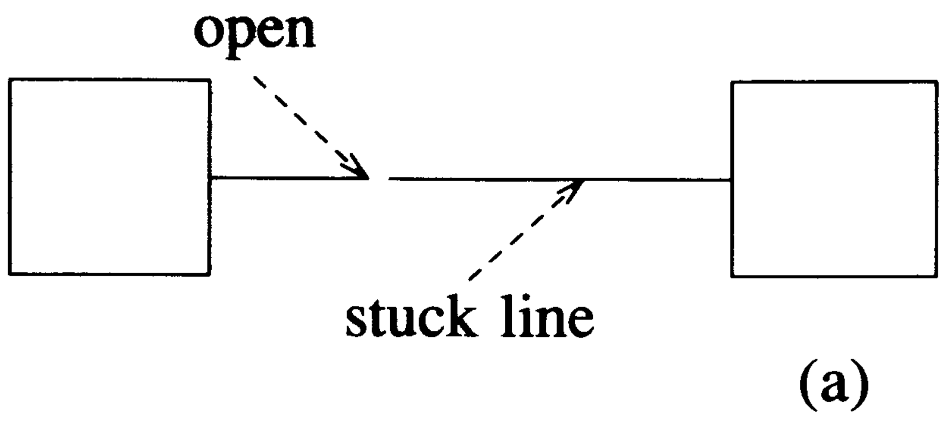
Single Fault Assumption - Justification

- Assumption – one logical fault in the system.
- Justification
  - Frequent testing strategy (test often so that prob. of multiple faults developing in between too low)
  - Usually tests derived for individual single faults are applicable for detecting multiple faults composed of the single ones.

Structural Fault Models

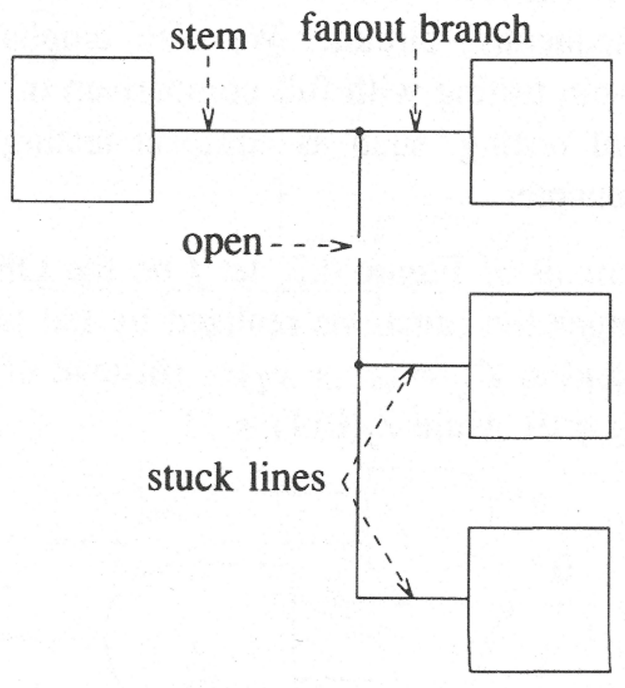
- Assumptions:
  - Components are fault free and,
  - Only interconnections are affected – shorts & opens.
- **Stuck-at-v Fault:**
  - ***Short*** (with supply/gnd) or **Open** lines behave as **“stuck at”** fixed logic value *v* (*v* ∈ {0,1})
- **Bridging fault:**
  - Short between two lines $\rightarrow$ usually new logic function (AND or OR bridging)
  - We will discuss bridging faults later

Single Stuck Fault – Open Line



- → Open on an unidirectional line
- → Unconnected input assumes a constant logic value
- → Single logical fault, *i-stuck-at-a* can represent
  - → Line *i* open
  - → Line *i* shorted to Vdd or GND (a=1 or a=0)
  - → Internal fault in component driving line *i*

Multiple Stuck Fault

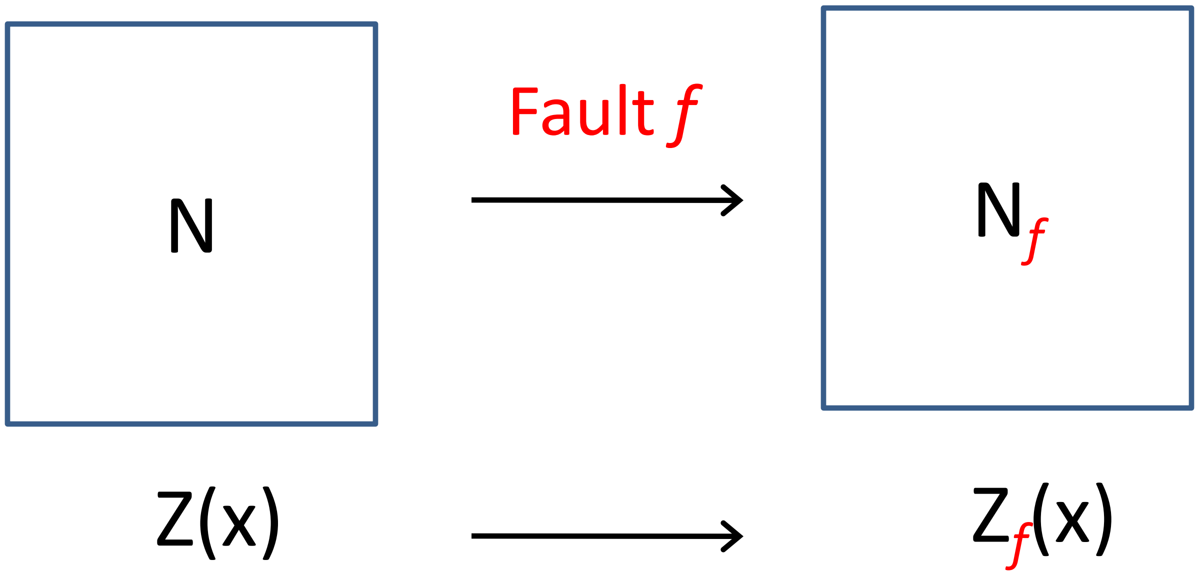


(b)

- Single open can result in multiple faults
- Under single fault model, we need to consider faults on all fanout branches separately.

4.2 Fault Detection and Redundancy

Fault Detection – Combinational Circuits



- Input vector *t* to *N* results in output response *Z(t).*
- Test T = $<t_1, t_2 \ldots t_m>$ will yield $< Z(t_1), Z(t_2), \ldots Z(t_m)>$

Fault Detection – Comb Circuits

Definition 4.1 A test (vector) *t* **detects** a fault *f* iff

$$z_f(t) \neq z(t)$$

- → Note the above is applicable to comb. circuits only.
- → Test vectors in *T* can applied in any order, so T is *set of tests*
- → Applicable to *edge-pin testing*
  - → Components are assumed to be fault free.

Example 1

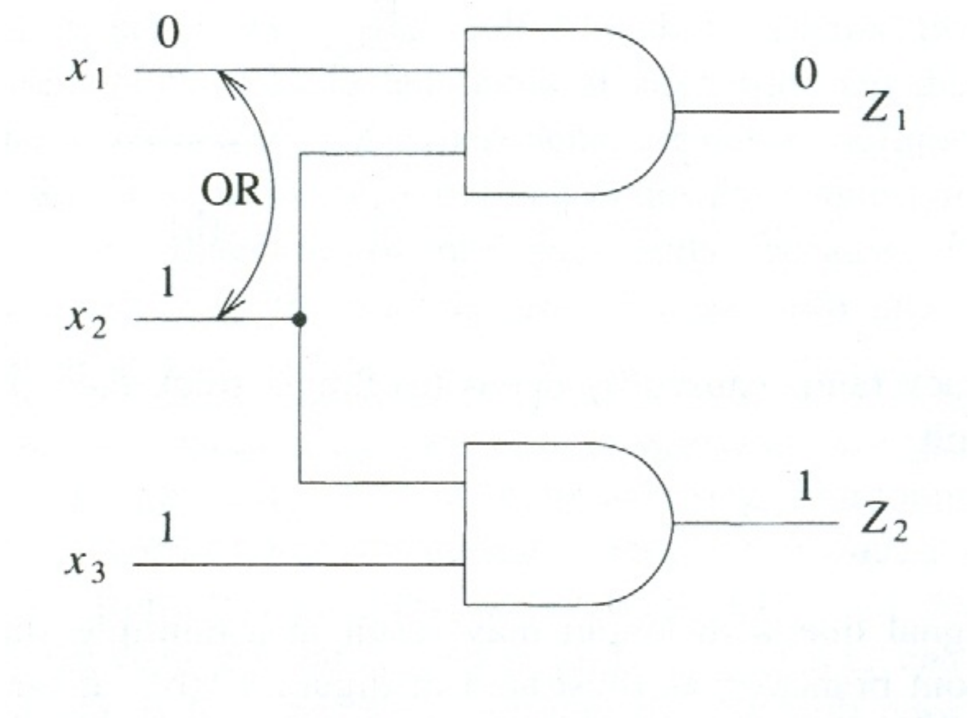


Figure 4.2

Let *f* = OR-bridging fault between x1 and x2

Z_1 =_______________ Z_{1f} =_______________

Z_2 =_______________ Z_{2f} =_______________

Give a test vector that detects the fault:

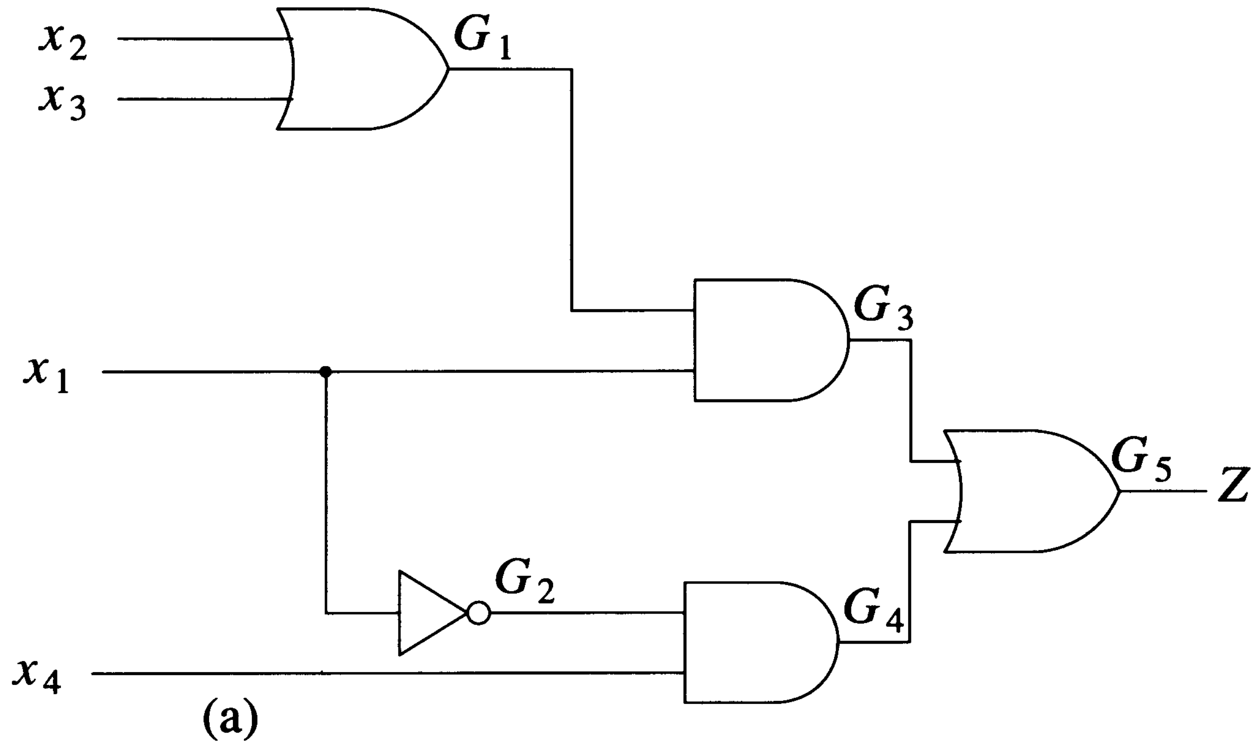
<x1, x2, x3> = _____________

Fault Detection and Test Vectors

- For a single-output circuit, a test *t* that detects a fault *f* makes
  - $Z(t) = 0$ and $Z_f(t) = 1$ or
  - $Z(t) = 1$ and $Z_f(t) = 0$
- Thus, the set of all tests that detect *f* is given by

$$Z_f(t) \oplus Z(t) = 1$$

Example 2

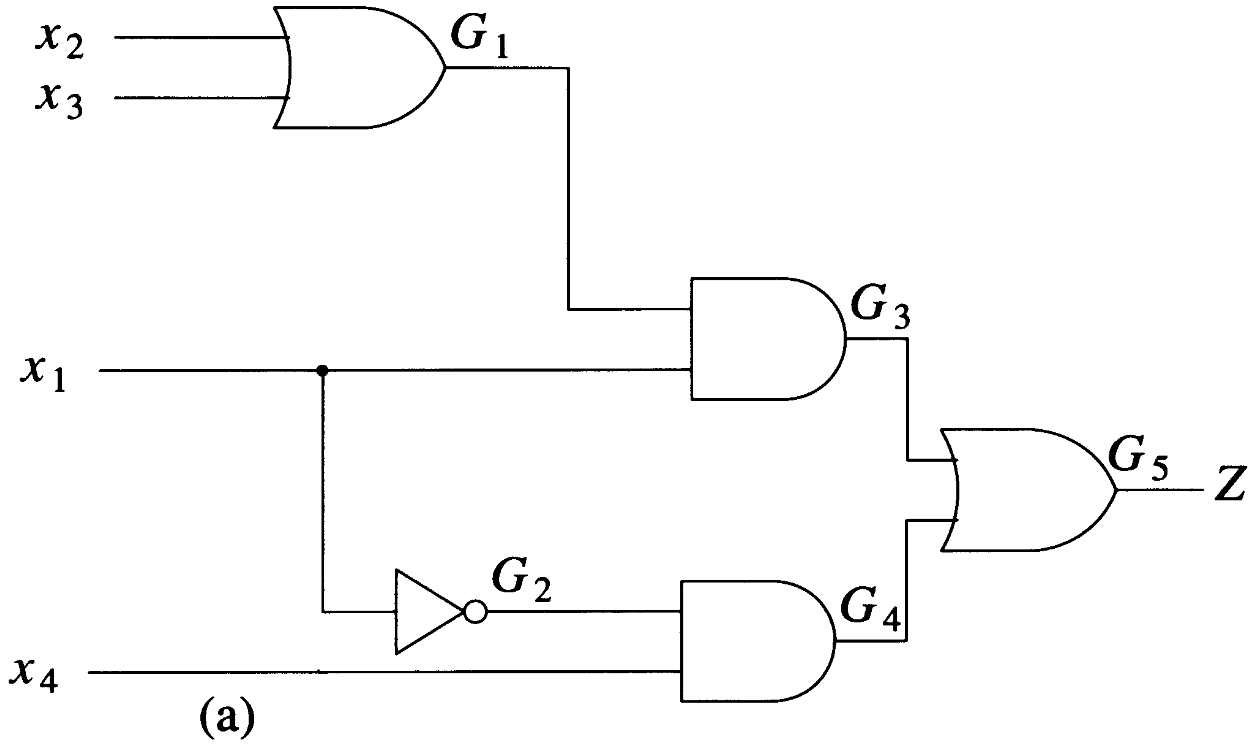


Let the fault *f* be x_4 s-a-0. Find all test vectors that detect *f*

Example 3

Let *f* = x_4 s-a-0.

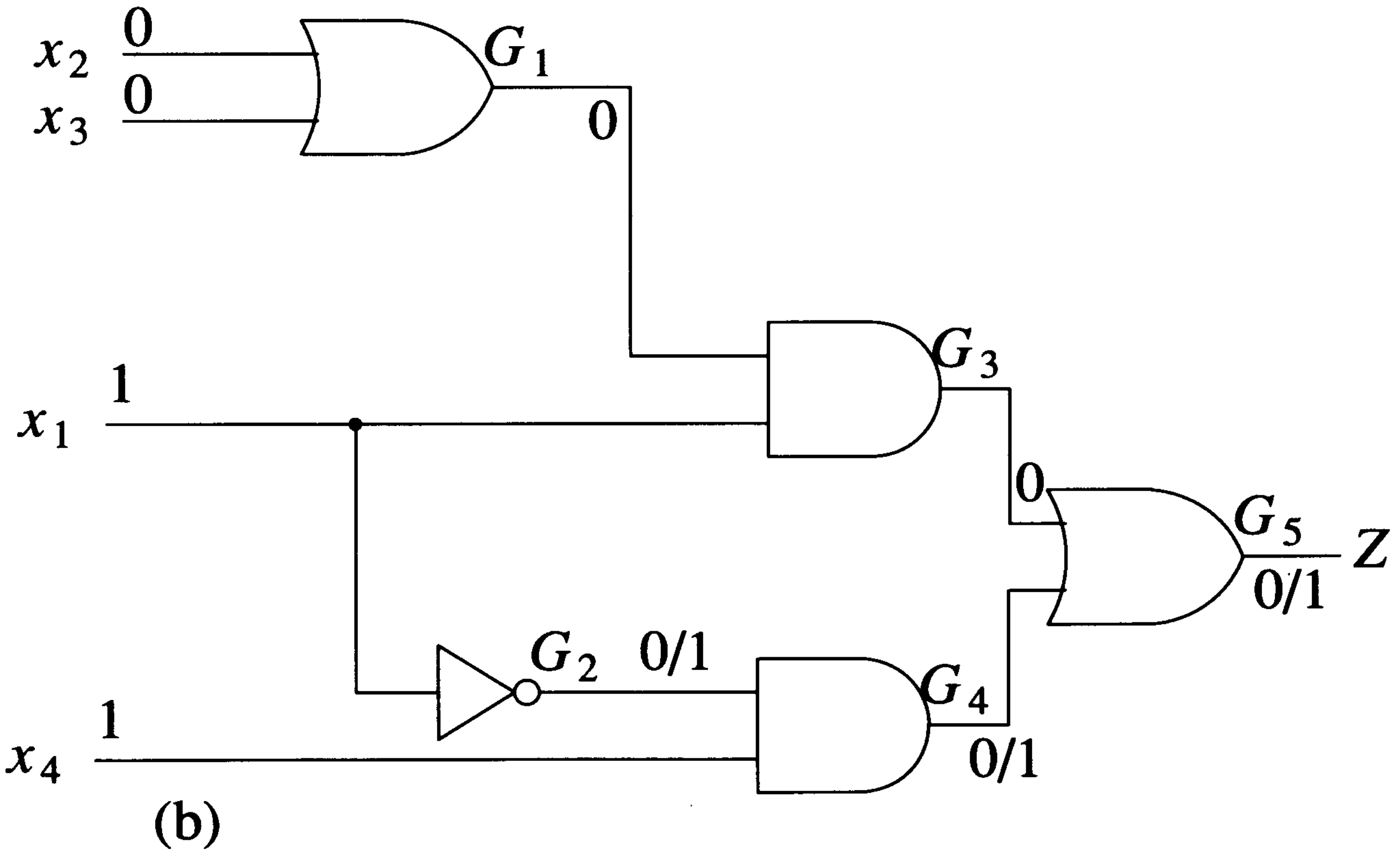
For test vector 1001 that detects *f*, simulate without and with fault *f*



Example 3

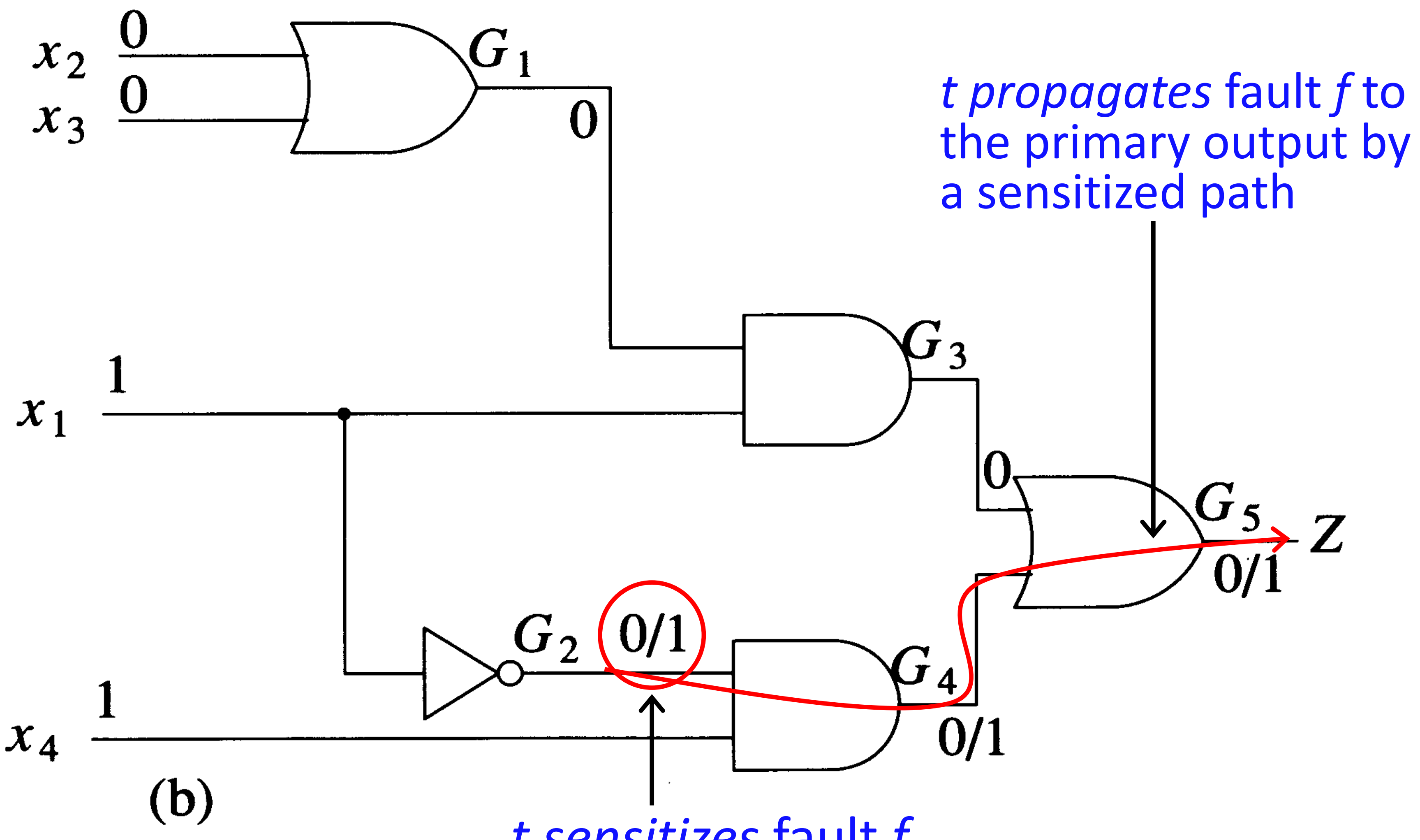
v/v_f: fault-free/faulty

Fault: G2 s-a-1



Fault – Sensitization

f = G2 s-a-1, Test vector *t* = 1 0 0 1



t propagates fault *f* to the primary output by a sensitized path

t sensitizes fault *f*

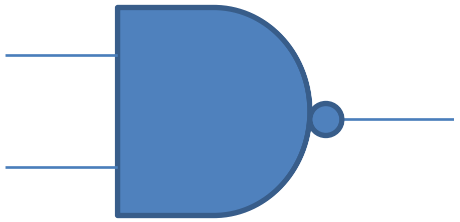
Fault Sensitization – Terminology

- **Fault Activation**: A test *t* activates a fault on a line if it generates an error at the site of the fault.
- **Fault Propagation:** A test *t* propagates the error to a primary output by creating at least one path from fault site to the primary output.
- **Line Sensitization:** A line whose value under the test *t* changes in the presence of the fault *f* is said to *be sensitized to the fault f by the test t*
- **Sensitized Path:** A path composed of sensitive lines

Gate Controlling and Enabling Values

- **Controlling value *c*:** If at least one input assumes value *c*, then the gate's output assumes value $c \oplus i$.
- **Enabling value $\bar{c}$:** If all inputs of a gate have the enabling value, then the gate's output assumes the value $\bar{c} \oplus i$.

Example



Control value =_____

Enabling value = ____

NAND: c= 0, i=1.

Lemma 4.1

Let G be a gate with inversion i and controlling value *c*, whose output is sensitized to a fault *f* (by a test *t*).

1. All inputs of G sensitized to *f* have the same value (say, *a*).
2. All inputs of G not sensitized to *f* (if any) have value $\bar{c}$.
3. The output of G has value $a \oplus i$.

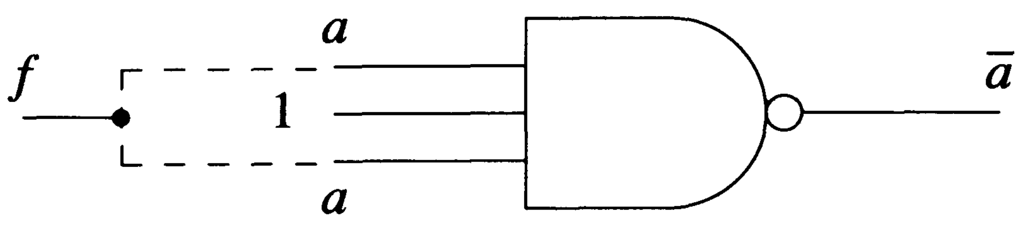


Figure 4.4 NAND gate satisfying Lemma 4.1 (c=0, i=1)

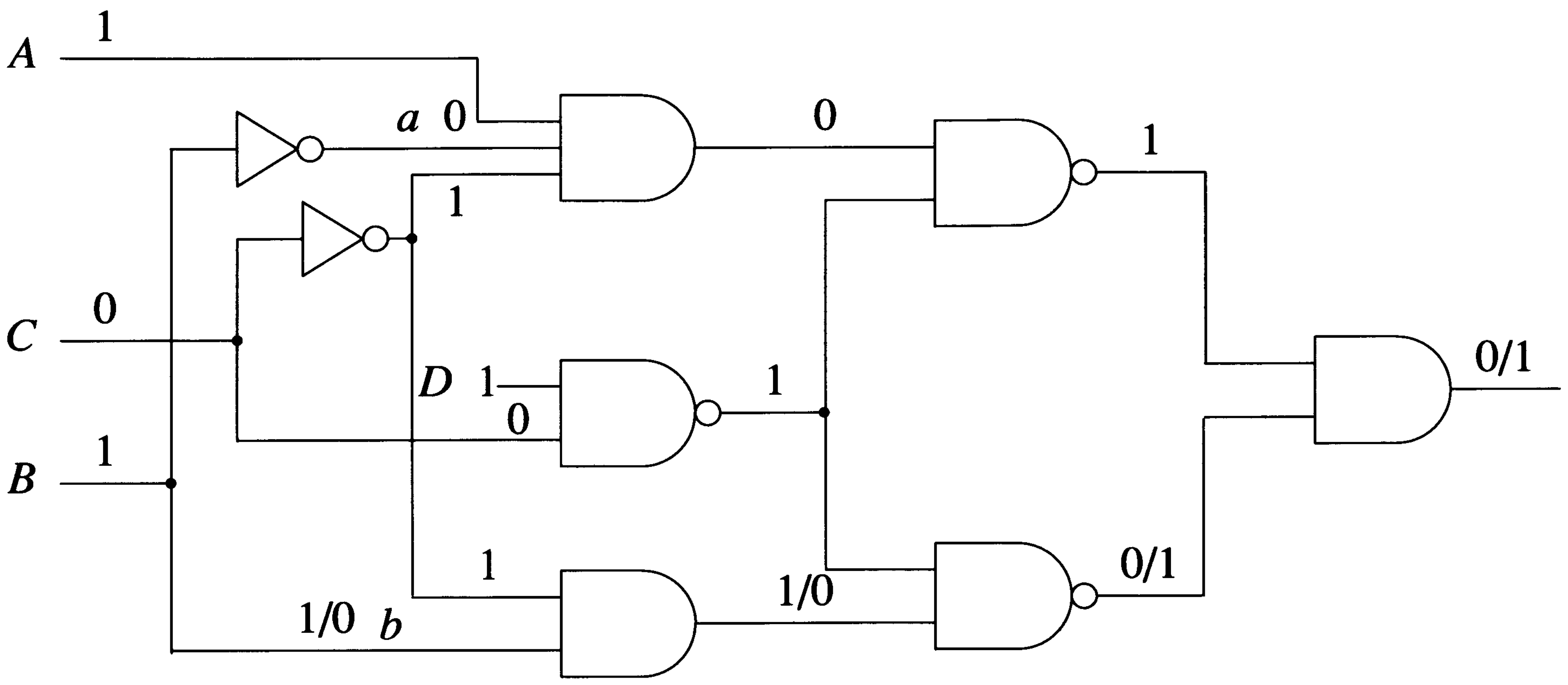
Faults – Detectability

A fault *f* is said to be **detectable** if there exists a test *t* that detects *f*

Otherwise, *f* is **undetectable.**

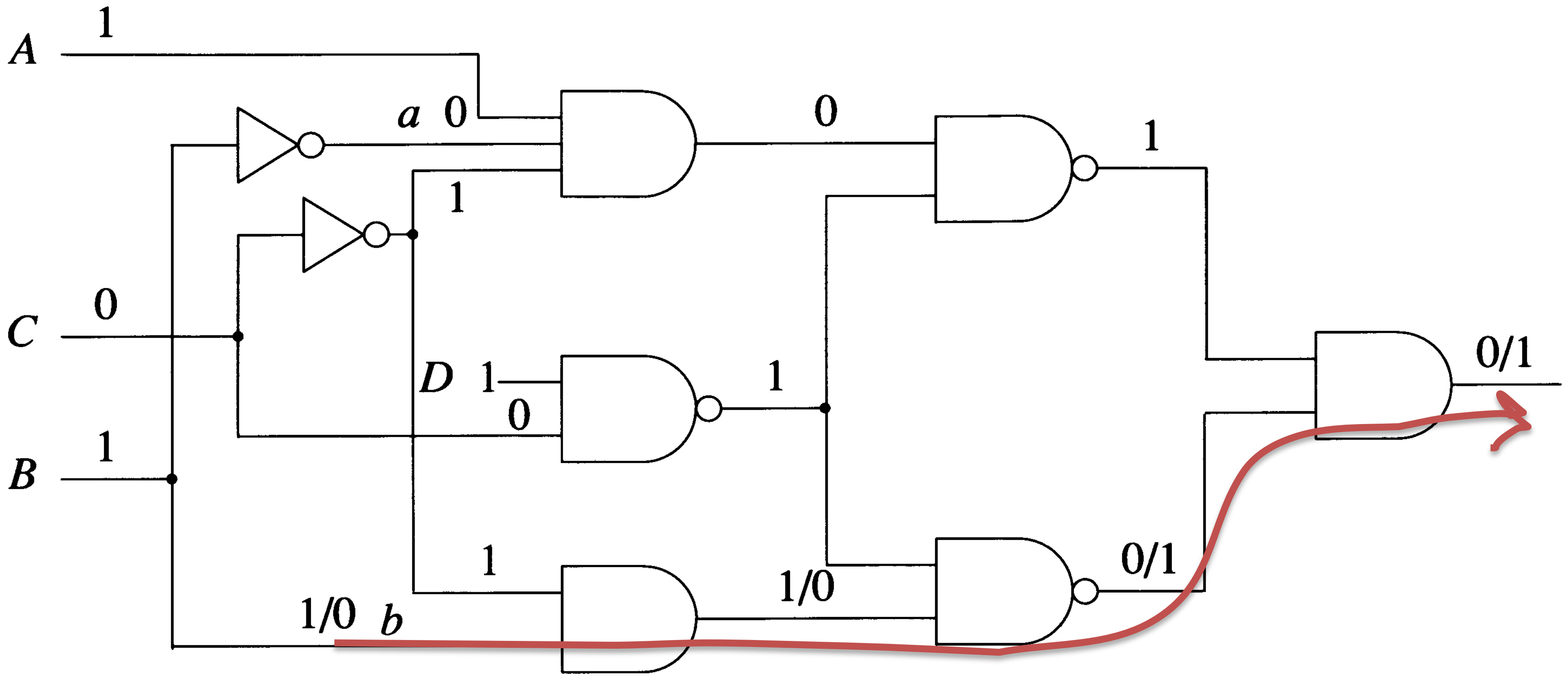
For undetectable fault, no test exists that can simultaneously activate and propagate the fault to the primary output.

Example: Undetectable Fault *a s-a-1*



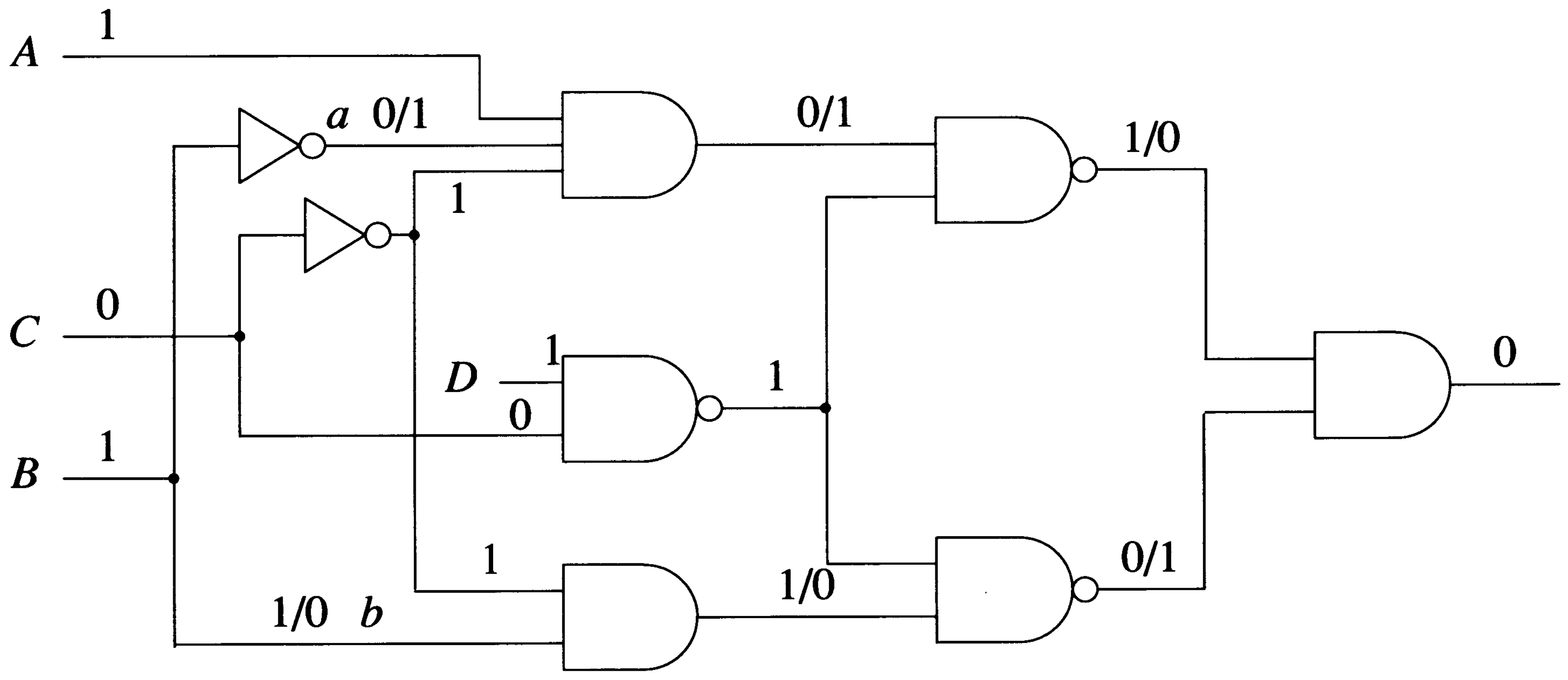
Example: Detectable Fault *b s-a-0*

b s-a-0 is detectable with t = 1101



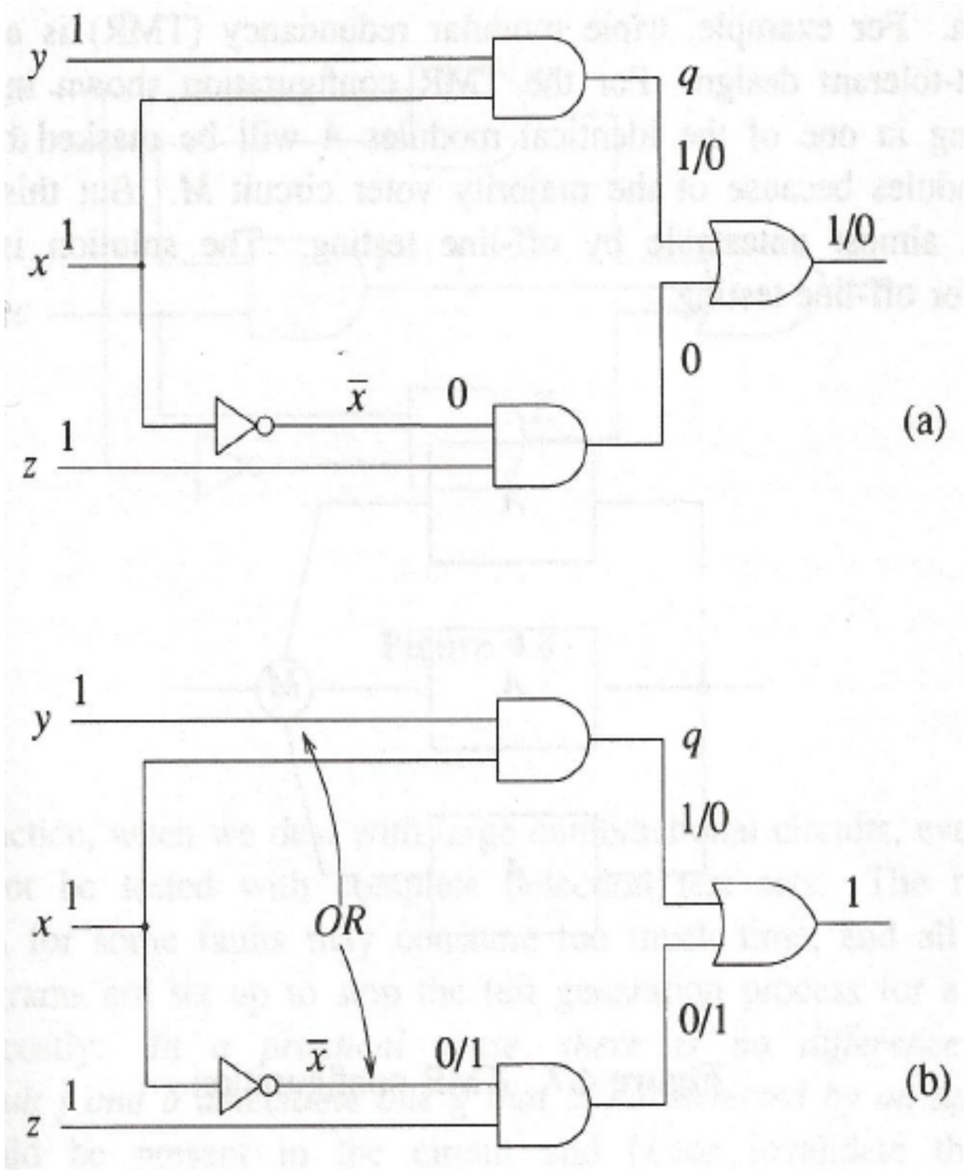
Example: Undetectable Fault

b s-a-0 becomes undetectable *in the presence of a s-a-1* by test *t=1101*



Example 2: Undetectable Fault

q s-a-0 is detectable with
t = 111

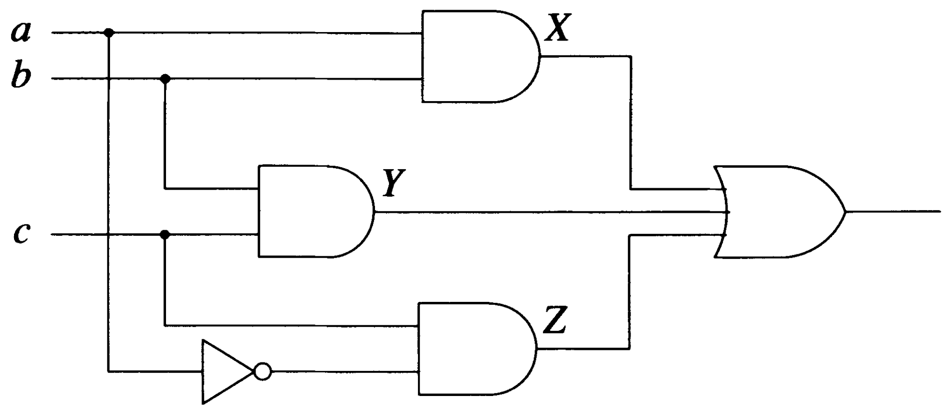


q s-a-0 is undetectable in the presence of OR Bridging fault between y and $\bar{x}$.

Fault – Redundancy

- A combinational circuit that contains an undetectable stuck fault is said to be **redundant**
- Such circuit can always be reduced by eliminating a gate or a gate input
- A combinational circuit in which all stuck faults are detectable is said to **irredundant**

Example: Y s-a-0 is undetectable. Gate Y can be dropped.



Fault Interaction

- → If *f* is a detectable fault and *g* is an undetectable fault, then *f* may become undetectable in the presence of *g*. Such a fault *f* is called a *second-generation redundant fault.*

- → Two undetectable single faults *f* and *g* may become detectable if simultaneously present in the circuit. In other words, the multiple fault {*f* , *g*} may be detectable even if its single-fault components are not.

Detecting Redundancy

- → To show a line is redundant => to prove that no test exists for the corresponding fault
- → Detecting Redundancy Problem => Test Generation Problem
- → Test generation problem is an *NP*-complete problem
- → Practical test generation algorithms run in polynomial time
- → Redundant faults make test generation algorithms exhibit worst-case behavior

Large Combinational Circuits

- Even if the circuit is *irredundant,* we may not have complete test set due to time limitations
- In such a case, the fault (say *f*) for which no test exists, is no different from an *undetectable fault* (say *g*)
- Undetectable fault *g* may be present in the circuit and invalidate the single fault assumption.

Sequential Circuits

- Testing more difficult than combinational circuits
- Need a test sequence
- Response is a function of initial state
    - Let *T* be a test sequence – a sequence of test vectors.
    - $R(q, T)$ be the response to *T* with initial state *q*
    - $R_f(q_f, T)$ be the response for faulty circuit

T Strongly Detects *f*

- Definition 4.2: A test sequence *T* **strongly detects** the fault *f*

  if and only if

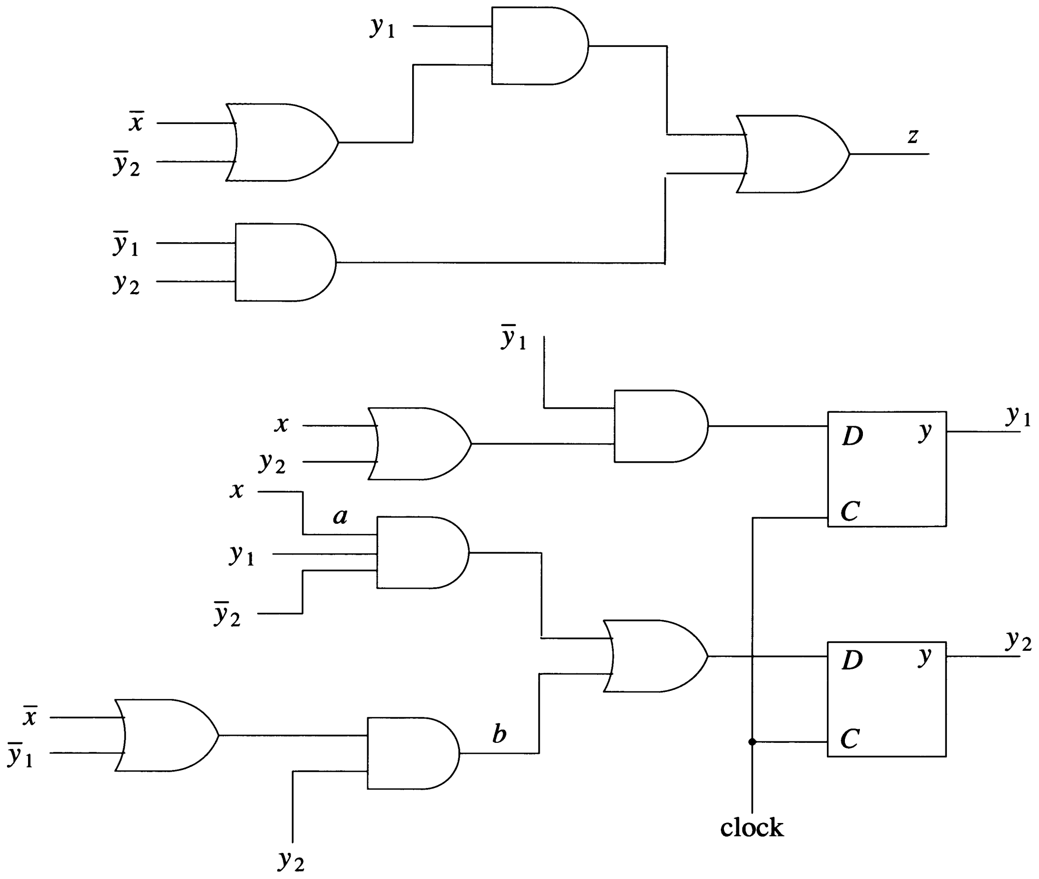
  the output sequences $R(q, T) \neq R_f(q_f, T)$ for every possible pair of initial states q and q_f

Example

α line *a s-a-1*

β *line b s-a-0*

Test sequence T=10111



	$x=0$	$x=1$	y_1	y_2
A	*A*,0	*D*,0	0	0
B	*C*,1	*C*,1	0	1
C	*B*,1	*A*,0	1	1
D	*A*,1	*B*,1	1	0

Initial state	Output sequence		
	Fault-free	α (*a s-a*-1)	β (*b s-a*-0)
A	01011	01010	01101
B	11100	11100	11101
C	00011	00010	01010
D	11001	10010	11010

Figure 4.9 Output sequences as a function of initial state and fault

Example

- T does not strongly detect α
- T strongly detects β

	0	1	y_1	y_2
A	*A*,0	*D*,0	0	0
B	*C*,1	*C*,1	0	1
C	*B*,1	*A*,0	1	1
D	*A*,1	*B*,1	1	0

: *10111*

Initial state	Output sequence		
	Fault-free	α (*a s-a*-1)	β (*b s-a*-0)
A	01011	01010	01101
B	11100	11100	11101
C	00011	00010	01010
D	11001	10010	11010

T Detects *f*

- Definition 4.3: A test sequence *T* ***detects*** the fault *f*

  if and only if

  for every possible pair of initial states q and q_f the output sequences $R(q, T) \neq R_f(q_f, T)$ for *some* specified vector $t_i \in T$

Testing with Initialization

- Phase I: Initialization sequence T_I such that N and N_f are brought to known states q_I and q_{If}
  - Output responses ignored during initialization
- Phase II: Apply T' (output responses are predictable)
  - t_i is first vector of T' for which an error is observed

Drawback

- Initialization may not be possible for faulty circuit
- Example:

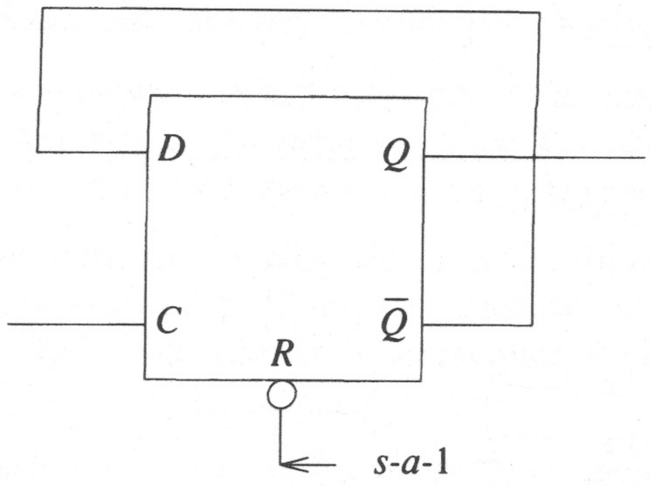


Figure 4.10 Example of a fault preventing initialization

4.3 Fault Equivalence and Fault Location

Fault Equivalence – Combinational Circuits

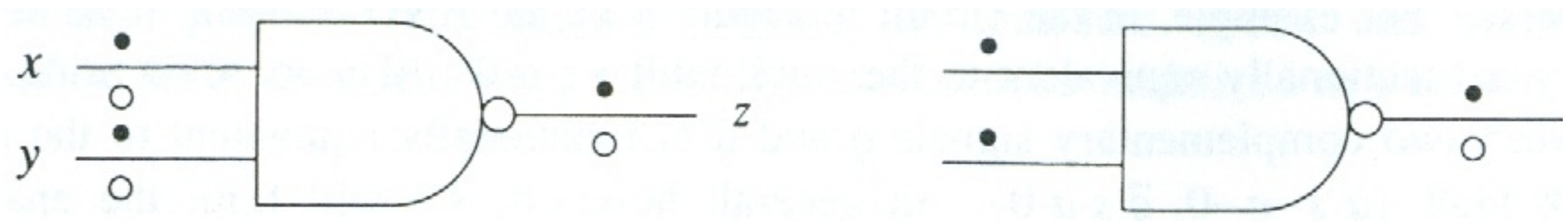
- **Definition 4.4**: Two faults *f* and *g* are said to be **functionally equivalent** iff

$$Z_f(x) = Z_g(x)$$

  - A test *t* is said to **distinguish** between two faults *f* and *g* if $Z_f(t) \neq Z_g(t)$
  - There exists no test that can distinguish functionally equivalent faults
  - Faults are divided into equivalent classes.

Equivalence Fault Collapsing

- Reduce faults into equivalent classes
- For a NAND gate,
  - All input *s-a-0* faults and the output *s-a-1* are functionally equivalent



(a) (b)

$\bullet : s-a-1, \quad \circ : s-a-0$

Equivalence Fault Collapsing

- In general, for a gate with controlling value *c* and inversion *i*,
  - All input s-a-c faults are *functionally equivalent* to output s-a-($c \oplus i$) faults
  - Reduce 2(n+1) faults to n+2 faults for a *n*-input NAND gate.

Fault Location

- Goal of testing is to locate the fault besides detecting the fault
- A *complete location test* distinguishes between every pair of distinguishable faults in a circuit
- A fault-free circuit contains *empty fault,* denoted by Φ.
  - Therefore $Z_{\Phi}(x) = Z(x)$
- A fault detection is a particular case of fault location, since a test that detects *f* distinguishes between *f* and Φ.

Functional Equivalence Under a Test

- In practice, test sets are not complete
  - Affect diagnostic resultion
- **Definition 4.5**: Two *f* and *g* are **functionally equivalent under a test set T,**

iff

$Z_f(t) = Z_g(t)$ for every test $t \in T$

Note:

Functional equivalence implies functional equivalence under any test set, but not vice versa.

Fault Equivalence – Sequential Circuits

- **Definition 4.6**: Two *f* and *g* are **strongly functionally equivalent** iff their corresponding state tables are equivalent.
  - Impractical

- **Definition 4.6**: Two *f* and *g* are **functionally equivalent** iff for any *T'*,

$$R_f(q_{If}, T') = Z_g(q_{Ig}, T')$$

4.4 Fault Dominance

Fault Dominance – Combinational Circuits

→ Another fault relation to reduce faults to be considered

→ **Definition 4.8**: Let T_g be the set of all tests that detect a fault *g*. We say that a fault *f* **dominates** the fault *g*

iff

f and *g* are functionally equivalent under *Tg*.

Fault Dominance

- If *f* dominates *g,* then any test *t* that detects *g, i.e,* $Z_g(t) \neq Z(t)$, will also detect *f* (on the same primary inputs) because $Z_f(t) = Z_g(t)$
- Therefore, we can drop *f* from the fault set

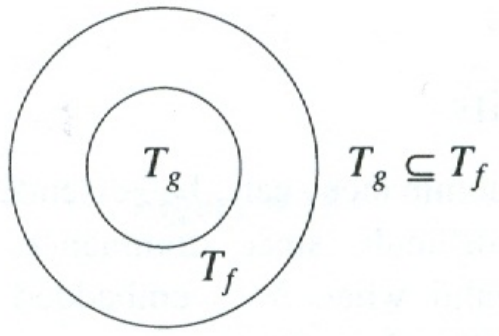
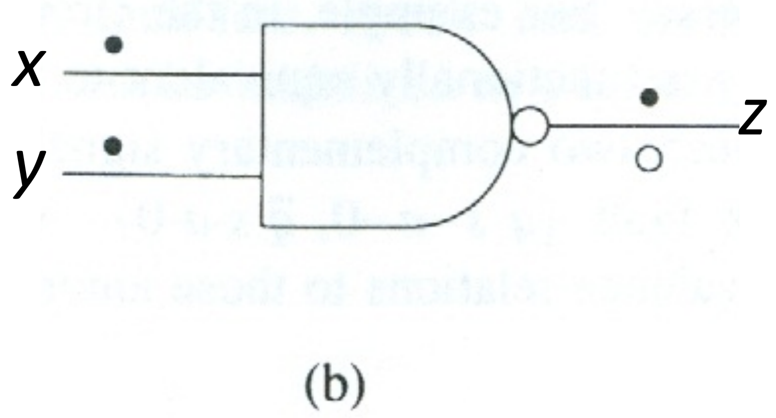
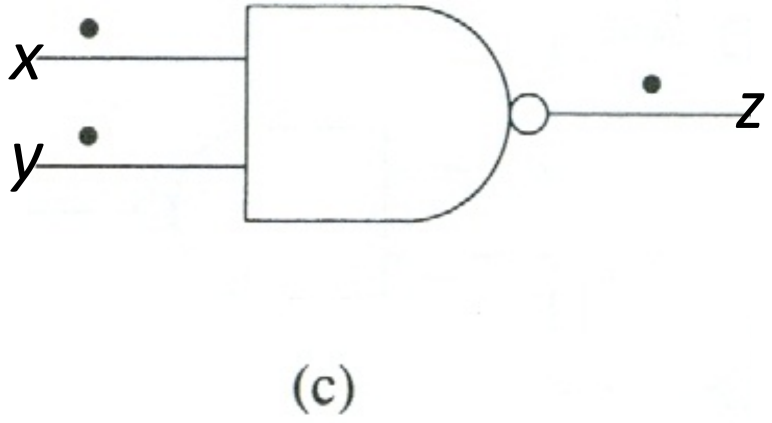


Figure 4.13 The sets T_f and T_g when *f* dominates *g*

Example





$\bullet : s - a - 1, \quad \circ : s - a - 0$

Fault g : y s-a-1

Fault f : z s-a-0

f dominates g

$T_g = \{xy = 10\}$

$T_f = \{???\}$

Dominant Fault Collapsing

- In general, for a gate with controlling value *c* and inversion *i*,
  - Output *s-a-(c⊕i)* fault dominates input $s\text{-}a\text{-}\bar{c}$ faults.

- Better to choose a fault model dominated by other models
  - Tests detecting one model also detects other faults models

An Example

- Faults: *f : z2 s-a-0, g : y1 s-a-1,* test *t= 10*
  - *t* detects both *f* and *g*, but they do not dominate each other

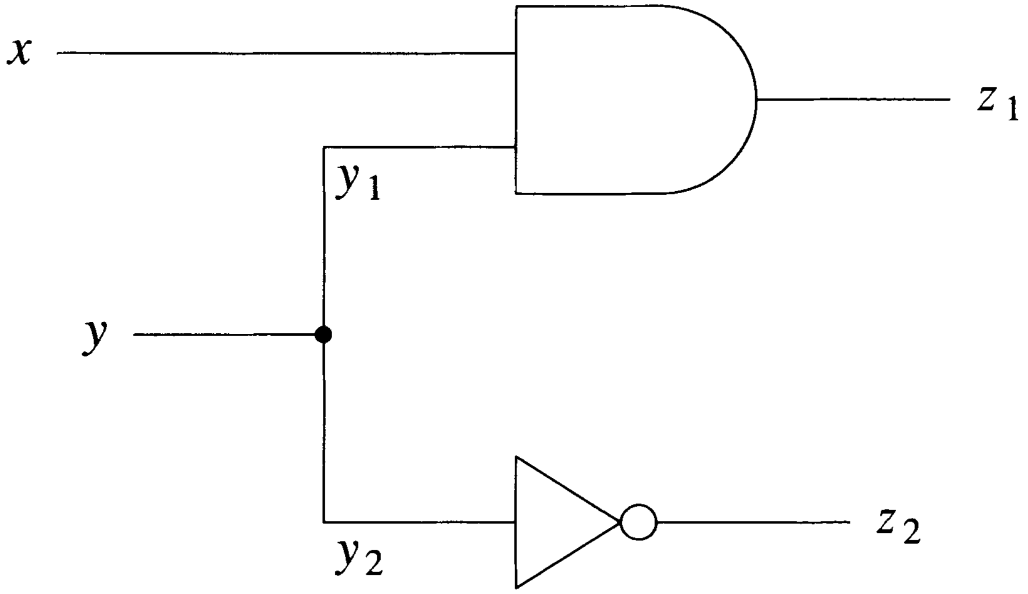


Figure 4.14

4.5 The Single Stuck-Fault Model

Fault Universe

- For *n* lines where SSFs can be defined, there are *2n* stuck faults.
- For fanout-free circuits, the number of faults is 2(G+I).
  - G: gate count
  - I: number of primary inputs
- For circuits with fanout, the estimate is $2Gf$
  - *f*: average fanout count

Checking Functional Equivalence is Hard!

- In general, to determine whether two arbitrary faults are functionally equivalent is NP-complete
- Example: Are *c s-a-1* and *d s-a-1* functionally equivalent?

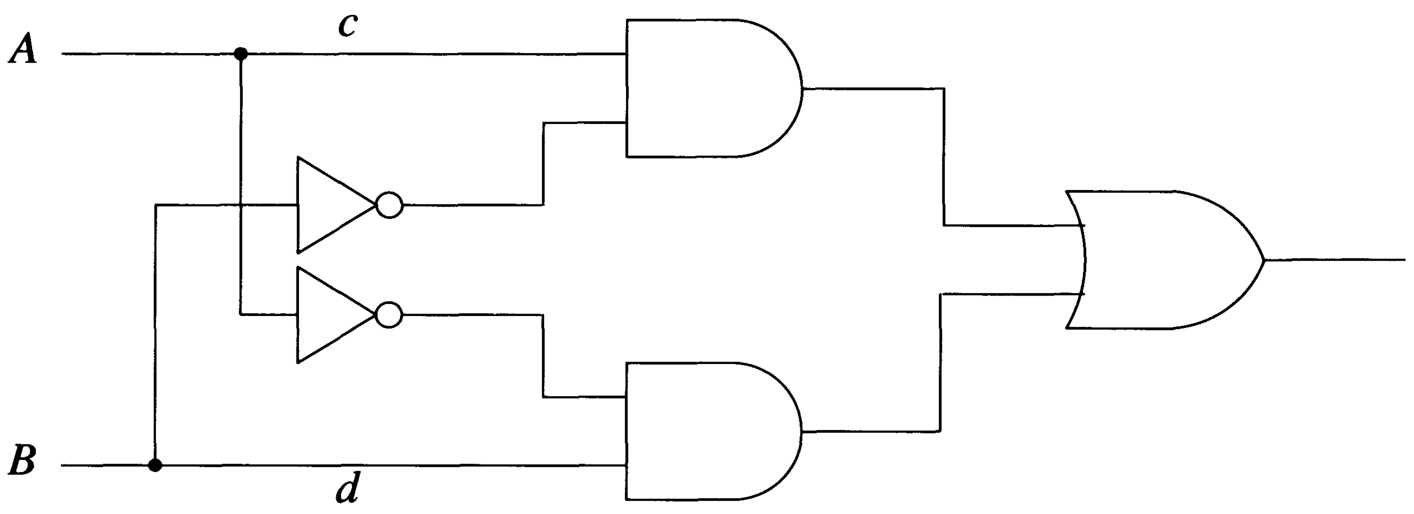


Figure 4.17

Structural Equivalence

- In a circuit N_f the fault f creates a set of lines with constant values.
- By removing all these lines (except POs), we obtain $S(N_f)$
- Two faults f and g are structurally equivalent if $S(N_f)$ and $S(N_g)$ are identical
- All structurally equivalent faults are functionally equivalent
- But not all functionally equivalent faults are structurally equivalent

Example: Structural Equivalence

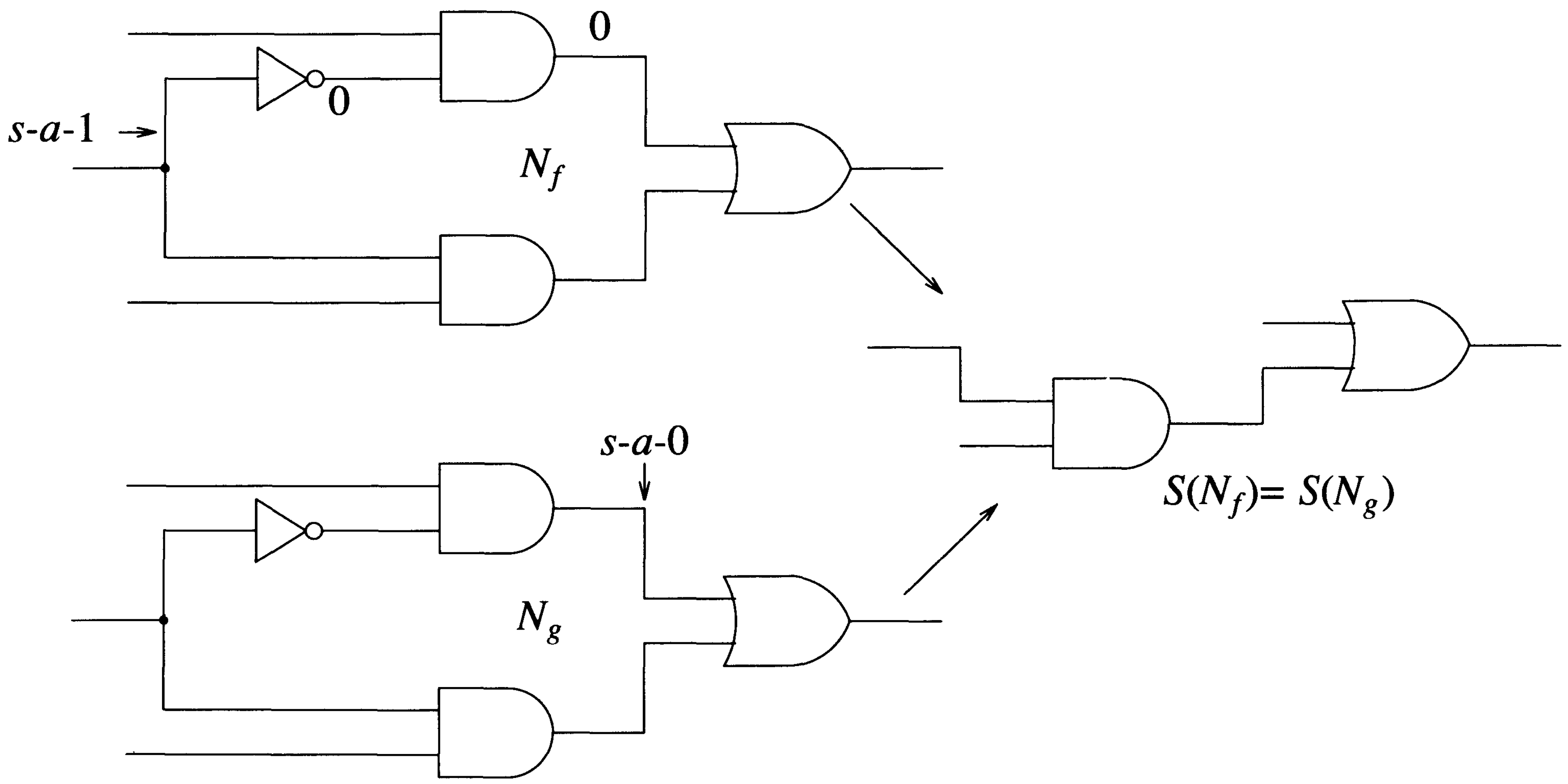


Figure 4.18 Illustration for structural fault equivalence

Structurally Equivalent Fault Collapsing

- For a line with fanout of 1, the faults at its sources are structurally eq. to those at its destination.
- For a gate, input $s\text{-}a\text{-}c$ is structurally eq. to output $s\text{-}a\text{-}(c \oplus i)$.
- For each eq. class, retain only one fault.

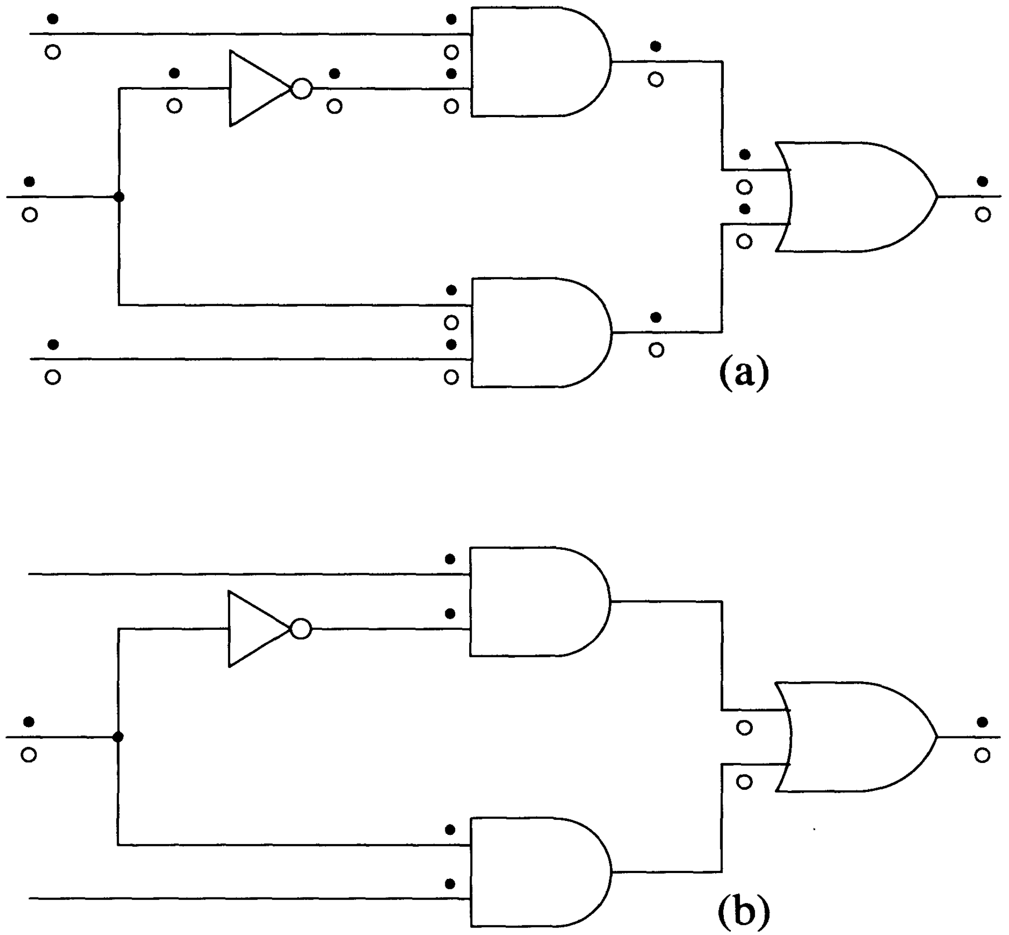
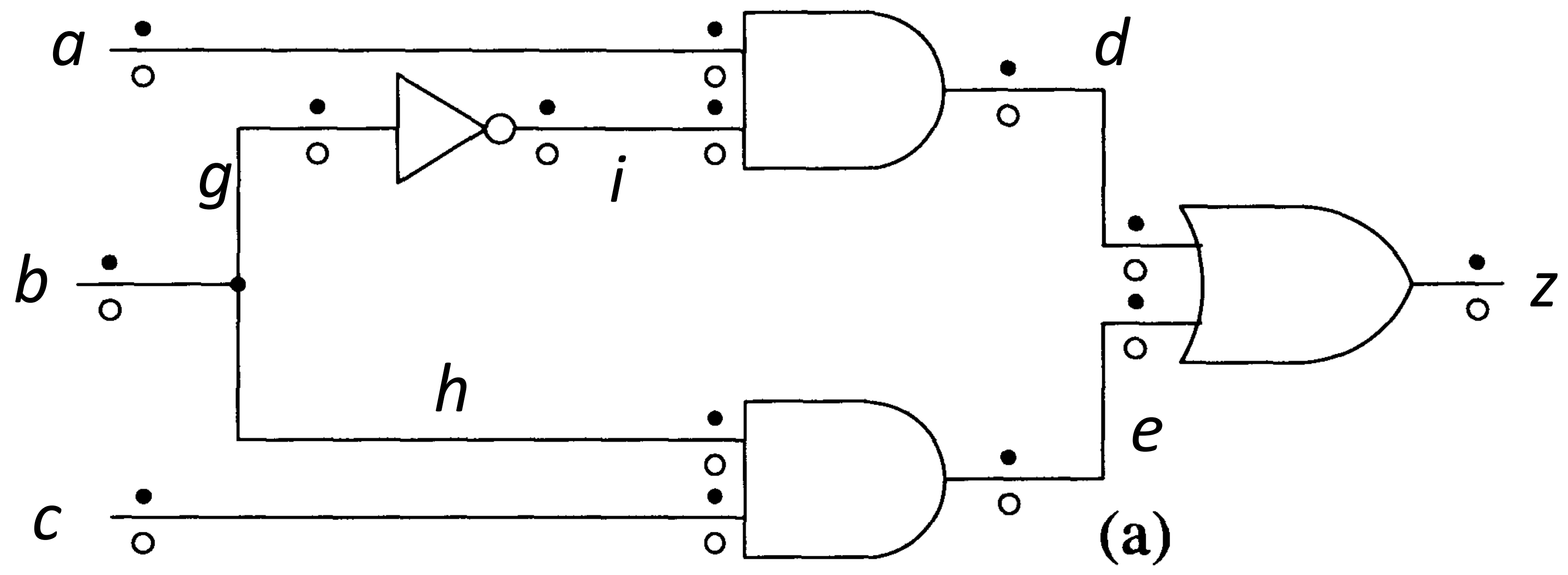


Figure 4.19 Example of structural equivalence fault collapsing

Structurally Equivalent Faults – Example



g s-a-1 and i s-a-0 ?

On average, about 50% SSFs can be reduced !

Fault Reduction by Dominance Relation

- **Theorem 4.1**: In a fanout-free combinational circuit *C*, any test set that detects all SSFs on the primary inputs of *C* detects all SSFs in *C*.
- **Theorem 4.2**: In a combinational circuit *C* any test set that detects all SSFs on the primary inputs and the fanout branches of *C* detects all SSFs in *C*.
  - These faults can be further reduced by structural equivalence and dominance relations.

Example

- How many faults after fault collapsing?
- Answer: 10 (verify this)

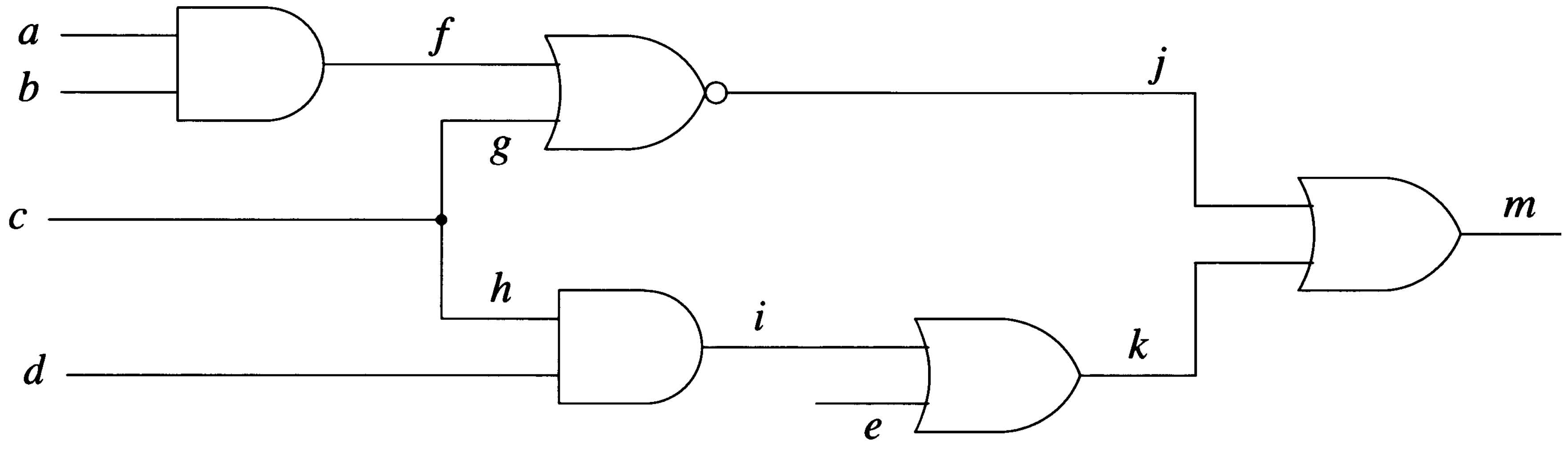


Figure 4.21

Stem and Fanout Faults – Example 1

- In general neither functional equivalence nor dominance relations exists between stem and fanout faults
- Stem fault *j s-a-0* is detected but *k s-a-0* and *m s-a-0* are not

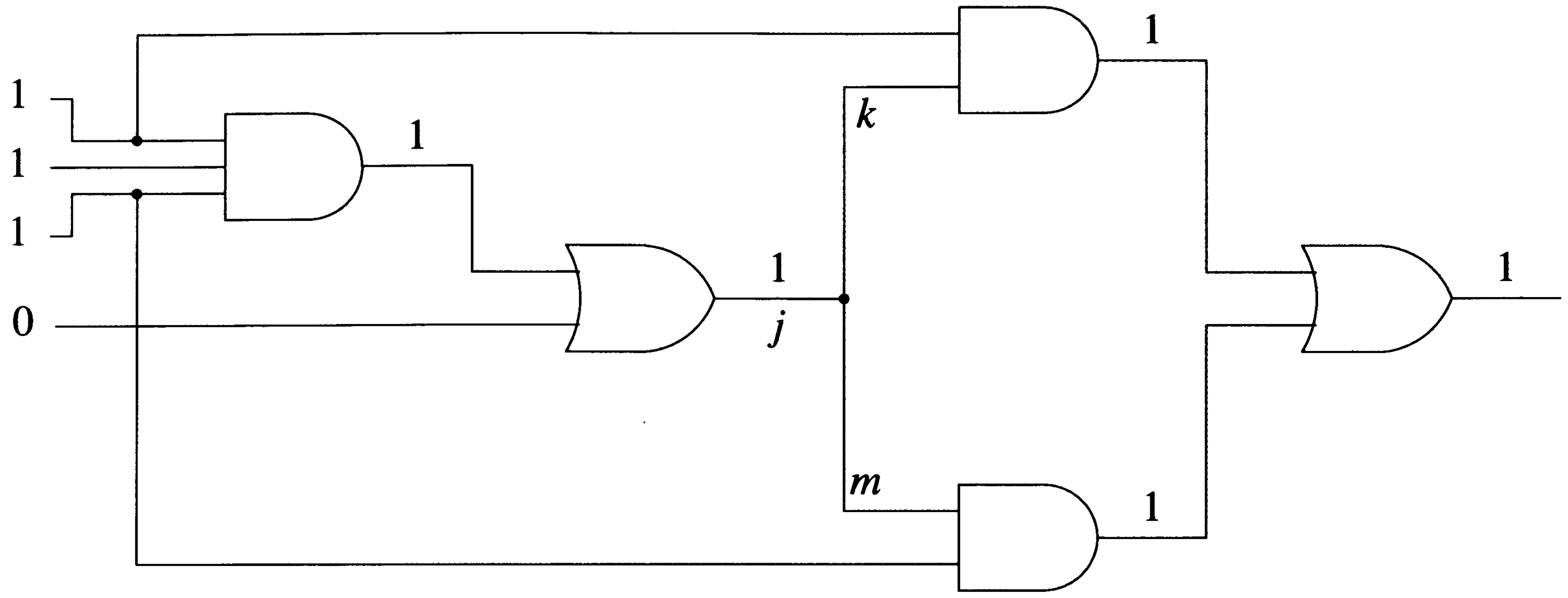


Figure 4.22

Stem and Fanout Faults – Example 2

→ Fanout faults x_1 *s-a-0* and x_2 *s-a-0* are detectable but stem fault *x s-a-0* is not

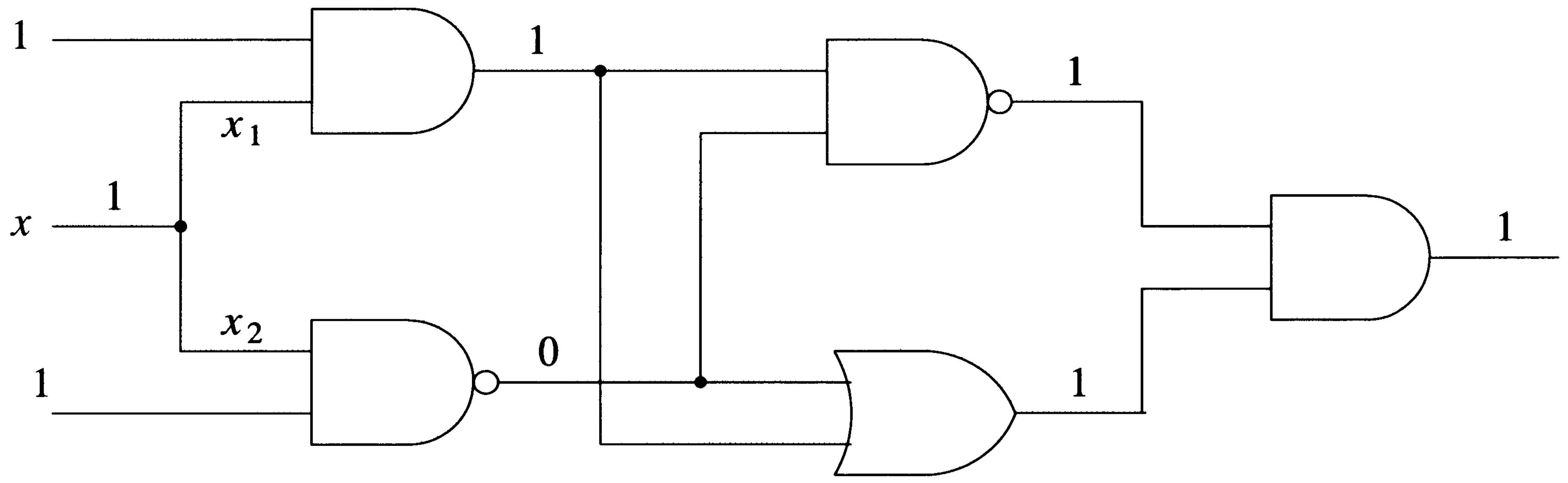


Figure 4.23

4.6 The Multiple Stuck-Fault Model

Multiple Stuck Fault (MSF) Model

- Several lines can be stuck simultaneously.
  - Straightforward extension of SSF
- For a *n* line circuit, there are
  - 2*n* SSFs, but
  - $3^n - 1$ possible MSFs
  - $\sum_{i=1}^{k} \binom{n}{i} 2^i$ possible MSFs for multiplicity *k*.
  - Too large a number to handle.
- A MSF *F* can be viewed as $\{f_1, f_2, \ldots, f_k\}$.
- Why do we need to consider MSF?
  - Because of masking relation between faults

Functional Masking

- **Definition 4.9**: Let T_g be the set of *all* tests that detect *g*. We say that *f* **functionally masks** *g* iff the multiple fault {*f, g*} is not detected by any test in T_g
- Example: *a s-a-1* masks *c s-a-0*
  - 011 only vector that detects *c s-a-0*
  - {*c s-a-0, a s-a-1*} is not detectable by 011

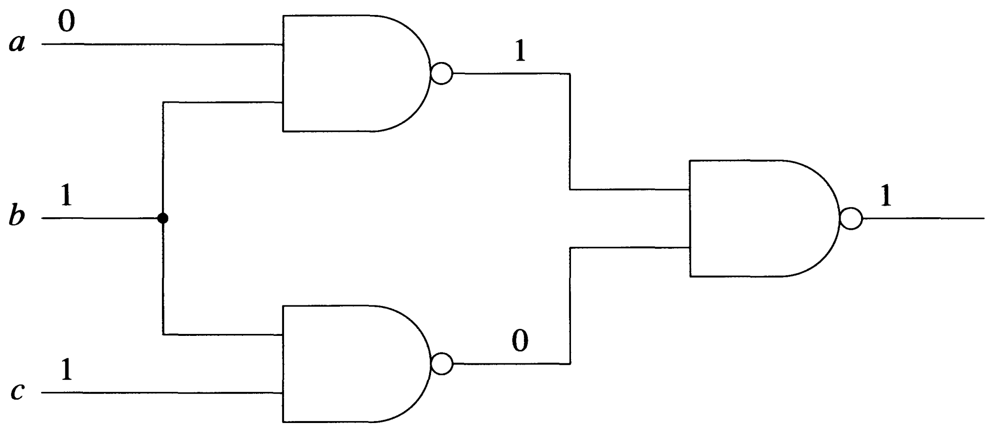


Figure 4.24

Masking under a Test Set

- **Definition 4.10**: Let $T_g' \subseteq T$ be the set of all tests in T that detect a fault g. We say that a fault f **masks** the fault g ***under a test set*** **T** iff the multiple fault $\{f, g\}$ is not detected by any test in T_g'

- Functional masking → masking under a test set
  - $$T_g' \subseteq T_g$$

Problem

→ If *f* masks *g*, then fault {*f*, *g*} is not detected by tests that detect *g* alone. But can there be other tests to detect {*f*, *g*}?

$\{c\ s\text{-}a\text{-}0,\ \ a\ s\text{-}a\text{-}1\}$

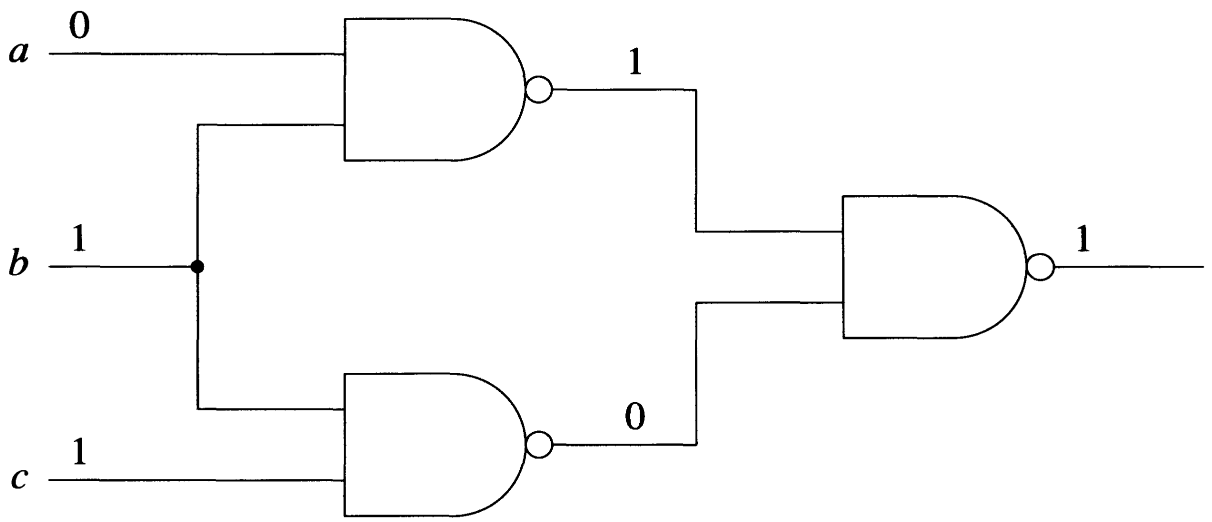


Figure 4.24

Problem – Detection of Multiple Faults

→ Given a complete test set T that detects all single faults, can there exist a multiple fault $F = \{f_1, f_2, \ldots, f_k\}$ such that F is not detected by T?

Example

- *T* = {1111, 0111, 1110, 1001, 1010, 0101}
  - *f* = *B s-a-1* and *g* = *C s-a-1*
  - 1001 is the only vector that detects *f* and *g* SSFs, but...

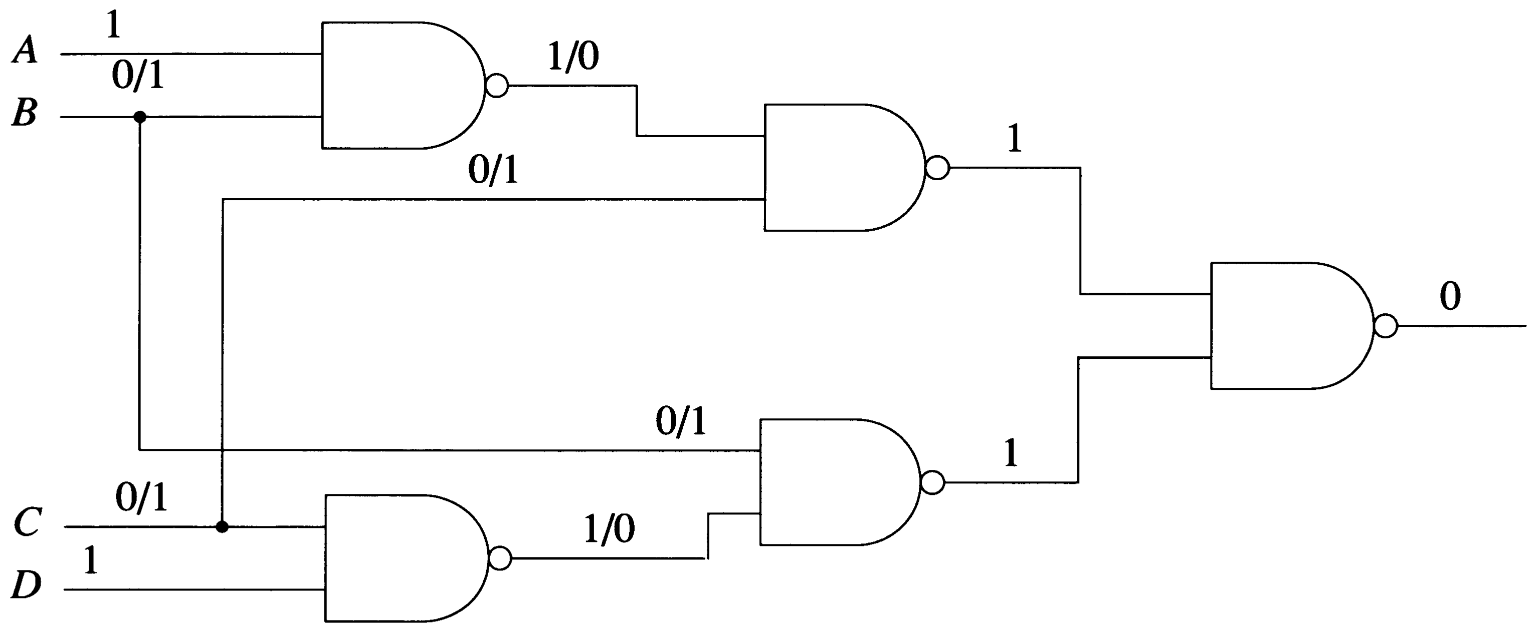


Figure 4.25

Example

- → 1001 cannot detect {*f*, *g*}
  - → *f* masks *g*, and *g* masks *f* – **circular masking**

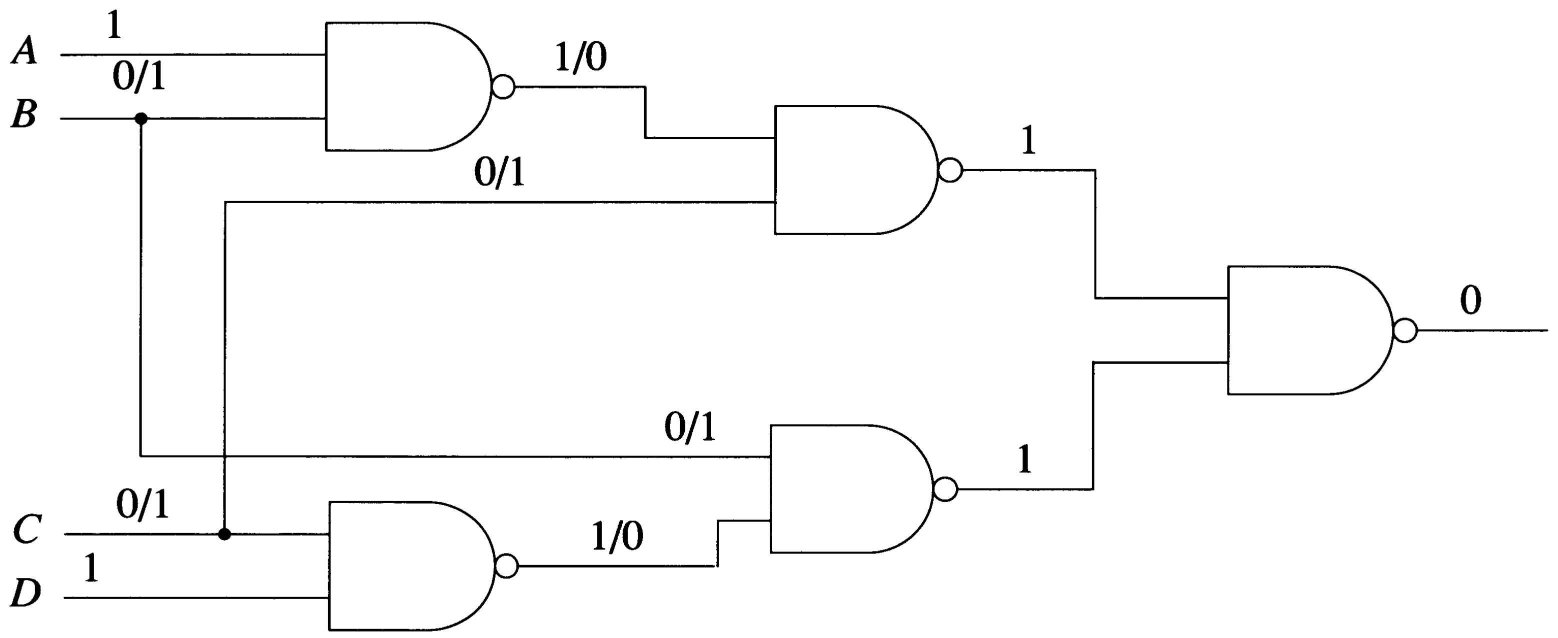


Figure 4.25

What % of MSFs can escape detection by a complete SSF test set (T)?

- → In an irredundant 2-level circuit, *T* also detects all MSFs
- → In a fanout-free circuit, any *T* detects all double and triple faults, and there exists a T for SSFs that detects all MSFs
- → In an internal fanout free circuit, any *T* detects at least 98% of MSFs with multiplicity $k < 6$
- → A test set that detects all MSFs defined on all primary inputs without fanout and all fanout branches of a circuit *C* detects all multiple faults in *C.*

MSFs or Not?

- A test set *T* for SSFs also detects most MSFs.
  - Typically focus on MSFs not detectable by *T*.
- Probability of MSFs is low.
  - Modern semi-manufacturing is highly reliable
- A SSF is guaranteed to be detected (GTBD) if it can be detected unconditionally.
  - Any MSF that includes a GTBD SSF is also GTBD.

Summary

- Logic faults can represent various physical faults
- Fault detection
  - Find a test that causes deviation in output responses
  - Undetectable faults → redundancy
- Fault collapsing
  - Fault equivalence & dominance
- Single stuck-fault model
  - Possible to find a complete test set
- Multiple fault model
  - A SSF detection test set can find many MSFs

Backup

Corollary 4.1

Let *j* be a line sensitized to the fault *l* *s-a-v* (by a test *t*), and let *p* be the inversion parity of a sensitized path between *l* and *j*.

1. The value of *j* in *t* is $v \oplus p$.
2. If there are several sensitized paths between *l* and *j*, then all of them have the same inversion parity.

Redundancy in Circuits

→ Redundancy can be by design i.e., intentional

Fault tolerant design

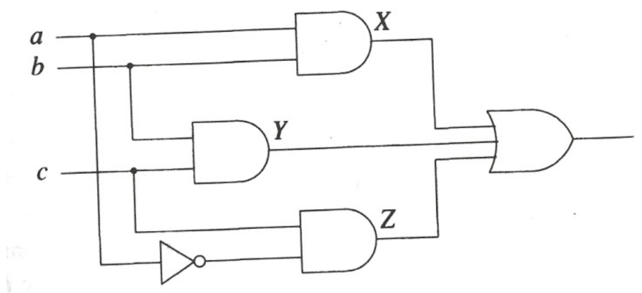


Figure 4.8

Hazard-free design

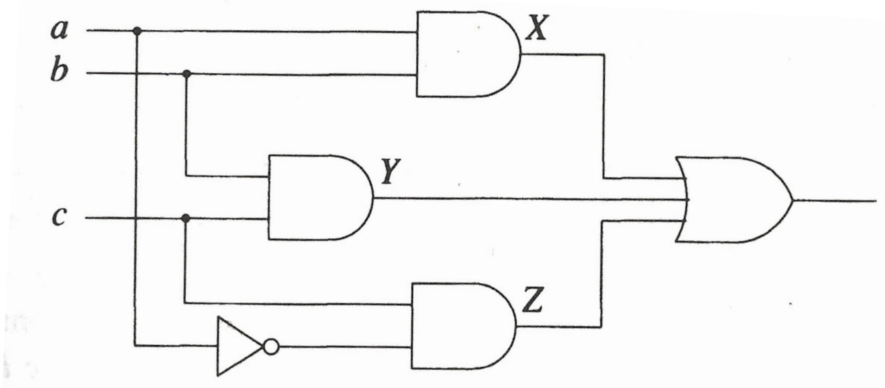


Figure 4.8