Einführung in die Technische Informatik

Test digitaler Schaltungen

Robert Wille

Begriffe

- Entwurf:
  Entwicklung einer Schaltung aus der (funktionalen) Spezifikation

- Verifikation:
  Nachweis der Korrektheit des Entwurfs

- Test:
  Überprüfung der produzierten Schaltung auf Fertigungsfehler

Verifikation vs. Test

Verifikation

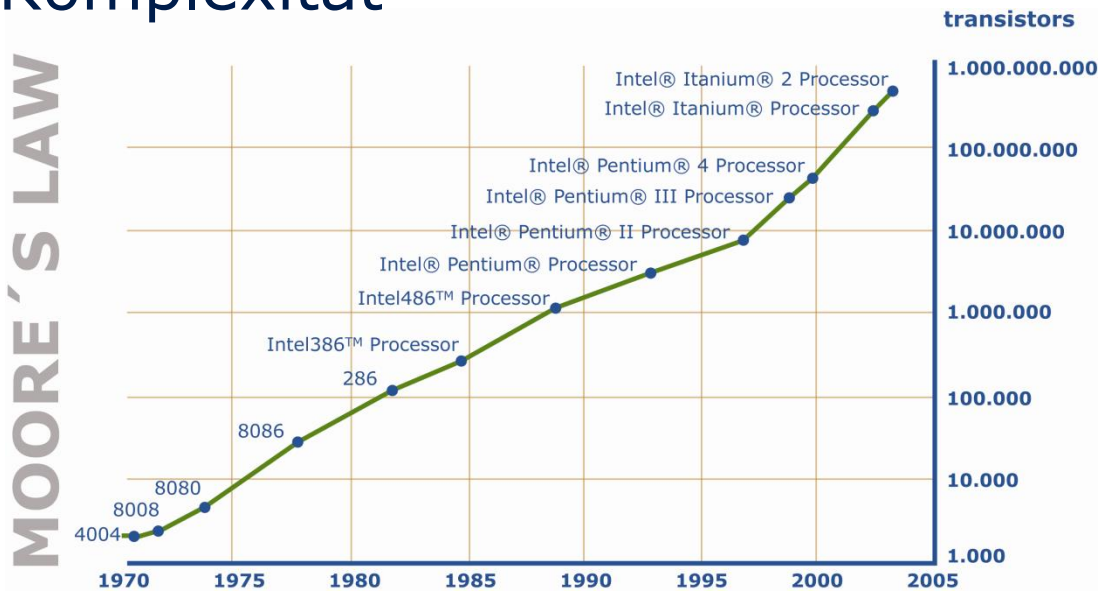
- Zeigt die Korrektheit des Entwurfs
- Durchgeführt durch Simulation, Hardware-Emulation oder formale Methoden
- Durchgeführt einmal vor der Produktion
- Verantwortlich für die Qualität des Entwurfs

Test

- Zeigt die Korrektheit der produzierten Hardware
- Zweiteiliger Prozess:
  - 1. Testgenerierung
  - 2. Testanwendung
- Testanwendung ausgeführt auf jedem Schaltkreis
- Verantwortlich für die Qualität des produzierten Schaltkreises

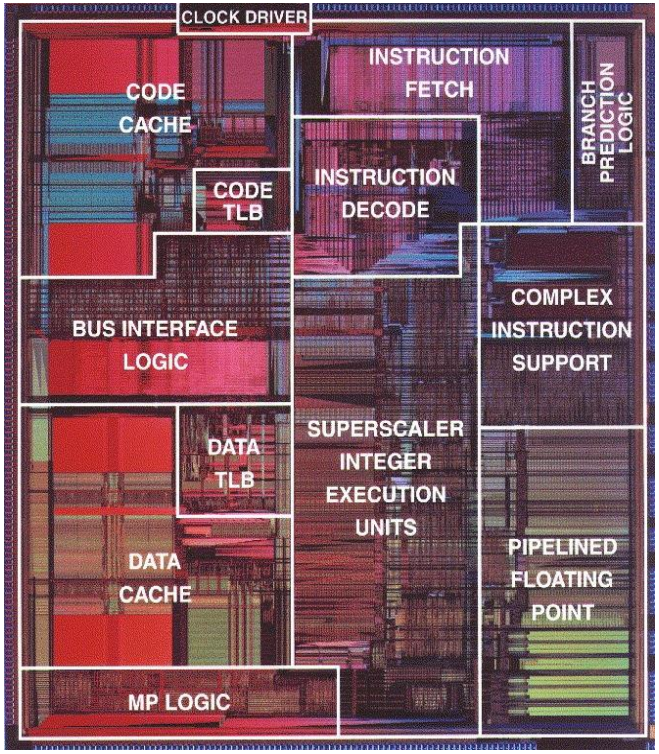
Warum Testen?

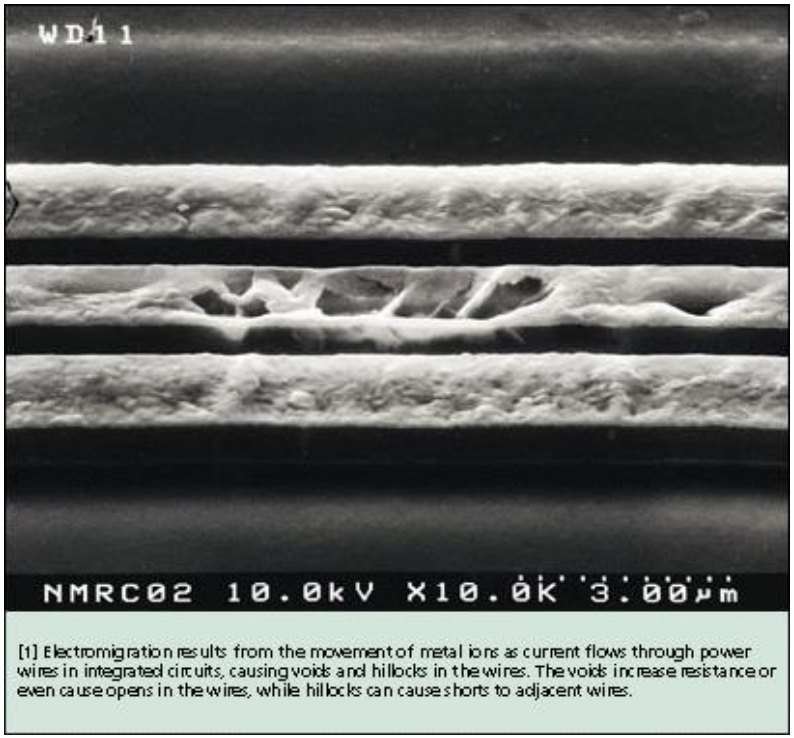
- Ständig steigende Komplexität
- Oftmals 2/3 der produ-zierten Chips fehlerhaft!
- 35% Gesamtproduktionskosten



Probleme bei der Fertigung

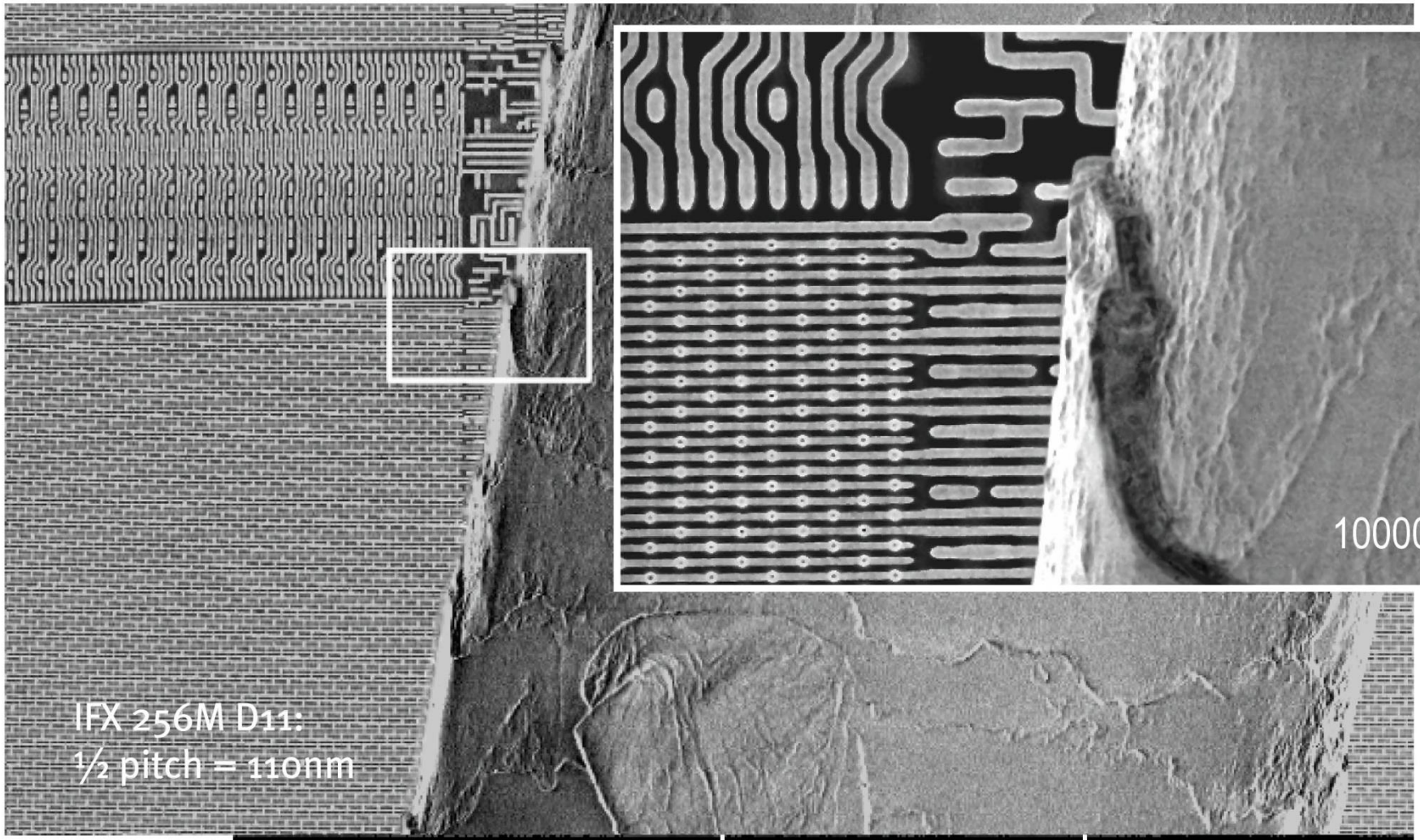
- Integration von bis zu 1 Milliarde Komponenten



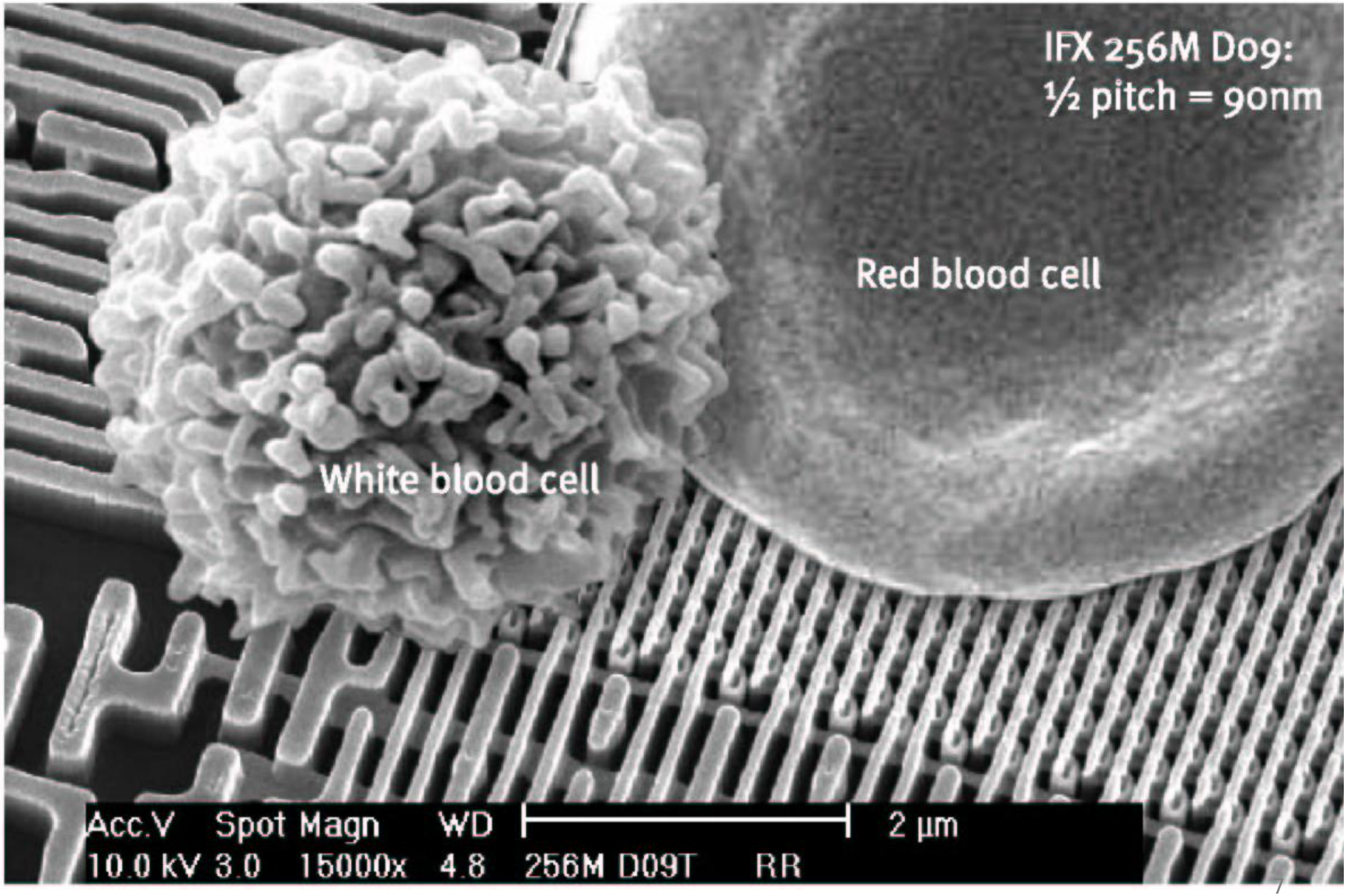


[1] Electromigration results from the movement of metal ions as current flows through power wires in integrated circuits, causing voids and hillocks in the wires. The voids increase resistance or even cause opens in the wires, while hillocks can cause shorts to adjacent wires.

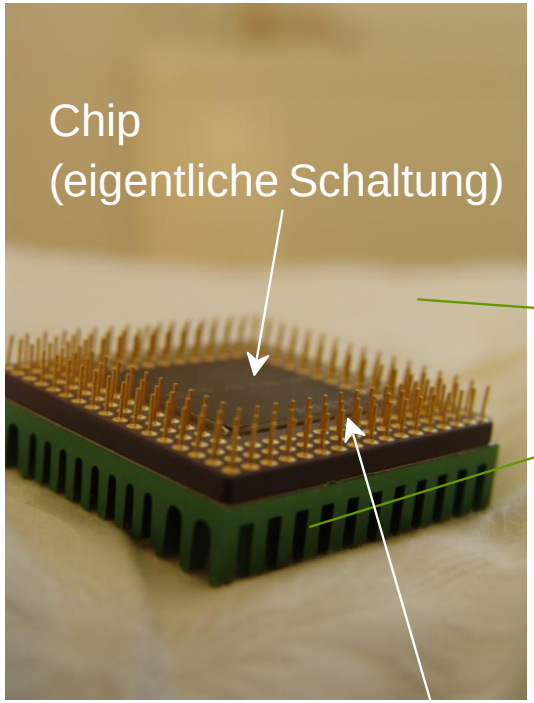
It´s a small, small, small World



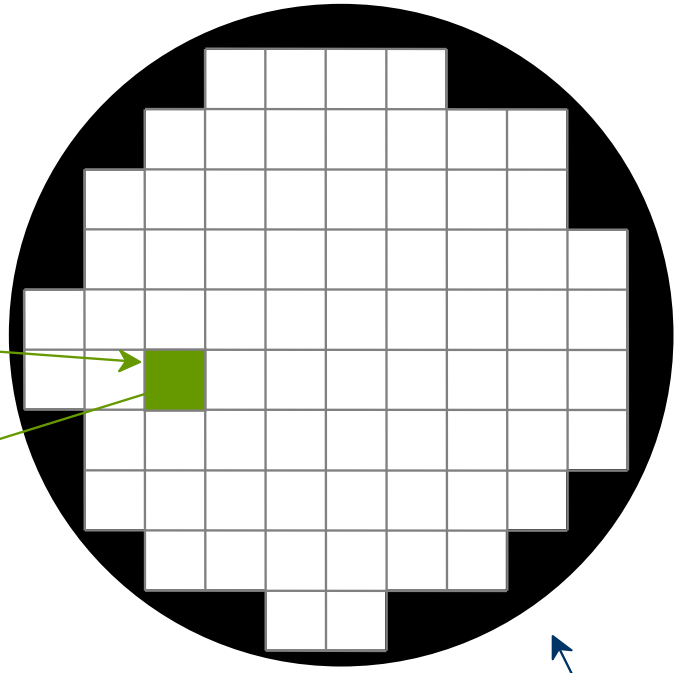
It´s a small, small, small World



Chip und Wafer



Gehäuse

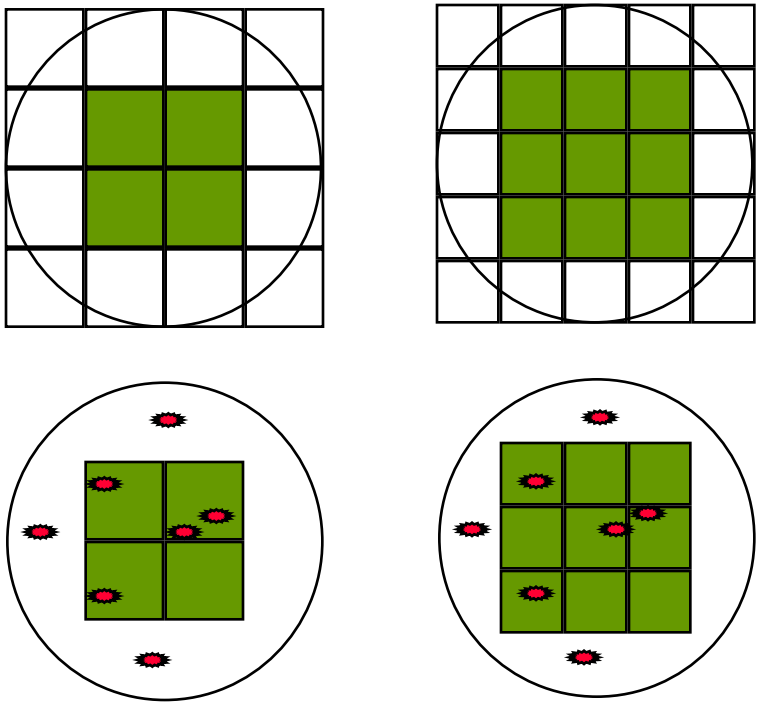


Wafer

Anschlüsse für Platine

Chipfläche und Ausbeute

Ausbeute: (Anzahl der defektfreien Chips) / (Anzahl der hergestellten Chips) · 100 %



Ausbeute:

Einfluss der Chipfläche auf die Ausbeute

Flächenoptimierung ⇒ höhere Produktivität !

Der „ideale“ Test

- Entdeckt alle Defekte
- Alle funktional korrekten Bausteine „passieren“
- Probleme:
  - Sehr große Anzahl an Defekten muss getestet werden
  - Schwierig für einige reale Defekte, Tests zu generieren

Reale Tests

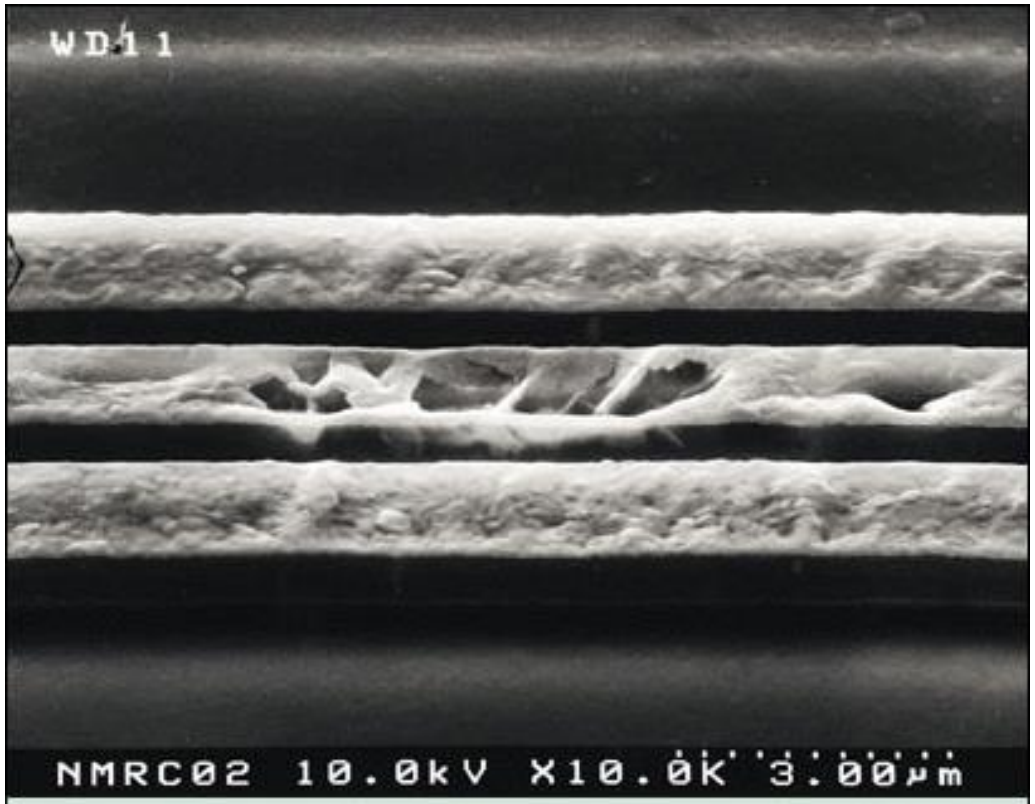
- Lege analysierbares Fehlermodell zu Grunde

- Mögliche Probleme:
  - Unvollständige Überdeckung durch Fehlermodelle
  - Korrekte Chips werden aussortiert
  - Fehlerhafte Chips passieren den Test $\rightarrow$ *Fehlerrate*

Warum Fehlermodelle?

- Reale Defekte zu zahlreich und oft nicht „sauber“ analysierbar
- Fehlermodell identifiziert Ziele des Testens
- Fehlermodell ermöglicht Analyse
- Effektivität muss durch Experimente nachgewiesen werden
  - Fehlermodelle kommen aus Praxis

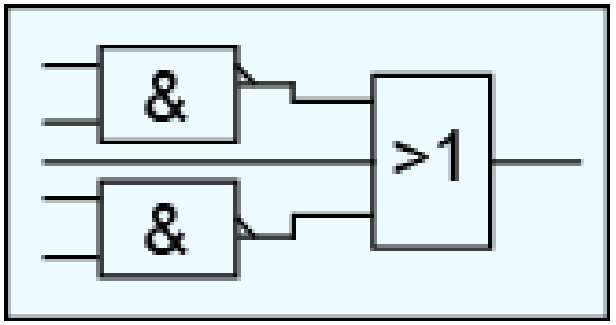
Reale Fehler



[1] Electromigration results from the movement of metal ions as current flows through power wires in integrated circuits, causing voids and hillocks in the wires. The voids increase resistance or even cause opens in the wires, while hillocks can cause shorts to adjacent wires.

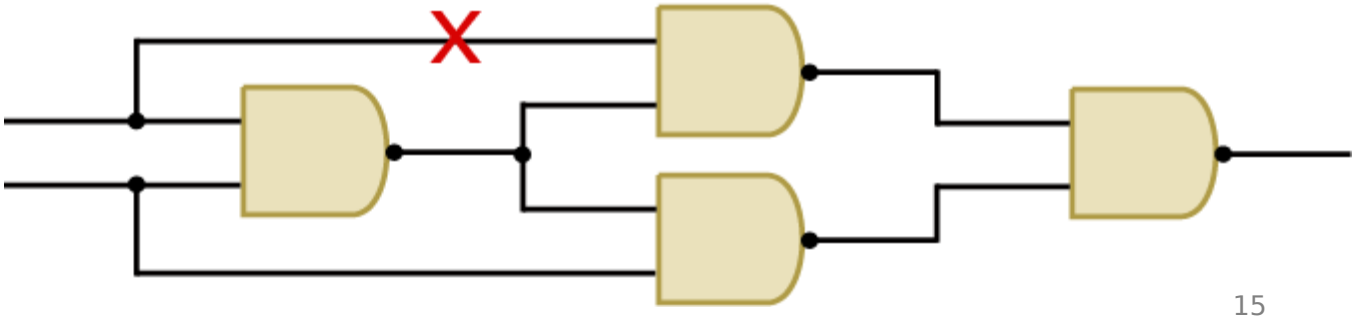
Strukturelle Fehlermodelle

- Enthält Fehler, die die Verbindungen zwischen Gattern betreffen
  - z.B. Stuck-at Fehler
- Mehrheit physikalischer Defekte lässt sich auf Stuck-at Fehler abbilden
  - Bis zu 85% der Defekte werden erkannt



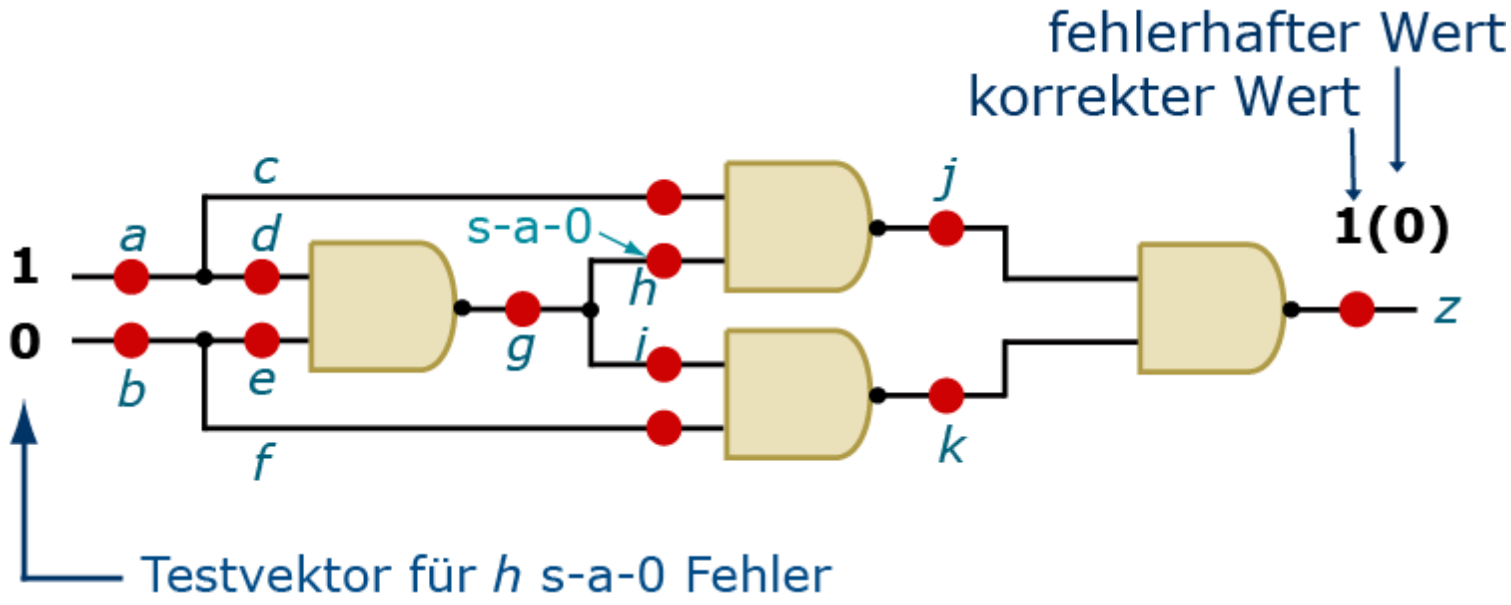
Single Stuck-at Fehler

- Nur eine Leitung ist fehlerhaft
  - Daher: single
- Fehlerhafte Leitung konstant auf 0 oder 1
- Fehler kann an Eingang oder Ausgang eines Gatters liegen
- Bezeichnung auch als Haftfehler



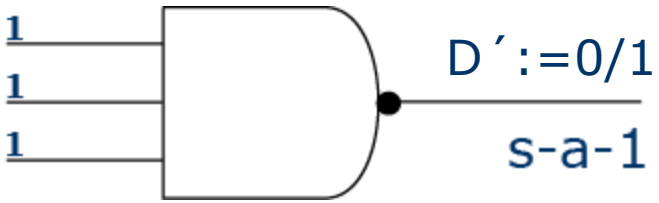
Single Stuck-at Fehler

- Beispiel:
  - XOR Schaltkreis hat 12 Fehlerstellen (●) und 24 Single Stuck-at Fehler.



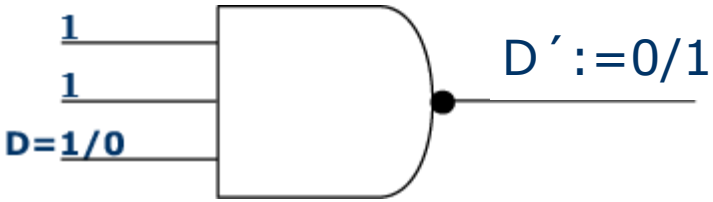
Berechnung eines Testmusters

- Zwei grundsätzliche Schritte
  - Primäre Eingänge belegen, so dass Fehler am Fehlerort erkennbar ist
    - Fehler einstellen



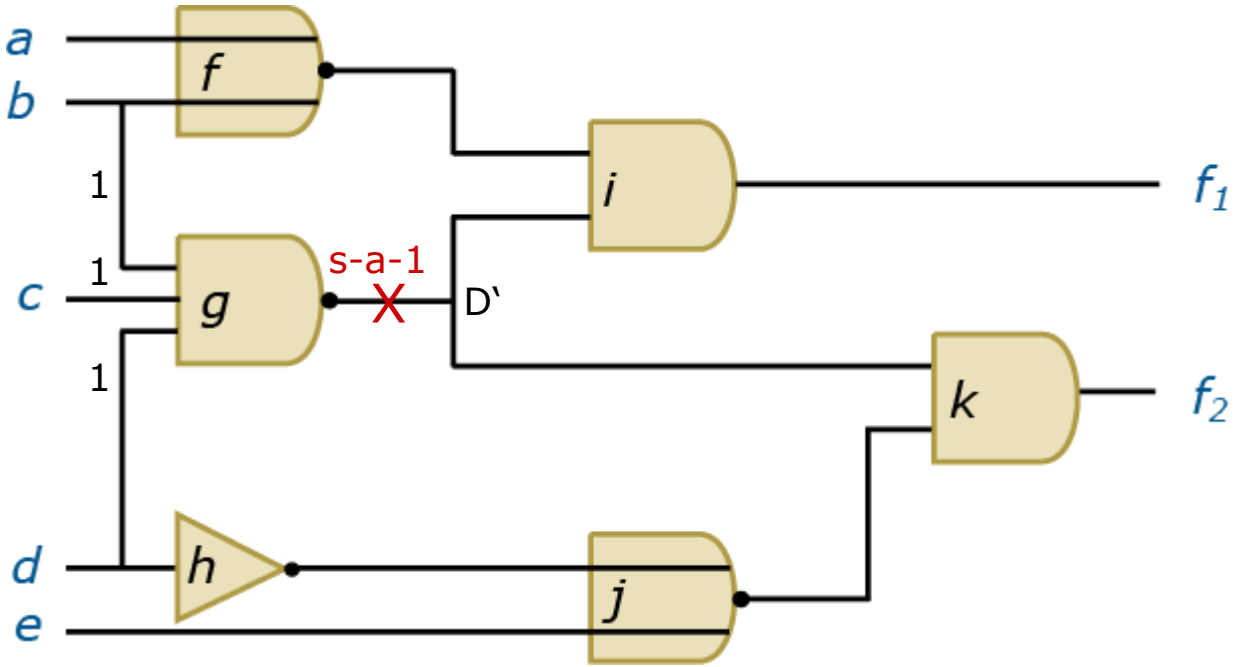
Berechnung eines Testmusters

- Zwei grundsätzliche Schritte
  - Primäre Eingänge belegen, so dass Fehler am Fehlerort erkennbar ist
    - Fehler einstellen
  - Weitere Signale belegen, so dass Fehler am Ausgang der Schaltung erkennbar ist
    - Fehler propagieren



Schaltungen mit Rekonvergenzen

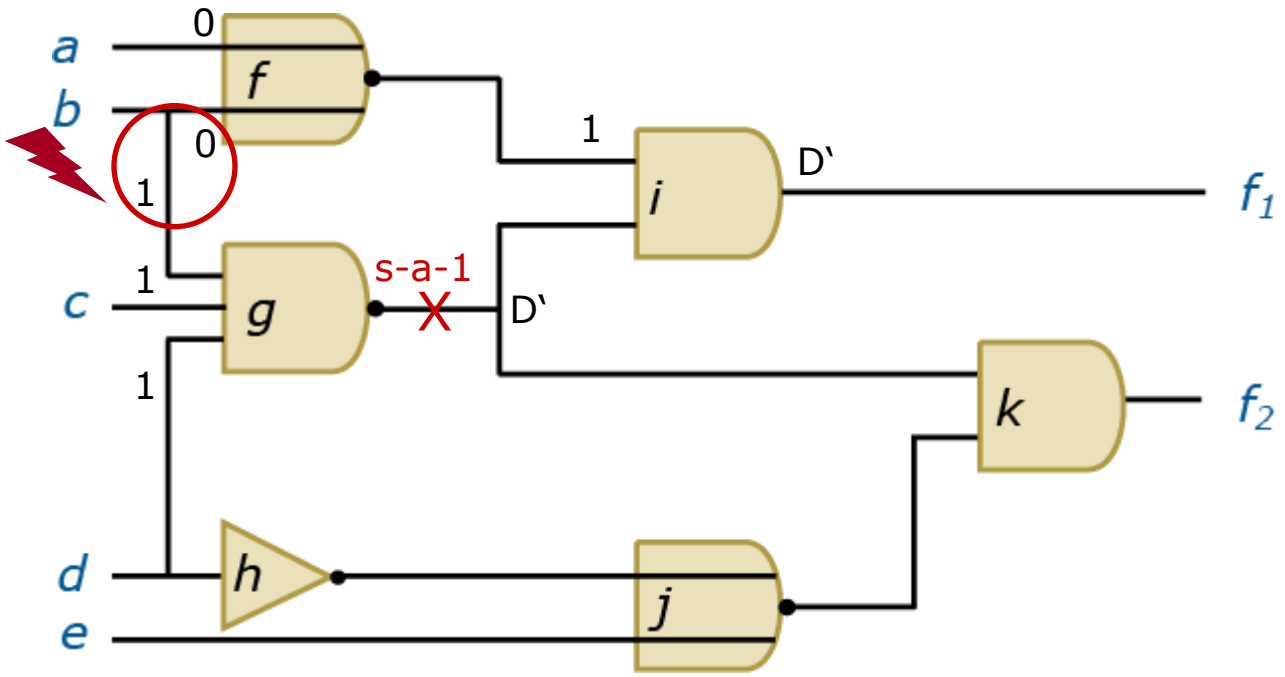
Einstellen des Fehlers:
justify(g,0)



Schaltungen mit Rekonvergenzen

Propagiere den Fehler:

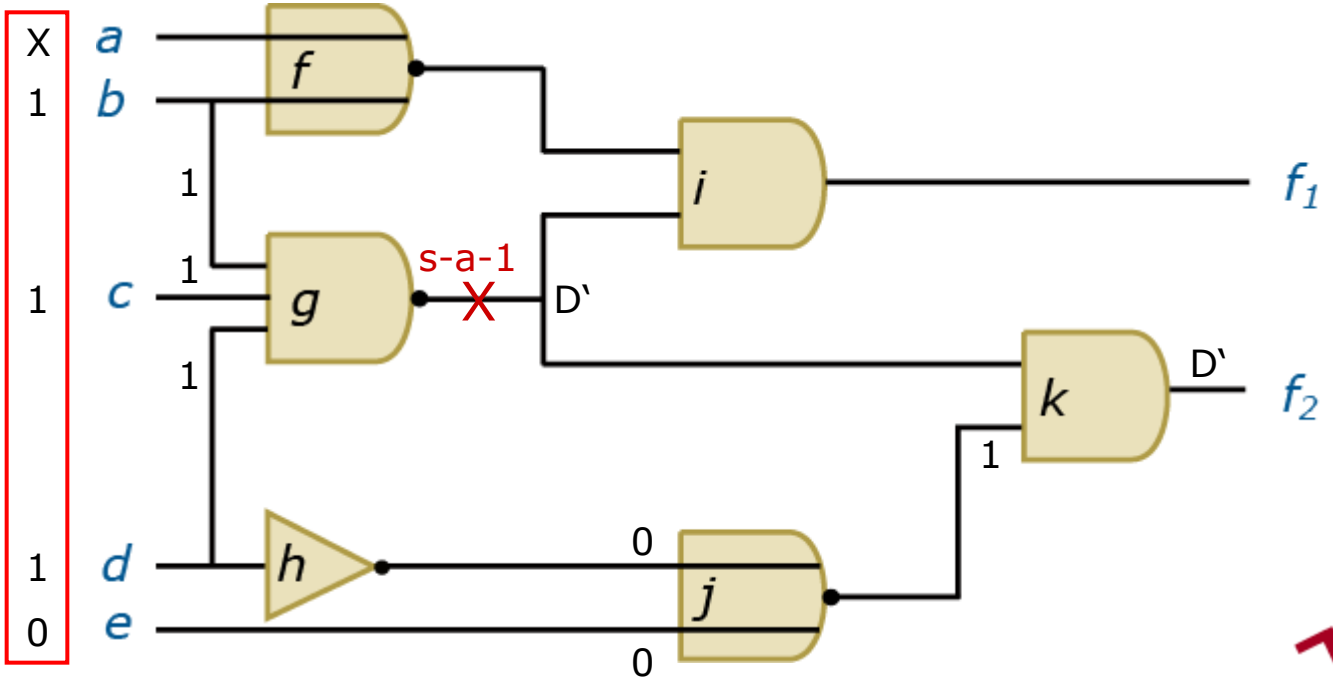
1. Versuch: propagiere über Baustein *i*



Schaltungen mit Rekonvergenzen

Propagiere den Fehler:

2. Versuch: propagiere über Baustein *k*



Problem bei Algorithmus

- Testmustergenerierung erfordert Entscheidungen, die nicht lokal sind

- Ist Konflikt aufgetreten, müssen Entscheidungen rückgängig gemacht werden

→ Backtracking

Ansätze zur Lösung

- PODEM
- FAN
- SOCRATES

- SAT-basiert

SAT-based TPG

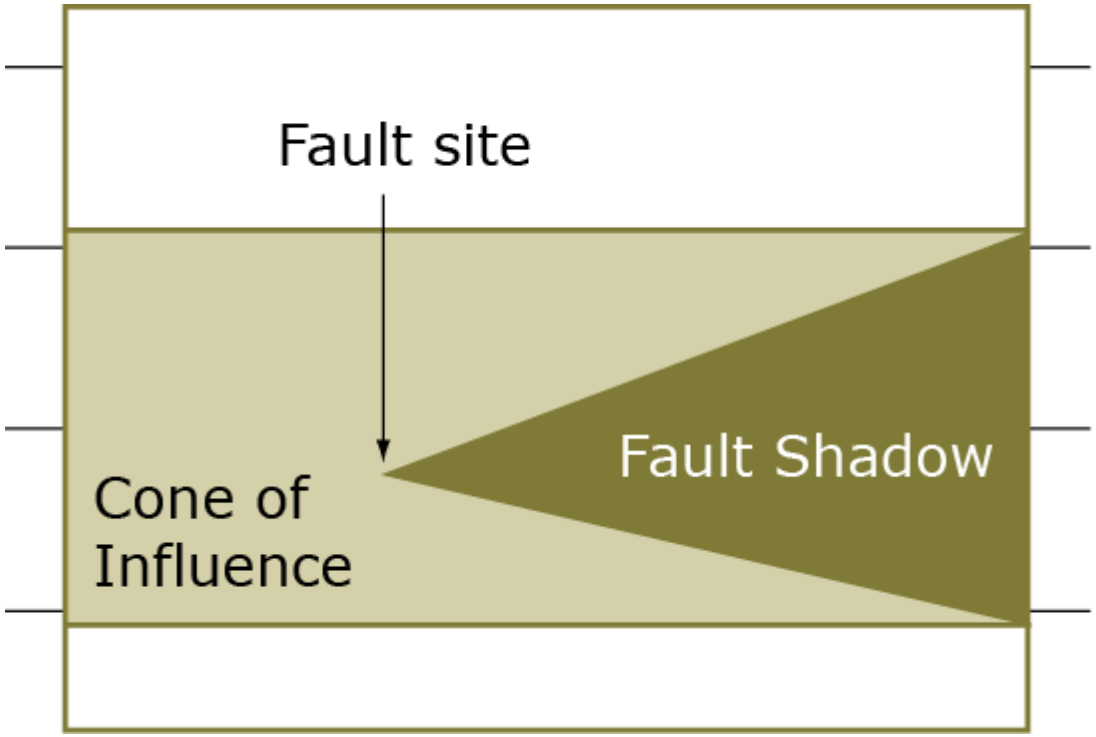
- **Eingabe:** Schaltkreis C, Fehler F

1. Fehlermodellierung:
   BD zwischen fehlerfreiem und fehlerhaften Schaltkreis
2. Übersetze nach CNF
3. Benutze SAT Solver um eine Lösung zu berechnen

- **Ausgabe:** Klassifikation von F, Testvektor T

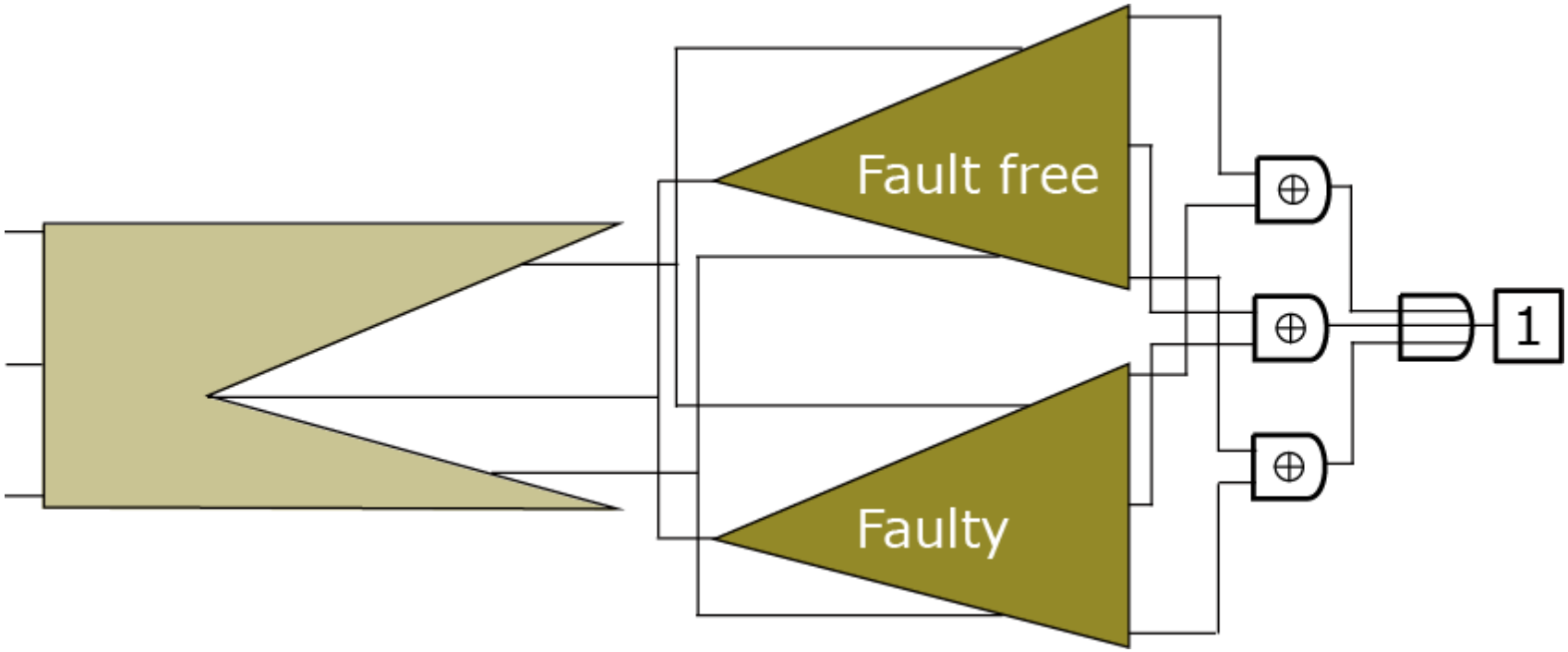
Schaltkreis → CNF

- Erstelle CNF nur von der Teilschaltung, die unmittelbar Einfluss auf den Fehler hat.

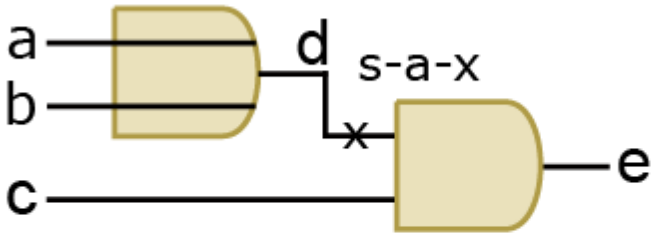


Schaltkreis → CNF

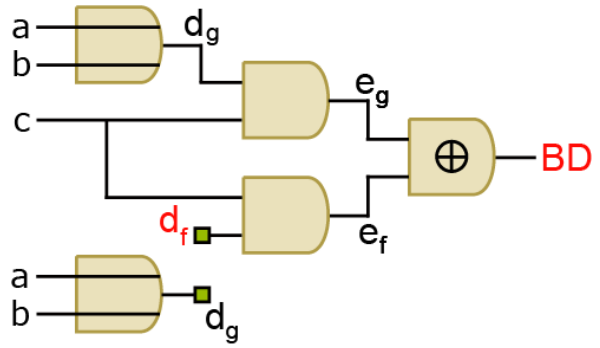
- Teilschaltung, die nicht von der fehlerhaften Leitung getrieben wird, kann gemeinsam genutzt werden.



Fehlermodellierung



KNF



- $F = (\overline{c} + \overline{d}_g + e_g) \cdot (c + \overline{e}_g) \cdot (d_g + \overline{e}_g)$
  $\cdot (a + b + \overline{d}_g) \cdot (\overline{a} + d_g) \cdot (\overline{b} + d_g) \cdot (d_f)$
  $\cdot (\overline{c} + \overline{d}_f + e_f) \cdot (c + \overline{e}_f) \cdot (d_f + \overline{e}_f)$
  $\cdot (e_g + e_f + \overline{BD}) \cdot (\overline{e}_g + \overline{e}_f + \overline{BD})$
  $\cdot (\overline{e}_g + e_f + BD) \cdot (e_g + \overline{e}_f + BD) \cdot (BD)$

- F ist die CNF für den Schaltkreis mit d s-a-1
- Eingabe erfüllt CNF → Fehler ist entdeckt
- CNF nicht erfüllbar → Fehler ist redundant

Fehlerorientierte Testmustergenerierung