



Test und Verlässlichkeit

Foliensatz 5: Hardware-Test und Selbsttest.

Prof. G. Kemnitz

Institut für Informatik, TU Clausthal (TV_F5)
6. November 2022



Inhalt TV_F5: Hardware-Test und Selbsttest

[Vorlesung 14]

Test digitaler Schaltungen

Fehlermodellierung

- 2.1 Schaltkreisfehler
- 2.2 Lokale Fehler
- 2.3 FM für DIC
- 2.4 Nachweisbeziehungen

Testsuche

- 3.1 Fehlersimulation
- 3.2 D-Algorithmus
- 3.3 Implikationstest

- 3.4 Suchraumstrukturierung
- 3.5 Komplexe Funktionsbausteine
- 3.6 Sequentielle Schaltungen
- 3.7 Speichertest

Selbsttest

- 4.1 Pseudo-Zufallsregister
- 4.2 Signaturregister
- 4.3 Selbsttest mit LFSR
- 4.4 Fehlerorientierte Wichtung
- 4.5 Testbus

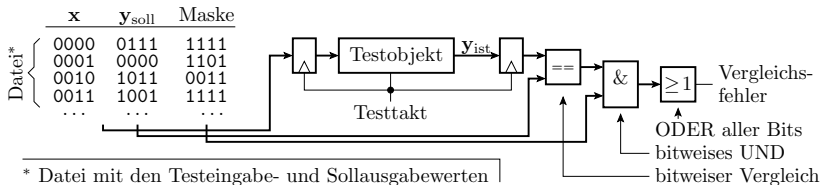
Vorlesung	1	2	3	4
bis Abschn.	2.?? (??)	4.?? (??)	4.?? (??)	5.?? (??)



Test digitaler Schaltungen



Test digitaler Bausteine



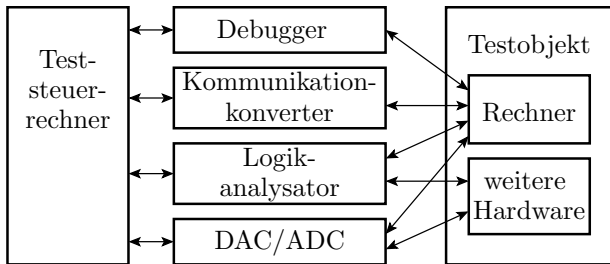
Wiederhole für jeden Taktschritt des Tests:

- Bereitstellung logischer Eingabewerte und
- Abtasten und Auswertung der vorherigen Ausgaben.

Auswertung vorzugsweise Vergleich mit Sollwerten unter Ausmaskierung von Ausgabebits ohne definierten Sollwert.

[Initialisierung, zusätzliche Steuer- und Beobachtungspunkte, LA Laufzeitkontrolle]

Test von Rechnersystemen



Typisches Prüfsystem ist ein Teststeuerrechner mit

- Einschüben /Adaptoren für die Kontaktierung der Schnittstellen,
- incl. ADUs und DAUs für Mixed-Signal Testobjekte,
- Logikanalysator zur Untersuchung von Zeitabläufen und
- Debugger für Schrittbetrieb, Setzen von Haltepunkten, Trace-Aufzeichnung, Lesen und Schreiben von Variablen und Speicherinhalten an Haltpunkten.

[LA Ablaufkontr., Debugger-HW-Unterstützung Prozessor, DFT]



Testauswahl und Güte

Bei einem Soll/Ist-Vergleich der Testausgaben mit korrekten Sollwerten sind Maskierungen bei der Kontrolle und Phantomfehler vernachlässigbar. Die Fehlerüberdeckung als Gütemaß hängt nur von der Anzahl und Auswahl der Testeingaben ab. Abschätzung mit Modellfehlern.

Wiederholung einiger Begriffe zum Thema:

Fehlermodell: Algorithmus, der aus einer simulier- oder abarbeitbaren Beschreibung eine Modellfehlermenge berechnet.

Modellfehler: geringfügige Verhaltes- oder Beschreibungsänderung.

Haftfehler: Annahme von ständig eins oder ständig null für einen Gatteranschluss.

Geziele Suche: Suche von Testeingaben, für die der Modellfehler die Ausgabe verfälscht.

Zufallstest: Auswahl unabhängig von den Modellfehlern und den zu findenden Fehlern.

[Schwerpunkt heute: tatsächliche Fehler \Leftrightarrow Haftfehler abhängig von Testauswahl]



Test mit allen Eingaben unmöglich

Für den Nachweis, dass ein Service für alle Eingabedaten korrekte Ergebnisse liefert, müsste er mindestens mit allen Eingaben ausprobiert werden. Bereits ab wenigen Eingabebits unmöglich:

	m	2^m	t^*
Gatter, 4 Eingänge	4	16	16 μ s
ALU, 68 Eingänge	68	$3 \cdot 10^{20}$	10^7 Jahre
vier Eingabevariablen vom Typ int32_t	128	$3 \cdot 10^{38}$	10^{25} Jahre

(m – Anzahl der Eingabebits; 2^m – Anzahl der Eingabemöglichkeiten; t^* – Testdauer bei einer Service-Ausführungszeit von 1μ s.)

- Die meisten Systeme verarbeiten mehr als 128 Eingabebits.
- Hinzu kommen oft tausende oder mehr gespeicherte Bits, die auch mit variiert werden müssten.
- Geschätzte Zeit seit dem Urknall $14 \cdot 10^9$ Jahre.
- Es gibt auch Fehler, die verlangen Eingabefolgen für den Nachweis. Alle Variationen von 2, 3, ... Eingabevektoren ...



Nachweis aller unterstellten Fehler möglich

Für regelmäßig strukturierte Schaltungen lassen sich oft Algorithmen für die Eingabegenerierung so formulieren, dass alle Fehler nach einem bestimmten Fehlermodell nachweisbar sind. Beispiel Nachweis alle Kurzschlüsse auf einer Leiterplatte, wenn auf allen Leitungen 0 oder 1 gesteuert werden kann und die Leitungen beobachtbar sind.

Hinreichende Nachweisbedingung für alle Kurzschlüsse ist, dass sich die gesteuerten Leitungspegel paarweise in mindestens einem Testschritt unterscheiden.

	<u>Leitungsnummer</u> →															
Testschritt ↓	0	1	0	1	0	1	0	1	0	1	0	1	0	1		
	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Logarithmische Zunahme der Testsatzlänge mit der Anzahl der potentiell kurzgeschlossenen Leitungen.

Ein Beispiel für einenen Speichertest folgt später in Abschn. 2.7.



Fehlermodellierung



Schaltkreisfehler



Warum Schaltkreise?

Die Konzepte prüfgerechter Entwurf, Fehlermodellierung, Testberechnung, ... überwiegend für digitale Schaltkreis entwickelt:

- Schlechte Steuer- und Beobachtbarkeit.
- Hohe Fehlerbeseitigungskosten.
- Zwang zu kompromissloser Fehlervermeidung und Beseitigung.
- Hohe Anforderungen an die Fehlerüberdeckung der Tests, ...

Schaltkreise heute fehlerarm und zuverlässig.

Gegenstück SW:

- änderungsfreundlich, relativ geringe Fehlerbeseitigungskosten.
- Features oft besser verkaufbar als Qualität.

SW oft voller Fehler und unzuverlässig, aber das muss nicht bleiben.

Mit den wachsenden Verlässlichkeitsanforderungen an IT-Systeme insgesamt kann die restliche IT-Branche von den Erfahrungen der Schaltkreisbranche lernen.



Einteilung in lokale und globale Fehler

Globale Fehler:

- Fehlerhafte Schichteigenschaften durch Prozesssteuerfehler. Betroffen sind alle Strukturelemente derselben Halbleiter-, Leitungs- oder Isolationsschicht.
- Großflächig überflüssiges oder fehlendes Material. Mehrfachkurzschlüsse oder Unterbrechungen.

Lokale Fehler:

- Unterbrechungen von Verbindungen,
- Kurzschlüsse zwischen benachbarten leitenden Gebieten.
- Transistoren, die nicht richtig ein- oder ausschalten,
- Leckströme ohne logische Fehlerwirkung (siehe auch später Foliensatz F6: Frühausfälle).



Globale Fehler für Testauswahl uninteressant

- Großflächige Fehler \Rightarrow hohe FF-Rate \Rightarrow Grobtest.
- Prozesssteuerfehler beeinträchtigen die Struktureigenschaften aller Elemente einer Schicht beeinträchtigen \Rightarrow Parametertest.

Grobtest:

- Spannung anlegen und
- kleine Funktionsstichprobe ausprobieren.

Parametertest:

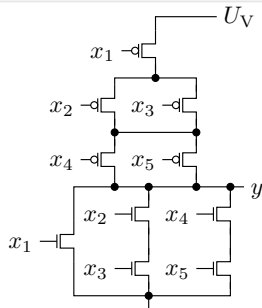
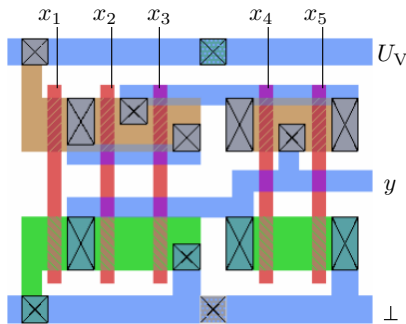
- Stichprobenkontrolle der Transistoreigenschaften, Leitwerte und Kapazitäten an speziellen Teststrukturen, auch schon nach Fertigungszwischenschritten.
- Elektrische Messungen an den Schaltkeisanschlüssen incl. Versorgungsstrom.






Testauswahl ist nur für kleinflächige lokale Fehler, die nur einzelne Transistoren und Verbindungen beeinträchtigen, anspruchsvoll.



Lokale Fehler

Transistorebene

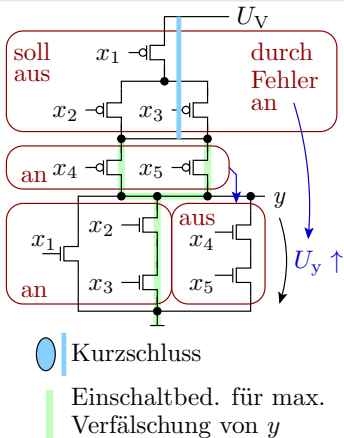
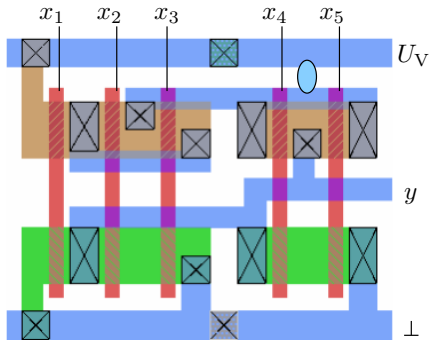


-  Durchkontaktierung
-  n-Gebiet
-  p-Gebiet
-  Polysilizium
-  Metall

$$y = \begin{cases} 1 & \text{wenn } \bar{x}_1 \wedge (\bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_4 \vee \bar{x}_5) \\ 0 & \text{wenn } x_1 \vee (x_2 \wedge x_3) \vee (x_4 \wedge x_5) \end{cases}$$

$$= \overline{x_1 \vee (x_2 \wedge x_3) \vee (x_4 \wedge x_5)}$$

Kurzschluss im Gatters



Fehlerbedingte Spannungserhöhung¹ an y :

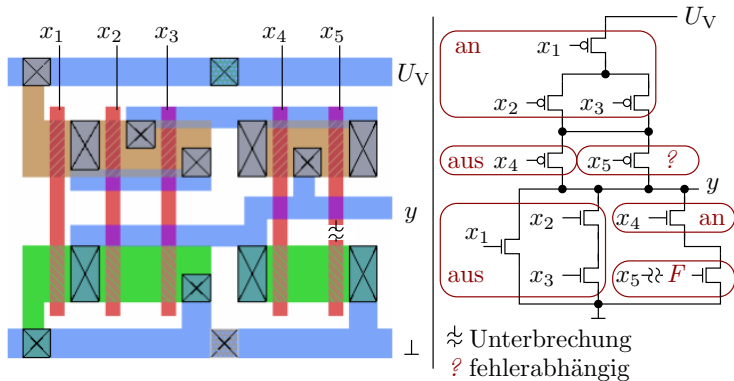
$$(\bar{x}_1 \wedge (\bar{x}_2 \vee \bar{x}_3) = 0 \text{ (aus)}) \wedge (\bar{x}_4 \vee \bar{x}_5 = 1 \text{ (an)})$$

Größte fehlerbedingte Spannungserhöhung an y :

$$(x_2 = x_3 = 1) \wedge (x_1 = x_4 = x_5 = 0)$$

¹Auch nachweisbar am statischen Ruhestrom (IDDQ) und verlängerter Einschaltzeit.

Offenes Gate

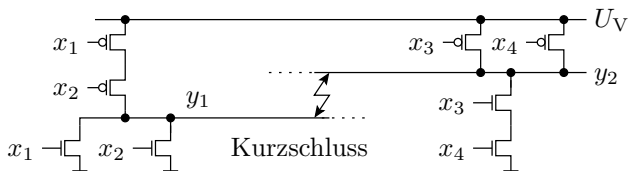


Nachweisvoraussetzung: $(x_1 \vee (x_2 \wedge x_3) = 0) \wedge (x_4 = 1) \wedge \dots$

- $(F = 0) \wedge (x_5 = 1)$: $y = 0 \rightarrow 1?$ oder verzögerter Abfall
 - $(F = 1) \wedge (x_5 = 0)$: $y = 1 \rightarrow 0?$, verzögerter Anstieg oder IDDQ
- (? – ob logische Verfälschung hängt ab von Transistorbreiten, ...).

[Gatepotential F auch zwischen 0 und 1 oder driftend]

Kurzschluss zweier Gatterausgänge



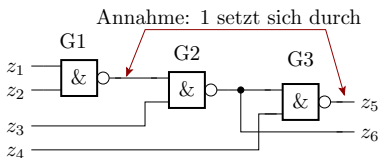
Mögliche Nachweisbedingungen:

- 1 $\bar{x}_1 \wedge \bar{x}_2 = 1$ und $x_3 \wedge x_4 = 1$ ($y_{1\text{Soll}} = 1$ und $y_{2\text{Soll}} = 0$)
- 2 $x_1 \vee x_2 = 1$ und $\bar{x}_3 \vee \bar{x}_4 = 1$ ($y_{1\text{Soll}} = 0$ und $y_{2\text{Soll}} = 1$)

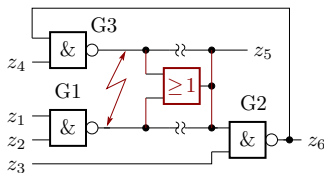
Ob sich dabei $y_1 = y_2 = 0$ oder $y_1 = y_2 = 1$ durchsetzt, hängt von den Transistorbreiten, bzw. Transistorsteilheiten ab. Der verfälschte y -Wert muss zusätzlich beobachtbar sein.

Zusätzliches Speicherverhalten durch Kurzschluss

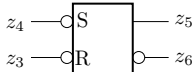
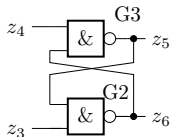
Schaltung mit Kurzschluss



Kurzschlussnachbildung durch ein ODER



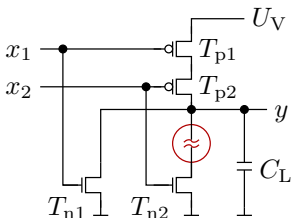
Ersatzschaltung
für $z_1 = z_2 = 1$



z_4	z_3	z_2	z_1	z_6	z_5
0	1	1	1	0	1 (setzen)
1	0	1	1	1	0 (löschen)
1	1	1	1		speichern

z_4	z_3	z_2	z_1	z_6	z_5
0	0	1	1	1	1
		↓↓			↓↓ ändern nach
1	1	1	1		unbestimmt

Stuck-Open-Fehler



x_2	x_1	y
0	0	1 (setzen)
0	1	0 (rücksetzen)
1	0	speichern
1	1	0 (rücksetzen)

Ausgewählte Unterbrechungen und Transistordefekte können bewirken, dass Gatterausgänge für bestimmte logische Eingaben isoliert sind oder nur zu langsame auf- oder entladen werden. Dieser Fehlertyp wird als Stuck-Open-Fehler bezeichnet und lässt sich nur über Schaltvorgänge am Gatterausgang zuverlässig nachweisen.



FM für DIC

Fehlermodelle (FM) für digitale Schaltkreise

Etablierte Fehlermodelle für digitale Schaltkreise:

- Haftfehler,
- Gatterverzögerungsfehler,
- IDDQ-Fehler und
- Zellenfehler (regelmäßig strukturierte DIC, RAM, ...).

Nicht praxistauglich:

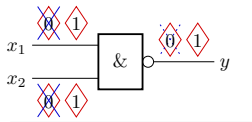
- 1 Toggle-Fehler,
[Korrel. Modellfehler- und Fehlernachweis gering]
- 2 Kurzschlüsse, Unterbrechungen,
[erfordert elektrische Fehlersim., Aufwand/Nutzen]
- 3 Mehrfachfehler, Pfadverzögerungsfehler, ...
[Überproportionale Zunahme der Modellfehleranzahl]

Bei Übernahme der fehlerorientierte Testauswahl für andere Systemtypen ist auch die Frage interessant, was für Fehlermodelle sich für Schaltkreise nicht bewährt haben und warum.

Haftfehler

Für jeden Gatteranschluss wird unterstellt:

- ein sa0 (stuck-at-0) Fehler
- ein sa1 (stuck-at-1) Fehler



- ◇ 0 sa0-Modellfehler
- ◇ 1 sa1-Modellfehler
- × identisch nachweisbar
- ⋯ implizit nachweisbar

x_2	x_1	$\overline{x_2 \wedge x_1}$	sa0(x_1)	sa1(x_1)	sa0(x_2)	sa1(x_2)	sa0(y)	sa1(y)
0	0	1	1	1	1	1	0	1
0	1	1	1	1	1	0	0	1
1	0	1	1	0	1	1	0	1
1	1	0	1	0	1	0	0	1

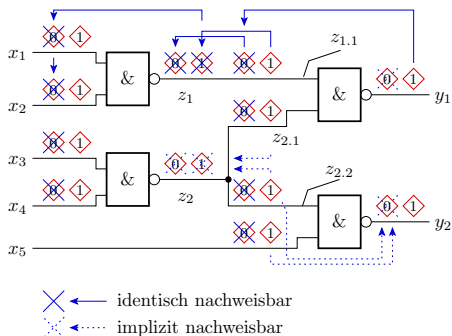
Nachweisidentität (gleiche Nachweismenge)

⋯→ Nachweisimplikation

■ zugehörige Eingabe ist Element der Nachweismenge

Zusammenfassung identisch nachweisbarer Fehler. Optionale Streichung redundanter und implizit nachweisbarer Modellfehler.

Identisch und implizit nachweisbare Fehler im Schaltungsverbund

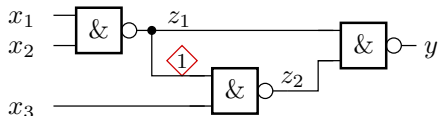


Größe der Anfangsfehlermenge:	24
Anzahl der nicht identisch nachweisbaren Fehler:	14
ohne implizit nachgewiesene Fehler:	10

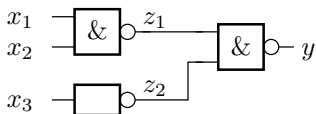
Mengen von identisch nachweisbaren Fehlern	Nachweis impliziert durch
1 sa0(x ₁), sa0(x ₂), sal(z ₁), sal(z _{1.1})	
2 sal(x ₁)	
3 sal(x ₂)	
4 sa0(x ₃), sa0(x ₄), sal(z ₂)	9, 12
5 sal(x ₃)	
6 sal(x ₄)	
7 sa0(z ₂)	5, 6, 8, 11
8 sa0(z ₁), sa0(z _{1.1}), sa0(z _{2.1}), sal(y ₁)	2, 3
9 sal(z _{2.1})	
10 sa0(y ₁)	1, 9
11 sa0(z _{2.2}), sa0(x ₅), sal(y ₂)	
12 sal(z _{2.2})	
13 sal(x ₅)	
14 sa0(y ₂)	12, 13

Redundante Fehler

redundanter Haftfehler

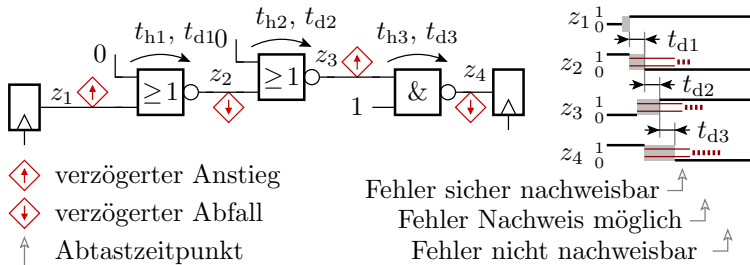


vereinfachte Schaltung



- Bei $z_1 = 0$ ist der Fehler an y nicht beobachtbar und mit $z_1 = 1$ wird der Fehler nicht angeregt. Gatteranschluss mit sa1-Fehler kann mit »1« verbunden werden, ohne dass sich die Funktion ändert. Möglichkeit der Schaltungsvereinfachung.
- Der Nachweis der Redundanz kann schwieriger sein als die Suche eines Tests. Falls nicht erkannt, werden redundante Fehler als nicht erkannt gezählt und verringern die berechnete gegenüber der tatsächlichen Modellfehlerüberdeckung..
- Fehlermodelle, die viele redundante Fehler erzeugen sind zur Schätzung der tatsächlichen Fehlerüberdeckung ungeeignet.

Gatterverzögerungsfehler

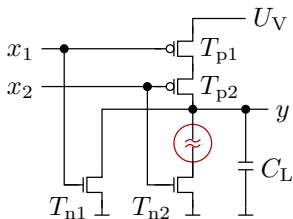


Statt der Haftfehler, Annahme an allen Gatteranschlüssen

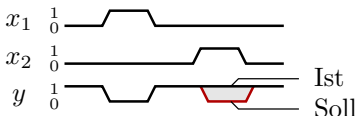
- eines Slow-To-Raise- (verzögerter Signalanstieg) und
- eines Slow-To-Fall- (verzögerter Signalabfall)

Streichen identisch nachweisbarere, redundanter und optional implizit nachweisbarer Modellefehler. Nachweis über einen Signalpfad durch die Schaltung, der an einem Abtastregister beginnt und endet durch eine Folge aus Initialisierungs- und Testeingabe.

Stuck-Open-Fehler

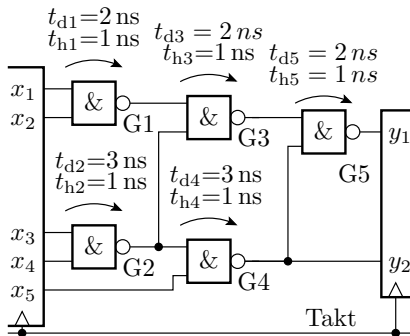


Nachweis (2-Pattern-Test)



- Stuck-Open-Fehler bewirken, dass der Gatterausgang bei lokaler Anregung hochohmig ist.
- Fehlerwirkung wie lange Zusatzverzögerung.
- Nachweis wie Verzögerungsfehler durch 2-Pattern-Tests.

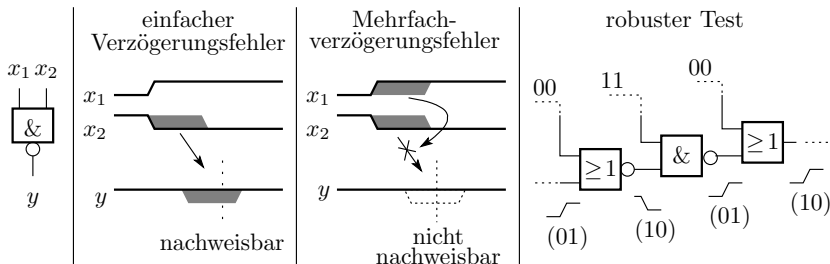
Nachweiswahrscheinlichkeit und Pfadlänge



Pfade	$\sum t_{h.i}$	$\sum t_{d.i}$
G1-G3-G5	3 ns	6 ns
G2-G3-G5	3 ns	7 ns
G2-G4-G5	3 ns	8 ns
G2-G4	2 ns	6 ns
G4-G5	2 ns	5 ns
G4	1 ns	3 ns

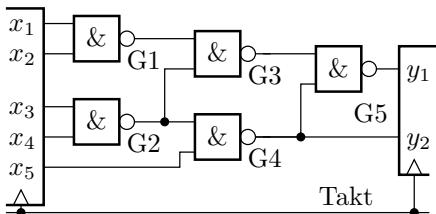
- Die minimal erkennbare Zusatzverzögerung ist die Differenz aus Taktperiode und Soll-Verzögerung.
- Je länger die Sollverzögerung, desto höher die Wahrscheinlichkeit, fehlerverursachte Zusatzverzögerungen zu erkennen.
- Tests über die Pfade mit den längsten Sollverzögerungen erkennen alle nachweisbaren Einzelverzögerungsfehler.

Mehrfachfehler



- Es sind Mehrfachfehler konstruierbar, die bei bestimmten Tests gegenseitig maskieren.
- Robuster Test: Ausschluss der gegenseitigen Maskierung, durch max. eine Signaländerung an den Eingängen jedes Gatters je Testschritt.
- Zusatzattribute wie »minimal erkennbare Zusatzverzögerung« und »Robustheit« bei der Fehlersimulation / Testberechnung mit berechnenbar bzw. berücksichtigbar.

Pfadverzögerungsfehler



Pfade:

G1-G3-G5

G2-G3-G5

G2-G4-G5

G2-G4

G4-G5

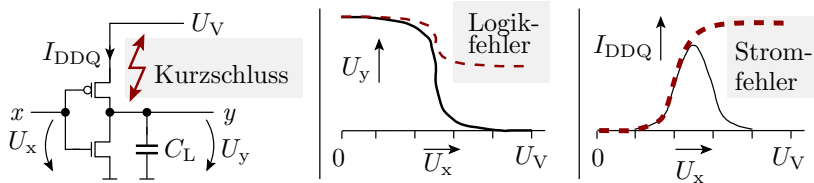
G4

Fehlerannahme »Slow-To-Rise« / »Slow-To-Fall« für alle Schaltungspfade, statt Gatter. Nicht zielführend weil:

- Pfadsanzahl wächst überproportional mit Systemgröße.
- Wegen Nachweisabhängigkeiten sind zu viele Modellfehler je Systemgröße kein Gewinn für die Vorhersagbarkeit der *FC*.
- Risiko redundanter Fehler durch nicht sensibilisierbarer Pfade.

Gatterverzögerungsfehler versprechen mit deutlich weniger Aufwand bessere Vorhersagen für die *FC*.

Ruhestromüberwachung

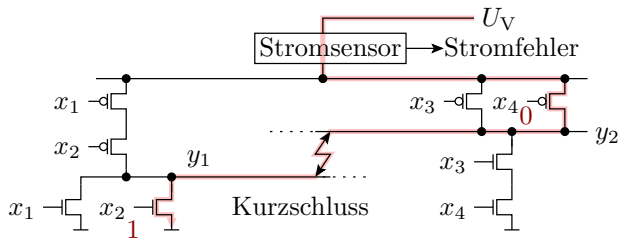


In einer CMOS-Schaltung ist der Gatterausgang nur entweder über NMOS-Transistoren mit 0 (Masse) oder über PMOS-Transistoren mit 1 (Versorgungsspannung) verbunden. Nach jedem Schaltvorgang klingt der Strom auf einen sehr kleinen Wert ab. Die Hälfte der zu erwartenden lokalen Schaltkreisfehler sind Kurzschlüssen, nicht richtig ausschaltende Transistoren, ... die, wenn sie zur Wirkung kommen, ein messbaren Ruhestrom I_{DDQ} verursachen.

I_{DDQ} -Überwachung erkennt Defekte auch ohne Beobachtungspfad zu einem Ausgang.

[messtechnische Probleme]

Kurzschlussnachweis über den Ruhestrom



Nachweis über statische Stromaufnahme, wenn $y_1 \neq y_2$

- Vorteile I_{DDQ} -Test: Einfachere logische Nachweisbedingungen, einfachere fehlerorientierte Testsuche, viel kürzere Zufallstetests für dieselbe Fehlerüberdeckung.
- Probleme I_{DDQ} -Test: Unterscheidung von zulässigem und überhöhtem Ruhestrom funktioniert nur bis einige tausend Gatter. Integrierte Stromsensoren, ...

Toggle-Test

Alle Signale sind während des Test mindestens einmal auf »0« und einmal auf »1« zu steuern. Als Fehlermodell:

- für alle Leitungen, Annahme von zwei Modellfehlern
 - keine 0 einstellbar,
 - keine 1 einstellbar.

Wahrscheinlichkeit, dass ein Toggle-Test einen Leitungshafffehler nachweist ist die Wahrscheinlichkeit der Beobachtbarkeit der lokalen Verfälschung.

Einen ermittelte Toggle-Überdeckung erlaubt kaum Rückschlüsse auf die Haft- oder tatsächliche Fehlerüberdeckung.



Zellenfehlermodell

- Definition einer Testvorschrift für jede Zelle (jedes Teilsystem).
- Durchführbarer Zelltestschritt zählt als nachweisbarer Fehler, z.B. für Gatter jede Eingabe der Wertetabelle.
- Für freistrukturierte Schaltungen sind typisch 10% bis 15% der Zellenfehler redundant, deshalb ungeeignet.

Soll-Funktion			Zellenfehler			
x_1	x_0	$x_1 \vee x_0$	F00	F01	F10	F11
0	0	0	1	0	0	0
0	1	1	1	0	1	1
1	0	1	1	1	0	1
1	1	1	1	1	1	0

Fehlermodelle mit unbekanntem hohen Anteil redundanter Fehler ungeeignet für Abschätzung großer Fehlerüberdeckungswerte.

- Bewährt für die Konstruktion von regelmäßig strukturierten Schaltungen wie Speicher, Addierer und Multiplizierer für 100% Zelltestüberdeckung.

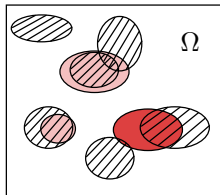


Nachweisbeziehungen

Nachweisbeziehungen Modellfehler \Leftrightarrow Fehler

Etablierte Fehlermodelle für digitale Schaltkreise:

- Haftfehler, Gatterverzögerungsfehler [+IDDQ-Test] und
- Zellenfehler (regelmäßig strukturierte DIC, RAM, ...).



- Ω Menge aller Testeingabewerte
- Eingabewerte, die den Fehler nachweisen
- Eingabewerte, die den Fehler eventuell (unter Zusatzbedingungen) nachweisen
- Nachweismenge ähnlich nachweisbarer Modellfehler

Ein Fehlermodelle generieren eine große Menge von Modellfehlern, in der für möglichst für alle zu erwartenden Fehler i ähnlich nachweisbare ähnliche nachweisbare Fehler j enthalten sind.

Ursachen für Nachweisähnlichkeit sind übereinstimmende Anregungs- Beobachtungs- und lokale Nachweisbedingung.



Haftfehler \Leftrightarrow Schalkreisfehlerüberdeckung

Die Wahrscheinlichkeit $p_{FT} \approx DL$, dass ein als gut befundenen Schaltkreise nach Ersatz aller erkennbar fehlerhaften Schaltkreise fehlerhaft ist, betrug nach Foliensatz F2, Abschnitt 3.2 Ersatziteration:

$$p_{FT} = \frac{p_F \cdot (1 - p_E)}{1 - p_F \cdot p_E}$$

($p_F \approx 1 - Y$ — Wahrscheinlichkeit, Schaltkreis vor Aussortieren defekt;
 $p_E \approx FC$ — Erkennungswahrscheinlichkeit).

	$FC = 90\%$	$FC = 99\%$	$FC = 99,9\%$
$Y = 30\%$	18,9%	2,28%	2328 dpm
$Y = 50\%$	9,09%	9901 dpm	999 dpm
$Y = 80\%$	2,43%	2494 dpm	250 dpm

Laut Literatur: Fehleranteil getesteter Schaltkreise 200 ... 500 dpm,
 $Y = 30\%..80\%$, Haftfehlerüberdeckung 95% bis 99%.

Minderung der Anzahl der nicht nachgewiesenen Fehler gegenüber Haftfehlern in der Größenordnung $C_{SA} = \frac{1-FC}{1-FC_{SA}} \approx 0,1$ glaubhaft?

Fehlerorientierte Testsuche

Für jeden Fehler i gibt es $\#MF_i$ ähnlich nachweisbare Modellfehler j , für die jeder gefundene Test Fehler i mit p_{ij} nachweist. Wenn ein Test gefunden wird, werden $m - 1$ weitere Tests gesucht und auch gefunden.

A_j Tests für Modellfehler j gefunden:

$$\mathbb{P}(A_j) = FC_M$$

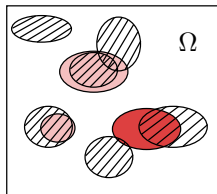
B_{ij} m Tests für Modellfehler j weisen Fehler i nach:

$$\mathbb{P}(B_{ij}) = 1 - (1 - p_{ij})^m$$

N_i Nachweis von Fehler i :

$$N_i = \bigcup_{j=1}^{\#MF} (A_j \cap B_{ij}) = \overline{\bigcap_{j=1}^{\#MF} \overline{(A_j \cap B_{ij})}}$$

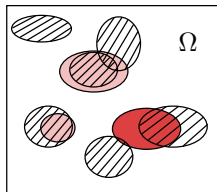
$$p_i = \mathbb{P}(N_i) = 1 - \prod_{j=1}^{\#MF} (1 - (FC_M \cdot (1 - (1 - p_{ij})^m)))$$



Zu erwartende Fehlerüberdeckung

$$p_i = 1 - \prod_{j=1}^{\#MF} (1 - (FC_M \cdot (1 - (1 - p_{ij})^m)))$$

$$\mathbb{E}[FC] = \frac{1}{\#F} \cdot \sum_{i=1}^{\#F} p_i$$



Modellrechnung mit $p_{ij} = p$ und im Mittel 3 bis 5
ähnlich nachweisbare Modellfehler je Fehler:

$\#MF$	1	2	3	4	5	6	7
$\mathbb{P}[X = \#MF]$	1%	8%	20%	35%	23%	10%	3%

$$\mathbb{E}[FC] = \sum_{\#MF=1}^{\#MF_{\max}} \mathbb{P}[X = \#MF] \cdot (1 - (1 - (FC_M \cdot (1 - (1 - p)^m)))^{\#MF})$$

FC_M	95%	95%	95%	99%	99%	99%
p	10%	25%	50%	10%	25%	50%
$\mathbb{E}[FC], m = 1$	33%	65,5%	90,6%	34%	67,2%	91,7%

Modellrechnung erwartete FC

$$\mathbb{E}[FC] = \sum_{\#MF=1}^{\#MF_{\max}} \mathbb{P}[X = \#MF] \cdot \left(1 - (1 - (FC_M \cdot (1 - (1 - p)^m)))^{\#MF}\right)$$

FC_M	95%	95%	95%	99%	99%	99%
p	10%	25%	50%	10%	25%	50%
$\mathbb{E}[FC], m = 1$	33%	65,5%	90,6%	34%	67,2%	91,7%
$\mathbb{E}[FC], m = 4$	78,1%	97,1%	99,8%	79,6%	97,7%	99,9%

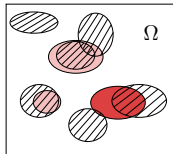
Die These $C_{SA} = \frac{1-FC}{1-FC_{SA}} \approx 0,1$ wäre nur glaubwürdig, wenn

- Tests für Haftfehler j die Fehler i mit $p_{ij} \approx 50\%$ nachweisen,
- zu jedem Fehler i $\#MF \approx 4 \dots 5$ Modellfehlern ähnlich nachweisbar sind und
- $m \approx 4$ Tests je Modellfehler gesucht werden.

Fehlerüberdeckung bei zufälliger Auswahl

Zu erwartende Fehler- und Modellfehlerüberdeckung bei pareto-verteilter Nachweislänge :

$$\begin{aligned} \mathbb{E}[FC(n)] &= 1 - C_{SA} \cdot (1 - \mathbb{E}[FC_{MF}(n)]) \\ &= 1 - c_{SA}^{-k} \cdot (1 - \mathbb{E}[FC_{MF}(n)]) \end{aligned}$$



k – Formfaktor der Pareto-Verteilung; n – Testsatzlänge;

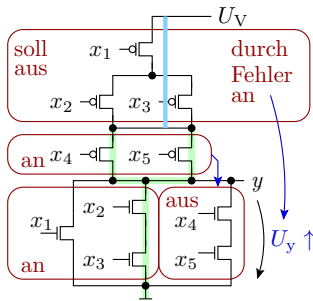
$c_{SA} = \frac{n_{[eff]}}{n_T} = \frac{\bar{\zeta}_{SA}}{\bar{\zeta}}$ – Verhältnis der mittleren FF-Rate Modellfehler $\bar{\zeta}_{SA}$ und mittleren FF-Rate realer Fehler $\bar{\zeta}$; C_{SA} – Verringerungsfaktor für die Anzahl der nicht nachweisbaren Fehler.

Die These $C_{SA} \approx 0,1$ ist glaubhaft, wenn:

$$\bar{\zeta} \approx c_{SA} \cdot \bar{\zeta}_{SA}; \quad n_{[eff]} \approx c_{SA} \cdot n_T \quad \text{mit} \quad c_{SA} = 20 \dots 10^5$$

gezeigt werden kann (vergl. Foliensatz F3, Abschn. 5 Test & Zuverlässigkeit).

Kurzschlussnachweis mit Haftfehlertests



x_1	1	1	1	1	0	1	1	1	1	0	1	1	1	1	0
x_2	0	0	1	1	1	0	0	1	1	1	0	0	1	1	1
x_3	0	1	0	1	1	0	1	0	1	1	0	1	0	1	1
x_4	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
x_5	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
Kurzschl.	x	x	x	x	+	x	x	x	x	x	x	x	x	x	x
sa0(x_1)	x					x					x				
sa0(x_2)					x					x					x

Kurzschluss

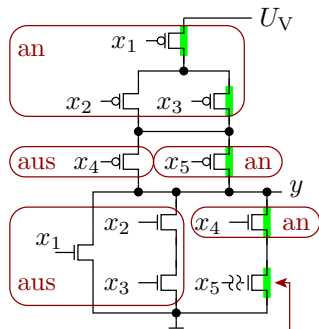
Einschaltbed. max.
Verfälschung von y

- Nachweis $\text{sa0}(x_2)^2$: $2 \times$ gute und $1 \times$ sicherste Nachweisbed.
- Nachweis $\text{sa0}(x_1)$: mehrere unsichere Nachweisbed.
- Slow-to-fall-Fehler für x_1 bis x_3 versprechen höherer p_{ij} .

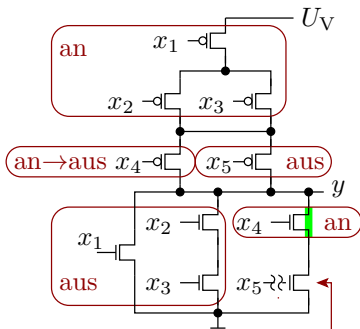
These $C_{\text{SA}} \approx 0,1$ glaubhaft? $\#MF \approx 2?$; $p_{ij} \approx 50\% \sqrt{\cdot}$, $c_{\text{SA}} \approx 1?$

$^2\text{sa0}(x_3)$ ist identisch nachweisbar.

»Offenes Gate« mit Haftfehlertests



$sa1(x_5)$: soll aus, ist an

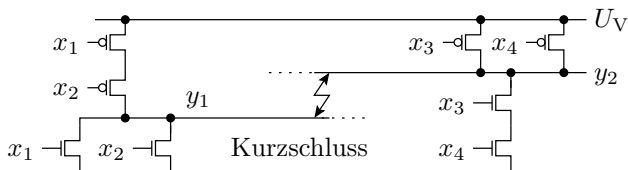


$sa0(x_5)$: soll an, ist aus

- Nachweis $sa1(x_5)$ für $F = 1$, am sichersten wenn $x_2 = \bar{x}_2$.
- Nachweis $sa0(x_5)$ für $F = 0$, wenn y_{soll} von 0 nach 1 schaltet.

These $C_{SA} \approx 0,1$ glaubhaft? $\#MF \approx 2?$; $p_{ij} \approx 50\% \sqrt{}$, $c_{SA} \approx 1?$

Kurzschluss zweier Gatterausgänge



Bedingungen für den logischen Nachweis:

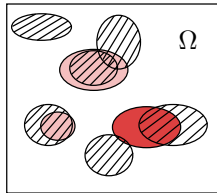
- y_1 und y_2 müssen sich im fehlerfreien Fall unterscheiden.
- Je nach Fehlerwirkung müssen y_1 oder y_2 dabei beobachtbar sein.

Ähnlich nachweisbare Haftfehler:

- $sa0(x_1)$, $sa0(x_2)$ wenn $y_2 = 1$ ist und sich durchsetzt
- $sa1(x_1)$, $sa1(x_2)$ wenn $y_2 = 0$ ist und sich durchsetzt
- $sa0(x_3)$, $sa0(x_4)$ wenn $y_1 = 1$ ist und sich durchsetzt
- $sa1(x_3)$, $sa3(x_4)$ wenn $y_1 = 0$ ist und sich durchsetzt

These $C_{SA} \approx 0,1$ glaubhaft? $\#MF \approx 8\sqrt{}$; $p_{ij} \approx 25\%\sqrt{}$, $c_{SA} \approx 2?$

Zusammenfassung



- Kurzschlüsse, Unterbrechungen und Transistordefekte verursachen kein eindeutiges Fehlverhalten.
- Für jeden der untersuchten Fehler i erzeugt das Haftfehlermodell ähnlich nachweisbare Fehler j , die mit hoher Wahrscheinlichkeit p_{ij} auch Fehler i nachweisen.
- Ganz grob sind die Größenordnungen der Ähnlichkeitsparameter für Haftfehler: $\#MF \approx 1 \dots 10$, $p_{ij} = 10\% \dots 50\%$ und $c_{SA} \approx 0,5 \dots 10$.
- Die These $C_{SA} \approx 0,1$ ist für zufällige Testauswahl nicht glaubhaft und für gezielte Testauswahl nur, wenn für jeden Modellfehler mehrere Tests gesucht werden.

Es gibt noch viel zu erforschen.



Testsuche



Testsuche

Zusammenstellen der Modellfehlermenge
Wiederhole, bis genug Modellfehler nachgewiesen werden
geziele, manuelle, zufällige Auswahl weiterer Testeingaben
Fehlersimulation, abhaken der nachweisbaren Modellfehler

In der innersten Schleife:

- solange ≥ 1 Nachweis aller N_{\min} Tests: zufällige Auswahl
- danach: gezielte Suche
- Abhaken aller (weiteren) nachweisbaren Modellfehler.

Problematisch sind redundante Fehler, genauer der Nachweis, dass es für einen Modellfehler keine Nachweismöglichkeit gibt und er folglich nicht als Modellfehler zählt.

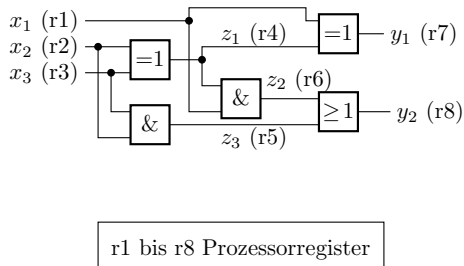
[Umgang mit Zufallstests, die keinen neuen Modellfehler nachweisen]
[mehrere Tests je Modellfehler]



Fehlersimulation

Simulation von Haftfehlern

Schaltung eines Volladdierers



Programm für die Gutsimulation

```

1 lade x1 in Register r1
2 lade x2 in Register r2
3 lade x3 in Register r3
4 r4 = r2 xor r3
5 speichere Inhalt r4 in z1
6 r5 = r2 and r3
7 speichere Inhalt r5 in z3
8 r6 = r1 and r4
9 speichere Inhalt r6 in z2
10 r7 = r1 xor r4
11 speichere Inhalt r7 in y1
12 r8 = r5 or r6
13 speichere Inhalt r8 in y2
    
```

- Jede zweistellige Logikoperation ist ein Maschinenbefehl.
- In jeder der 8, 16, 32 oder 64 Bits der Operanden kann ein anderer Testfall oder ein anderer Fehler simuliert werden.

[Prozeduren für 64 Fehler 1 Test, Compile+Link, ca. 1/3 x 64 Logik-Op. je Prozessortakt]



Aufwandsabschätzung am Beispiel

- Schaltungsgröße: 10^4 Gatter
- Anzahl der Testschritte / Testeingaben: 10^4
- Anzahl der Modellfehler: 10^4
- Simulationsaufwand je Gatter: 10 ns

Rechenaufwand:

- wenn jeder Fehler mit allen Testeingaben simuliert wird ohne bitparallele Simulation: 10^4 s, ca. 3 h.
- Wenn mit jedem der 32 bzw. 64 Bits ein anderer Fehler simuliert wird, nur 6 bzw. 3 Minuten.
- Wenn bereits nachgewiesene Modellfehler nicht weiter mit simuliert werden, unter 1 Minute.

[Hauptaufwand redundante Fehler]

[10^4 Tests pro min; Zufallstest 10^8 Tests \Rightarrow 1 Woche Simulationszeit]



D-Algorithmus

D (Discrepancy)-Kalkül von Roth

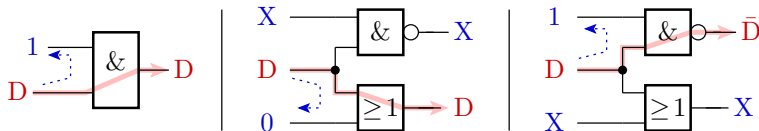
Erweiterung der Logikwerte um 3 Pseudo-Werte³:

D 0 wenn unverfälscht, 1 wenn verfälscht.

\bar{D} 1 wenn unverfälscht, 0 wenn verfälscht.

X Signalwert ist ungültig oder für den Fehlernachweis ohne Bedeutung.

Regeln für die Sensibilisierung eines Beobachtungspfades:



³W. Daehn: Testverfahren in der Mikroelektronik: Methoden und Werkzeuge. Springer 1997.

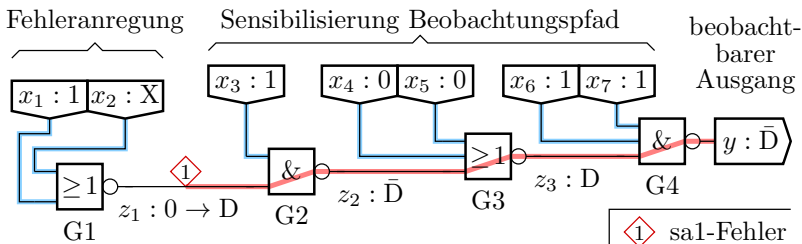
Testsuche für Haftfehler

Ein Haftfehler unterstellt für den Fehlerort, dass der Wert

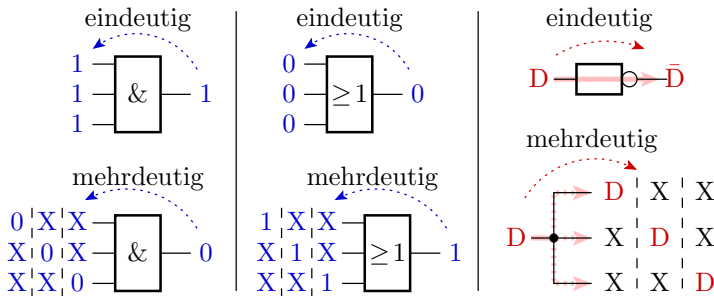
- entweder ständig 0 (sa0) oder
- ständig 1 ist (sa1) ist.

Ausgehend vom Fehlerort werden Eingaben gesucht,

- für die der Wert am Fehlerort invertiert wird und
- bei denen die Invertierung am Fehlerort an einem Ausgang beobachtbar ist.



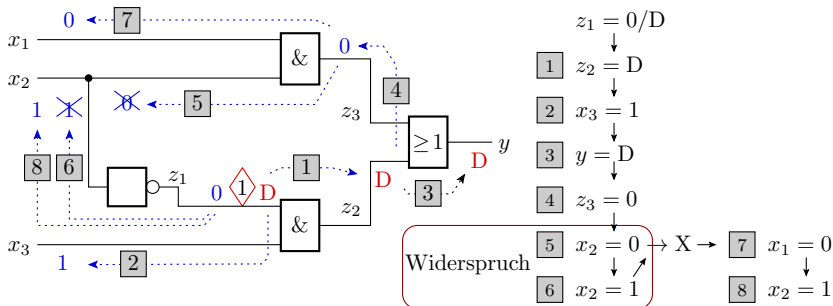
Ein- und mehrdeutige Pfade



Ausgehend vom Fehlerort:

- Festlegen von Werten zur Weiterführung des Beobachtungs- oder eines Steuerpfads.
- Bei Widersprüchen zurück zu letzten Möglichkeit einer Alternativentscheidung, ... \Rightarrow Baumsuche

[kein Test gefunden und keine weiter Alternative \Rightarrow Fehler redundant]



Baumsuche:

- Bei der Wertefestlegung können Widersprüche auftreten.
- Zurück zur letzten mehrdeutigen Entscheidung.
- Keine Lösung nach Durchmusterung des gesamten Baums. ⇒ Fehler nicht nachweisbar

	x_3	x_2	x_1	z_3	z_2	z_1	y
	X	X	X	X	X	0D	X
1	X	X	X	X	D	0D	X
2	1	X	X	X	D	0D	X
3	1	X	X	X	D	0D	D
4	1	X	X	0	D	0D	D
5	1	0	X	0	D	0D	D
6	1	X	X	0	D	0D	D
7	1	X	0	0	D	0D	D
8	1	1	0	0	D	0D	D

[Suchreihenfolge verbesserbar, z.B. eindeutige Wertefestlegungen vor mehrdeutigen]



Optimierung Suchalgorithmus

Erfolgsrate der Testberechnung:

- Anteil der Fehler, für die ein Test gefunden oder für die der Beweis »nicht nachweisbar« erbracht wird.
-

- Die Testsuche für einen Fehler kann hunderte von Wertefestlegungen beinhalten.
 - Der Suchraum wächst exponentiell mit der Anzahl der mehrdeutigen Festlegungen. Suchräume der Größen $> 2^{30...40}$ nicht mehr vollständig durchsuchbar.
 - Abbruch der Suche nach einer bestimmten Rechenzeit.
-

Heuristiken:

- Frühe Erkennung von Widersprüchen (Äste im Suchbaum abschneiden),
- Suchraumbegrenzung und
- gute Suchraumstrukturierung.

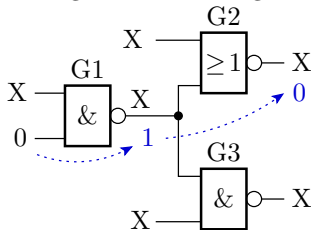


Implikationstest

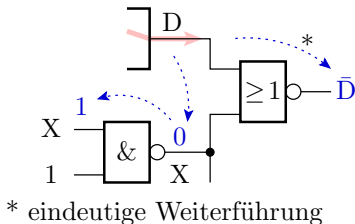
Implikationstest (Widerspruchsfürerkennung)

- Aus den berechneten Wertefestlegungen alle eindeutig folgenden Werte berechnen.

Implikation in
Signalflussrichtung

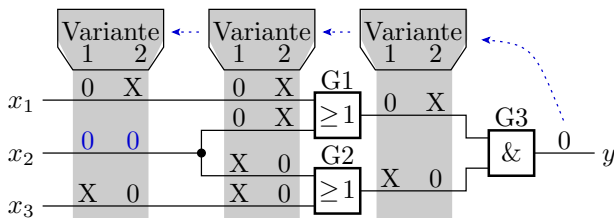


D-Pfad- und Rückwärtsimplikation



- Mindert die Entscheidungsbaumtiefe.

- Rückwärtsimplikation über mehrere Gatterebenen:



- Für $y = 0$ gibt es zwei Einstellmöglichkeiten.
- Für beide Möglichkeiten muss $x_2 = 0$ sein.
- Das Erkennen von Implikationen dieser Art mindert die Backtracking-Häufigkeit um bis zu 80 %.

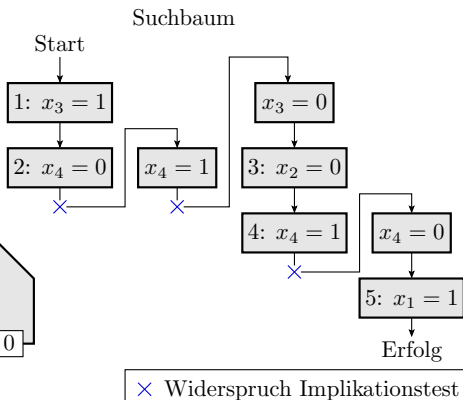
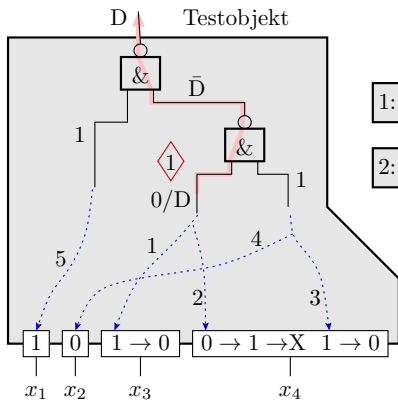


Suchraumstrukturierung



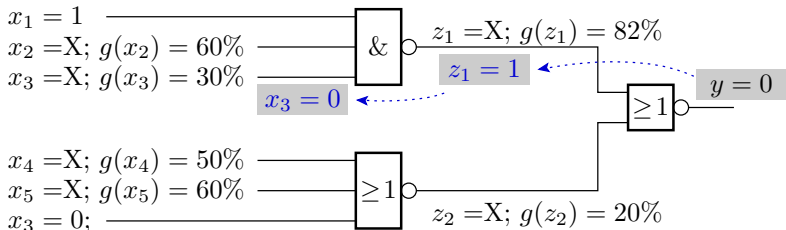
Suchraumbegrenzung

- Der D-Algorithmus baut den Suchbaum über alle mehrdeutigen Wertefestlegungen auf.
- Nur die Schaltungseingänge können unabhängig voneinander alle Wertevariationen annehmen.
- Es genügt, den Suchbaum mit den Eingabewertefestlegungen aufzubauen.
- Begrenzt Suchraum auf $2^{\#E}$ ($\#E$ – Eingangsanzahl). Verringert Rechenaufwand um Zehnerpotenzen.



- Lange Steuerpfade vom Fehlerort und vom D-Pfad zu Eingängen.
- Aufbau des Suchbaums über Eingangssignale.
- Wenn Implikationstest-Widerspruch, letzte Eingabefestlegung invertieren.

Geschätzte Erfolgswahrscheinlichkeiten



$[g(\dots)$ Signalgewicht, Auftrittshäufigkeit einer 1

- Schätzen der Signalwichtungen ⁴ $g(x_i)$ über eine kurze Simulation mit Zufallswerten oder analytisch.
- Wahl der Steuerwerte / Beobachtungspfade, die mit größerer Wahrscheinlichkeit aktivierbar / sensibilisierbar sind.

⁴Die Wichtung $g(x_i)$ eines Signals x_i ist die Auftrittswahrscheinlichkeit einer »1«:

$$g(x_i) = \mathbb{P}[x_i = 1]$$

(vergl. später Abschn. 4.4 Selbsttest. Fehlerorientierte Wichtung).



Komplexe Funktionsbausteine



Komplexe Funktionsbausteine

- Beschreibung durch Tabellenfunktion (Bsp. Volladdierer):

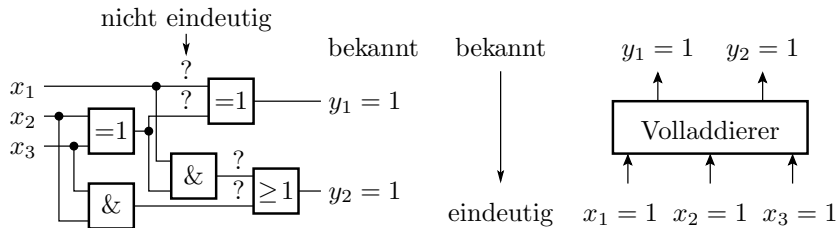
x_2	x_1	x_0	s	c	gegeben	Lösungsmenge
0	0	0	0	0	XXX00	\Rightarrow 00000 (eindeutig)
0	0	1	1	0	01DXX	\Rightarrow 01D \bar{D} D (eindeutig)
0	1	0	1	0		
0	1	1	0	1	1XXXD	\Rightarrow 10D \bar{D} D, 1D0 \bar{D} D (2 Alternativen)
1	0	0	1	0		
1	0	1	0	1	11XX1	\Rightarrow 11111, 11001 (2 Alternativen)
1	1	0	0	1		
1	1	1	1	1		

- Vervollständigung des Vektors der gegebenen Anschlusswerte durch Vergleich mit allen Tabellenzeilen:

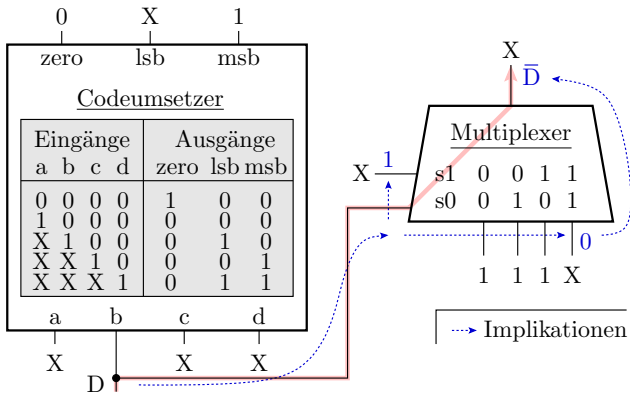
- »1« und »0« passen nur auf »1« und »0«.
- »X« passt immer.
- »D« muss für eine Zeile mit »D=0« und eine Zeile mit »D=1« passen.

[Bei Alternativen (untere Beispiele) Suchbaumverzweigung]

Implikationstest an einem Volladdierer



- An der Gatterbeschreibung eines Volladdierers ist die Implikation $y_1 = y_2 = 1 \Rightarrow x_1 = x_2 = x_3 = 1$ nicht zu erkennen. Lösungsfindung über Baumsuche.
- Bei Zusammenfassung zu einer Tabellenfunktion wird die Lösung bereits bei der Anschlusswertevervollständigung erkannt.



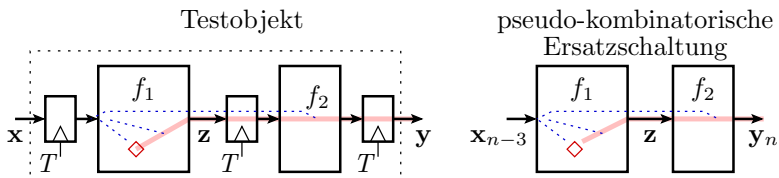
- »lsb« hängt bei »zero=0« und »msb=1« nicht von »b« ab. Eindeutiger D-Pfad über Multiplexer.
- Tabelleneingabewerte »X« (Eingang beeinflusst nicht die Ausgabe) führt zu Tabellen mit $\ll 2^{N_E}$ Tabellenzeilen (N_E – Anzahl der Eingänge).



Sequentielle Schaltungen

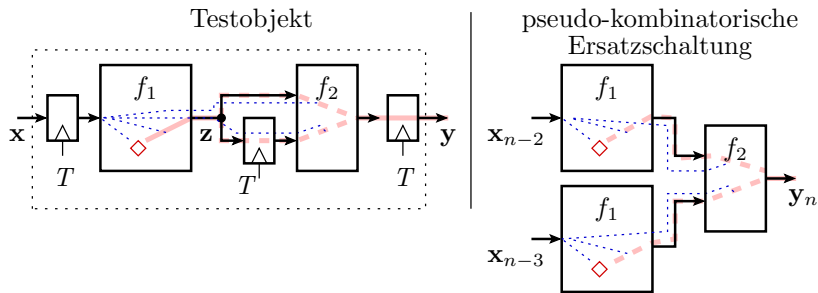
Pseudo-kombinatorische Ersatzschaltung

Schaltungen mit Speicherelementen werden für die Testsuche zu einer pseudo-kombinatorischen Ersatzschaltung aufgerollt. Abtastregister in einem geradlinigen Berechnungsfluss werden weggelassen:



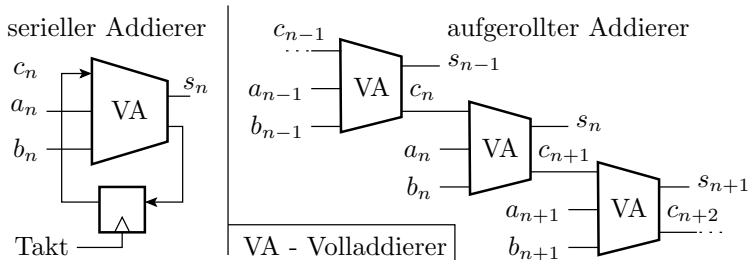
- Testberechnung wie für eine kombinatorische Schaltung.
- Die Verzögerungen der Ausgabe gegenüber der Eingabe wird erst beim Zusammensetzen der Testeingaben und Sollausgaben berücksichtigt.

Verarbeitung in mehreren Zeitebenen



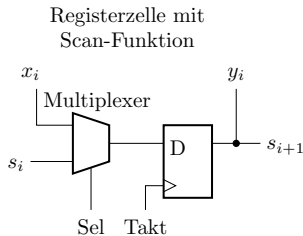
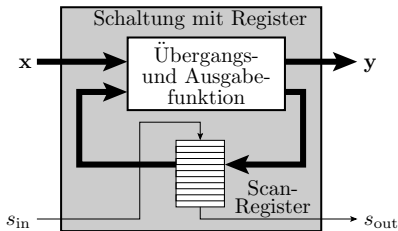
- Mehrere Kopien gleicher Schaltungsteile in der pseudo-kombinatorischen Ersatzschaltung.
- Der eingebaute Haftfehler ist in jeder Kopie der Teilschaltung.
- Für jeden Fehler wird eine Folge von Testeingaben für mehrere Zeitschritte berechnet (Mehr-Pattern-Test).

Schaltungen mit Rückführung



- Pseudo-kombinatorischen Ersatzschaltung mit endlos vielen Kopien der Übergangsfunktion.
- Längenbegrenzung der Steuer- und Beobachtungspfade.
- Alternative: Lese- und Schreibzugriff auf Zwischenergebnisse und -zustände, z.B. durch Verschalten der internen Speicherzeichen zu einem Scan-Register (siehe nächste Folie).

Scan-Verfahren



Lese- und Schreibzugriff während des Tests durch Umschalten des Zustandsspeicher in ein Schieberegister.

- Mindestschaltungsaufwand ein Multiplexer je Speicherzelle.
- Ablauf eines Testschritts: r Schiebeschritte zum Beschreiben des Zustandsspeichers, Testschritt, r Schiebeschritte zum Lesen (und Überschreiben) des Zustandsspeichers.

[2-Pattern-Test, Delay-Fehler, Erweiterung wie bei Boundary-Scan um »update«]

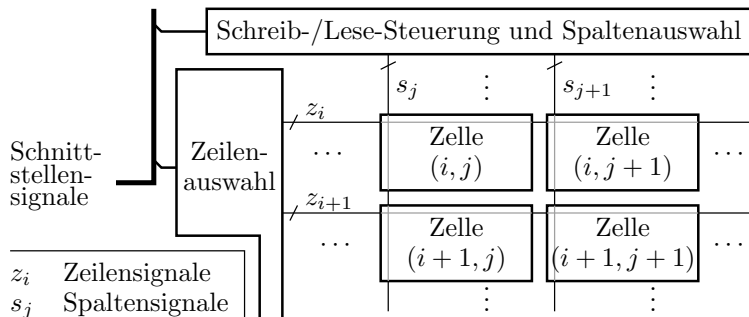


Speichertest



Blockspeicher

Große Speicher besteht im Kern aus einer regelmäßigen 2D-Anordnung von flächenminimierten Speicherzellen umgeben von der Zeilen- und Spaltenauswahlschaltung. Die Grundfunktionen (nur lesbar, beschreib- und lesbar, ...) hängen von der Zellenfunktion ab.



[Zellenfehler, c-Test]



Zellenfehler für RAM

beteiligte Zellen	Name	Definition	Fälle	Testfolge für den Nachweis
1	Haftfehler	Wert der Speicherzelle ist nicht setzbar	stuck-at-0 stuck-at-1	$W(i)1, R(i)1$ $W(i)0, R(i)0$
	Übergangsfehler	Wert der Speicherzelle i ist nur in einer Richtung änderbar	kein Übergang $1 \rightarrow 0$ $0 \rightarrow 1$	$W(i)1, R(i)1, W(i)0, R(i)0$ $W(i)0, R(i)0, W(i)1, R(i)1$
	Stuck-open-Fehler	kein Zugriff auf Speicherzelle i (Ausgabe des Wertes der vorherigen Leseoperation)		$W(i)0, R_1(j), R(i)0, W(i)1,$ $R_0(j), R(i)1$
	zerstörendes Lesen	Inhalt von Speicherzelle i wird beim Lesen verändert	$R(i) \Rightarrow C(i) = \overline{C(i)}$	$W(i)0, R(i)0, R(i)0$ $W(i)1, R(i)1, R(i)1$
2	Kopplung Typ 1	Veränderung des Inhalts von Zelle i bestimmt Zustand in Zelle j	$W(i)0 \Rightarrow C(j) = 0$ $W(i)0 \Rightarrow C(j) = 1$ $W(i)1 \Rightarrow C(j) = 0$ $W(i)1 \Rightarrow C(j) = 1$	$W(j)0, W(i)0, R(j)0,$ $W(i)1, R(j)0$ $W(j)1, W(i)0, R(j)1,$ $W(i)1, R(j)1$
	Kopplung Typ 2	Veränderung des Inhalts von Zelle i bewirkt eine Änderung in Zelle j	$C(i) = \overline{\overline{C(i)}} \Rightarrow$ $C(j) = \overline{C(j)}$	$W(j)0, W(i)0, R(j)0, W(i)1,$ $R(j)0, W(i)0, R(j)0$ $W(j)1, W(i)0, R(j)1, W(i)1,$ $R(j)1, W(i)0, R(j)1$

 $W(i)0$ Schreibe in Zelle i eine 0 $W(i)1$ Schreibe in Zelle i eine 1 $R(j)$ Lese eine beliebige andere Zelle $C(\dots)$ Inhalt Zelle ... $R(i)0$ Lese Inhalt Zelle i und vergleiche mit Sollwert 0 $R(i)1$ Lese Inhalt Zelle i und vergleiche mit Sollwert 1 $R_0(j)$ Lese eine andere Zelle, in der 0 steht $R_1(j)$ Lese eine andere Zelle, in der 1 steht



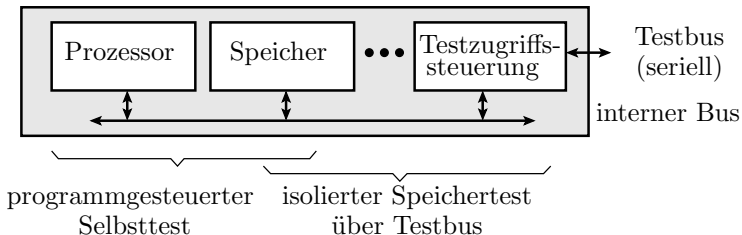
Beispiel Marching Test

Adresse i	Initialisierung	March 1	March 2	March 3	
0	$W(i)0$	$R(i)0, W(i)1$	$R(i)1, W(i)0$	$R(i)0, W(i)1$	
1	$W(i)0$	$R(i)0, W(i)1$	$R(i)1, W(i)0$	$R(i)0, W(i)1$	
2	$W(i)0$	$R(i)0, W(i)1$	$R(i)1, W(i)0$	$R(i)0, W(i)1$	
\vdots	\vdots	\vdots	\vdots	\vdots	
$N - 1$	$W(i)0$	$R(i)0, W(i)1$	$R(i)1, W(i)0$	$R(i)0, W(i)1$	
	March 4		March 1a		March 2a
0	$R(i)1, W(i)0$	Wartezeit	$R(i)0, W(i)1$	Wartezeit	$R(i)1$
1	$R(i)1, W(i)0$		$R(i)0, W(i)1$		$R(i)1$
2	$R(i)1, W(i)0$		$R(i)0, W(i)1$		$R(i)1$
\vdots	\vdots		\vdots		\vdots
$N - 1$	$R(i)1, W(i)0$		$R(i)0, W(i)1$		$R(i)1$

Mehrfaches Durchwandern des Speichers in unterschiedlicher Reihenfolge mit der Operationsfolge Zelle Lesen, Wert kontrollieren und inversen Wert zurückschreiben.

[Umkehrung Adressierungsreihenfolge \Rightarrow alle Überkopplung; Datenhaltetest]

Test eingebetteter Blockspeicher



Eingebettete Blockspeicher werden vorzugsweise isoliert von ihrer Schaltungsumgebung getestet:

- über herausgezogenen Bussignale,
- über den Testbus oder
- programmgesteuert vom Prozessor als eingebauter Selbsttest (BIST – Built-In Self-Test).

[Ende 15. Vorlesung]



Selbsttest



Selbsttest

[engl. BIST, IC-Testproblem begrenzter Zugang, DFT Scan, isolierter RAM-Test]

Erweiterung zum Selbsttest durch Ergänzung von

- Testmustergeneratoren: Pseudo-Zufallsgenerator, vorzugsweise LFSR (Linear Feedback Shift Register), Zähler, ...
- Ausgabekontrolle: vorzugsweise Prüfkennzeichen mit LFSR, ...
- Testablaufsteuerung:
 - Initialisierung Testmustergenerator, Prüfkennzeichen, Testobjekt
 - Wiederhole für alle Testschritte
 - Testmustergenerator weiterschalten,
 - Bildung der Schaltungsausgaben aus den Testeingaben,
 - Übernahme der Testausgaben in das Prüfkennzeichen,
- Prüfkennzeichenabfrage über Testbus.

Vorteile:

- lange Tests mit voller Systemgeschwindigkeit,
- in der Einsatzphase nutzbar als Einschalttest.

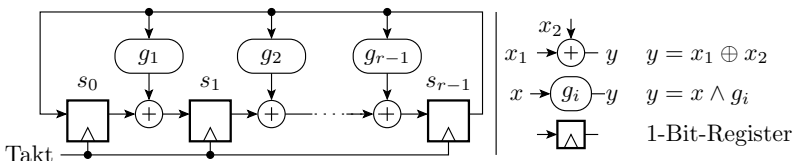


Pseudo-Zufallsregister

Linear rückgekoppelte Schieberegister

Gebräuchlichste Pseudo-Zufallsgeneratoren für Schaltungsselbsttest.

Ein linear rückgekoppelte Schieberegister (LFSR **L**inear **F**eedback **S**hift **R**egister) in einer ersten Ausführung verschiebt seinen r -Bit-Zustand $\mathbf{s} = (s_{r-1}, s_{r-2}, \dots, s_0)$ um eine Stelle nach links und addiert, wenn das herausgeschobene Bit s_{r-1} gleich »1« ist, eine Bitvektorkonstante $\mathbf{g} = (g_{r-1}, g_{r-1}, \dots, g_1, 1)$ zum Zustand \mathbf{s} :



Für jede Bitanzahl r des Zustandsvektors gibt es Konstanten \mathbf{g} , sog. »primitive Polynome«, bei denen alle Zustände außer 000...0 ineinander übergehen. Nur solche Konstanten \mathbf{g} werden verwendet.

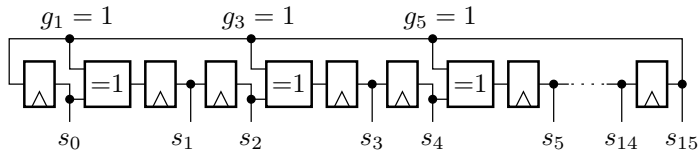
[Konstantenelimination entfernt in der Schaltung die EXOR mit $g = 0_i$]

Primitiven Polynome und die Konstante g

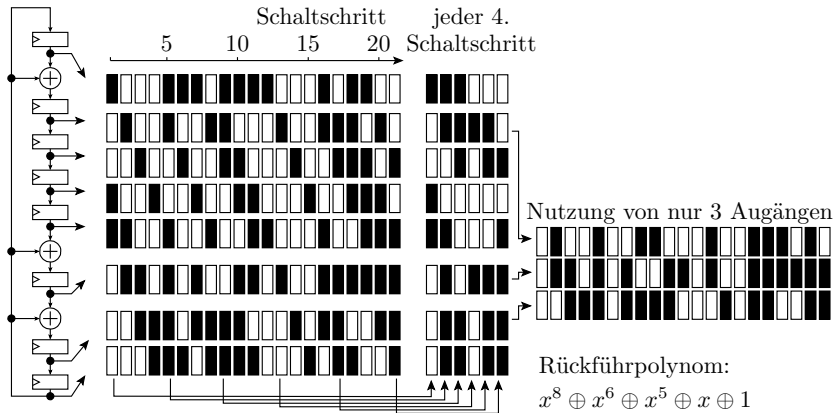
Mit dem Internet-Suchbegriff »Primitive Polynome« findet man z.B. für 16-Bit LFSR:

$$x^{16} \oplus x^5 \oplus x^3 \oplus x \oplus 1$$

Das bedeutet $g_1 = g_3 = g_5 = 1$ und alle anderen $g_i |_{i \notin \{1,3,5\}} = 0$. In Realisierung als Digitalschaltung für $g_i = 1$ EXOR-Gatter einfügen und für $g_i = 0$ EXOR-Gatter weglassen. g_0 und g_r sind immer 1.)



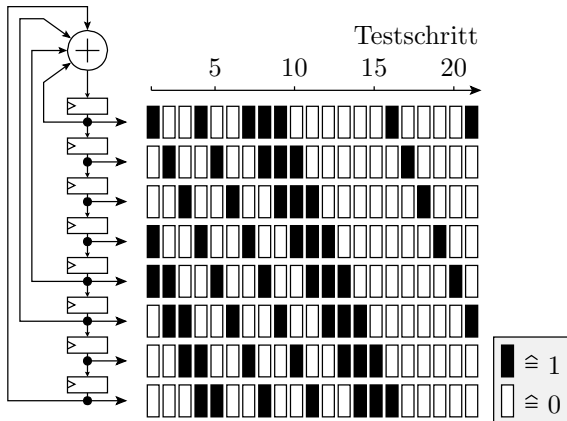
Pseudo-Zufallsfolge eines 8-Bit-LFSR



Falls die »Streifenmuster« durch die Schiebeoperationen stören, nur einen Teil der Ausgaben nutzen.

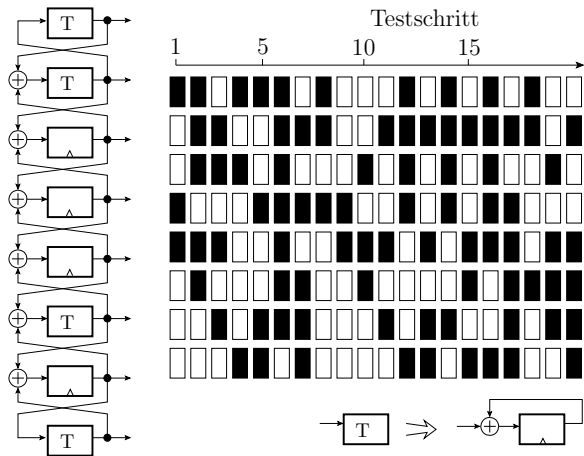
[nach EXOR keine sichtbare Korrelation; genauer phasenverschobene zykl. Bitfolgen]

LFSR mit zentraler Rückführung



Statt Ausgang auf mehrere Bits, Rückkopplung mehrere Bits auf den Eingang. Bei gleichen Rückführstellen haben LFSR mit zentraler und dezentraler Rückführung gleiche Zyklusstruktur.

Zellenautomaten



Folgebit gleich EXOR der Nachbarbits optional plus eigener Bitwert.

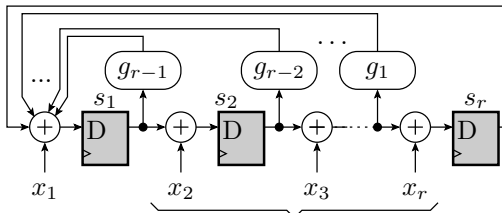
[Ausgewählte Toggle-Zellenzuordnungen \Rightarrow max. Zykluslänge]



Signaturregister

LFSR für parallele Datenströme

Für die Bildung auf Prüfkennzeichen ist es nur wichtig, dass die Abbildung pseudo-zufällig hinsichtlich der zu erwartenden Verfälschungen erfolgt. Diese Eigenschaft hat auch ein rückgekoppeltes Schieberegister, bei dem die Daten modulo-2 als Bitvektoren zu den Registerzuständen addiert werden (paralleles Signaturregister).



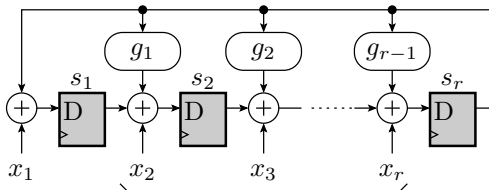
Erweiterungsmöglichkeit auf mehrere Eingänge

[auch dezentrale Rückführungen]

[auch wenige als r Eingänge oder über EXOR-Netzwerk für mehr als r Eingänge]

[EXOR mit $g_i = 0$ oder $x_i = 0$ entfallen durch Konstanteneliminitaion]

Dezentrale Rückführung auch ok



Erweiterungsmöglichkeit auf mehrere Eingänge

- Autonome Zyklusstruktur bei gleichen Rückführkoeffizienten g_i für zentrale und dezentrale Rückführung gleich.
- Für Signaturregister werden, wie für Pseudo-Zufallsgeneratoren auch, primitive Rückkopplungen bevorzugt, bei denen bei $x_i = 0$ alle Zustände $s \neq 00 \dots 0$ zyklisch durchlaufen werden.
- Keine primitive Rückführung bedeutet aber nicht, dass dann die Fehlermaskierungswahrscheinlichkeit signifikant größer 2^{-r} ist.

Experiment Maskierungswahrscheinlichkeit

Ein Signaturregister bildet mit Wahrscheinlichkeit 2^{-r} durch Fehler verfälschte Testausgabedaten auf die Sollsignatur ab und maskiert den Nachweis. Zu erwartende Anzahl der Maskierungen:

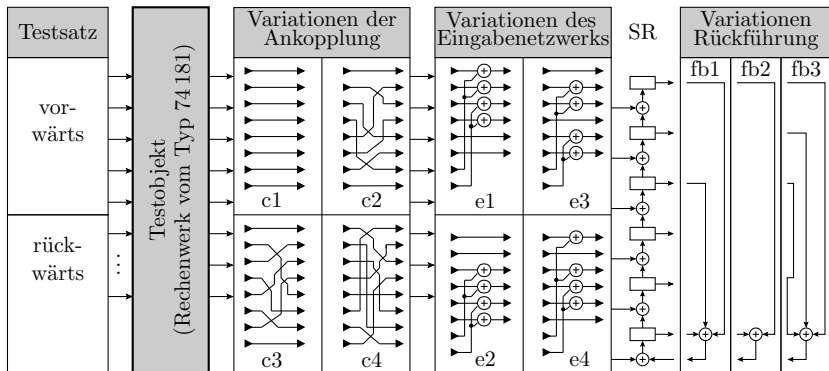
$$\lambda = \mathbb{E} [\#F_{\text{Msk}}] = \mathbb{E} [\#F_{\text{N}}] \cdot 2^{-r}$$

$\#F_{\text{N}}$ – Anzahl der vom Testsatz nachweisbaren Fehler; F_{Msk} – Anzahl der maskierten Fehler. Für größere r ist F_{Msk} poisson-verteilt:

$$\mathbb{P} [\#F_{\text{Msk}} = k] = e^{-\lambda} \cdot \frac{\lambda^k}{k!}$$

Experiment, dass die Anzahl der Maskierungen nur von r , aber nicht von der SR-Schaltung, der Rückführung oder Ankopplung abhängt:

- Simulation einer 4-Bit-ALU mit einem Testsatz, der alle 250 unterstellten Hauffehler erkennt. Berechnung der Prüfkennzeichen für alle Fehler. Zählen der Maskierungen.
- Variation der Testsatzreihenfolge,
- Variation der Ankopplung an das LFSR und
- Variation der Rückführung.



Zu erwartende Anzahl der Maskierungen für $r = 6$ und $\#F_N = 250$:

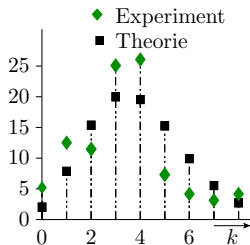
$$\lambda = \mathbb{E}[X] = 250 \cdot 2^{-6} = 3,9$$

Zu erwartende Zählwerte für jede Maskierungsanzahlen:

$$\#X(k) = 96 \cdot \mathbb{P}[\#F_{Msk} = k] = 96 \cdot e^{-3,9} \cdot \frac{3,9^k}{k!}$$

Experimentell bestimme Anzahl der Maskierungen:

		e1				e2				e3				e4			
		c1	c2	c3	c4	c1	c2	c3	c4	c1	c2	c3	c4	c1	c2	c3	c4
vorwärts	fb1	3	4	1	2	3	4	3	3	4	2	4	3	4	3	4	6
	fb2	3	4	1	7	2	2	1	4	2	1	1	3	2	5	3	7
	fb3	5	2	2	8	4	5	3	4	3	6	3	7	5	3	3	4
rückwärts	fb1	6	4	4	2	3	4	3	4	3	4	3	4	4	8	4	5
	fb2	2	0	0	1	4	1	4	1	0	0	0	1	1	1	4	1
	fb3	2	4	3	4	4	8	5	8	3	3	3	6	3	3	4	3



Zu erwartende Anzahl der Maskierungen:

$$\#X(k) = 96 \cdot e^{-3,9} \cdot \frac{3,9^k}{k!}$$

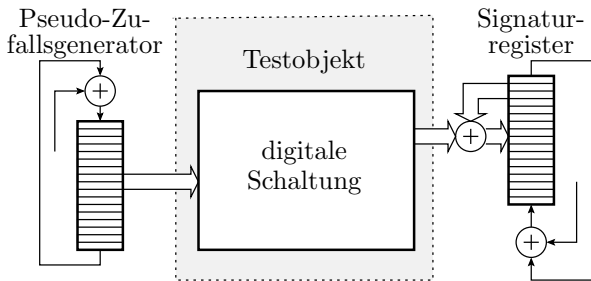
k	0	1	2	3	4	5	6	7	8
$\#X(k)$	1,3	7,5	14,7	19,2	18,73	14,6	9,5	5,3	2,6
Experiment	5	12	11	25	26	6	4	3	4

- Abweichungen zwischen $\#X(k)$ und Experiment nicht signifikant.
- Kein dominanter Einfluss der Rückkopplung etc. auf die Anzahl der Maskierungen.



Selbsttest mit LFSR

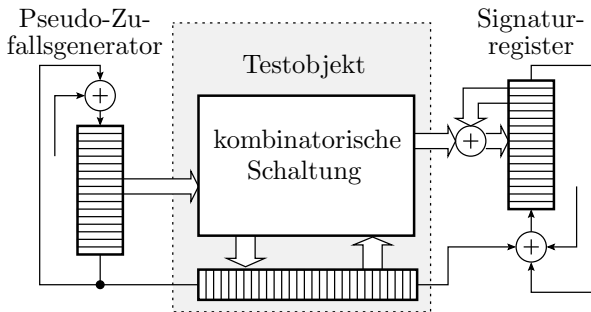
Selbsttest mit LFSR



- Einrahmen der Schaltung mit Schieberegistern und Ergänzung einiger EXOR-Gatter.
- Wenn als Schieberegister vorhandene Ein- und Ausgaberegister verwendet werden, besonders niedriger Zusatzaufwand.
- Test mit voller Schaltungsgeschwindigkeit von Millionen bis Milliarden Test pro Sekunde.

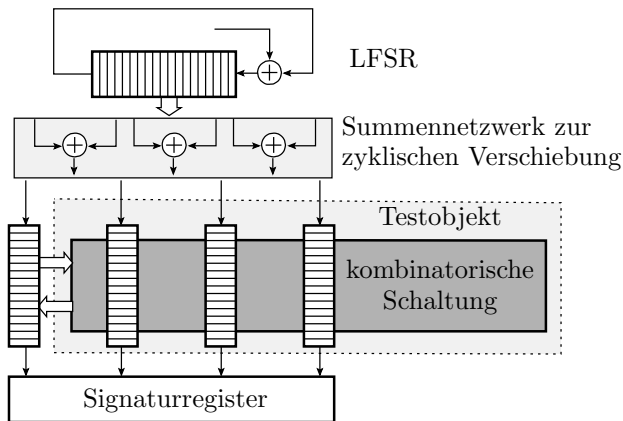
[Init. TMG \neq 0; Zyklus > Testanzahl; def. Sollausgabe]

BIST plus Scan



- Erweiterung der Speicherzellen des Testobjekts in Rückführungen zu einem Scan-Register.
- Zwischen den Testschritten Zustandsregister seriell in das Signaturregister auslesen und neu beschreiben.
- Der isolierte Test der Übergangsfunktion reduziert in die Regel die erforderliche Testzeit viel mehr, als sie sich durch die zusätzlichen Schiebeschritte erhöht.

Für sehr große Systeme, z.B. Multi-Chip-Module auch mehreren Scan-Registern, die zwischen den Testschritten parallel gelesen und mit neuen Zufallswerten beschrieben werden.



Zyklische Verschiebung der seriell in die Scan-Register eingespeisten Pseudo-Zufallsfolgen großer Scan-Registerlänge, ... (vergl. G. Kemnitz: Test und Verlässlichkeit von Rechnern. Springer 2007).



Fehlerorientierte Wichtung



Fehlerorientierte Wichtung

Wenn zu viele Modellfehler mit der akzeptierten Testsatzlänge für den Selbsttest nicht nachgewiesen werden:

- fehlerorientiert Testsuche und
- hinzufügen zum Zufallstest.

Wichtung ist eine kostengünstige Alternative zu einem Pattern-Speicher für die Ergänzung gezielt berechneter Eingaben zu einem Zufallstest.

Wichtung und Beobachtbarkeit eines Bits x_i :

Wichtung ist die Auftrittswahrscheinlichkeit von Signalwert $x_i = 1$.

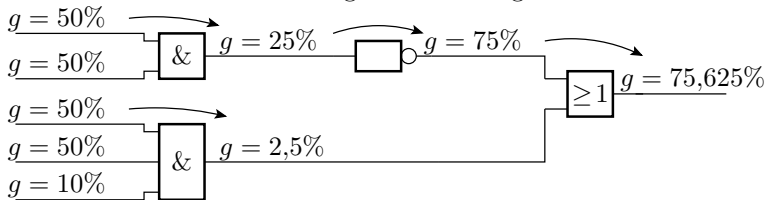
$$g(x_i) = \mathbb{P}[x_i = 1]$$

Beobachtbarkeit ist die Wahrscheinlichkeit, dass eine fehlerverursachte Invertierung eines Bits x_i mindestens ein Ausgabebit y_i verfälscht.

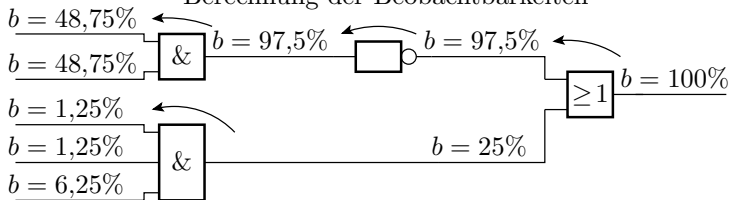
Über Eingabewichtung lassen die Wichtungen und Beobachtbarkeiten interner Knoten und damit die FF-Raten der Fehler verändern.

[Bereits erwähnt als Grundlage einer Heuristiken für die Testsuche]

Berechnung der Wichtungen



Berechnung der Beobachtbarkeiten



Berechnung der Nachweiswahrscheinlichkeiten:

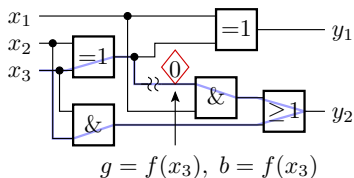
$$p_{\text{sa}0}(x_i) = b(x_i) \cdot g(x_i)$$

$$p_{\text{sa}1}(x_i) = b(x_i) \cdot (1 - g(x_i))$$

Rekonvergente Auffächerung

Bei rekonvergenter Auffächerung werden abhängige Zufallswerte log. verknüpft, so dass die einfachen Regel für $\mathbb{P}[A \wedge B]$ etc. nicht gelten.

Fehlernachweiswahrscheinlichkeitsberechnung über Wertetabelle:



— rekonvergente
Auffächerung

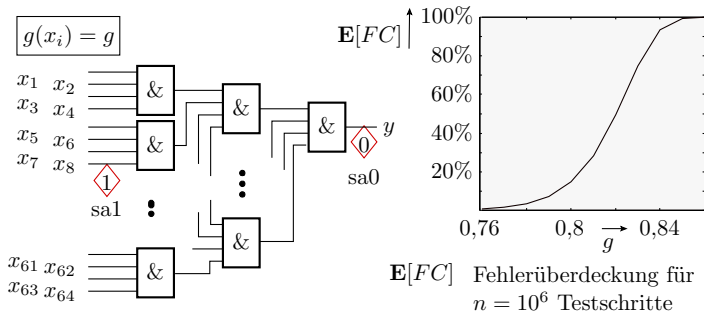
■ Eingaben die den
Fehler nachweisen

Eingabe			Ausg.		$h(\mathbf{x})$ für $g(x_i) = g$		
x_3	x_2	x_1	y_2	y_1	$g=0,5$	$g=0,3$	$g=0,7$
0	0	0	0	0	0,125	0,343	0,027
0	0	1	0	1	0,125	0,147	0,036
0	1	0	0	1	0,125	0,147	0,036
0	1	1	1	0	0,125	0,036	0,147
1	0	0	0	1	0,125	0,147	0,036
1	0	1	1	0	0,125	0,036	0,147
1	1	0	1	0	0,125	0,036	0,147
1	1	1	1	1	0,125	0,027	0,343

Nachweiswahrscheinlichkeit: 0,25 0,072 0,294

Fehlerüberdeckung und Wichtung

[Untersuchung der $FC(g)$ für Millionen Tests]



Nachweiswahrscheinlichkeiten:

der 64 sa1-Fehler

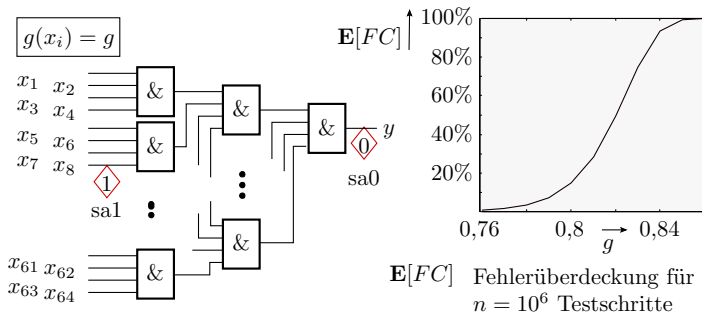
$$p_{sa1}(1) = g^{63} \cdot (1 - g)$$

$$p_{sa1}(n) = 1 - e^{-g^{63} \cdot (1-g) \cdot n}$$

des sa0-Fehlers

$$p_{sa0}(1) = g^{64}$$

$$p_{sa0}(n) = 1 - e^{-g^{64} \cdot n}$$



Zu erwartende Fehlerüberdeckung als Mittelwert der Nachweiswahrscheinlichkeit aller 65 Haftfehler:

$$\mathbb{E}[FC] = 1 - \frac{64 \cdot e^{-g^{63} \cdot (1-g) \cdot n} + e^{-g^{64} \cdot n}}{65}$$

Eine Wichtungserhöhung von $g \leq 76\%$ auf $g \geq 86\%$ erhöht die Fehlerüberdeckung von 0 auf 1.

[Eingänge mit Invertierung müssten im Beispiel mit $1 - g$ gewichtet werden]
 [allgemein individuelle Wichtungsoptimierung für jeden Eingang]



Fehlerorientierte Wichtungsauswahl

Statt einheitlicher Wichtung aller Eingabesignale

- Festlegung einer individuellen Wichtung für jeden Eingang,
- Wichtungsumschaltung jeweils nach einem längeren Zufallstest.

Ein pragmatischer Ansatz dazu aus⁵:

- 1 Festlegung einer größeren Menge von Modellfehlern.
- 2 Längerer Test mit ungewichteten Zufallswerten und Abhaken aller damit nachweisbaren Modellfehler.
- 3 Suche für die restlichen Modellfehler eine Eingabewichtung, die deren mittlere Nachweiswahrscheinlichkeiten erheblich erhöht.
- 4 Längerer Test mit den so gewichteten Zufallswerten und Abhaken aller damit nachweisbaren Modellfehler.
- 5 Wenn erforderlich, Wiederholung von Schritt 3 und 4.

⁵J. Hartmann, G. Kemnitz: How to do weighted random testing for BIST? ICCAD 1993.



Wichtungsauswahl

- Für alle Modellfehler, die der ungewichtete Zufallstest nicht nachweist, gezielte Berechnung einer Eingabevektors mit möglichst vielen »X« (Don't Care) -Stellen, z.B. mit D-Algorithmus.
- Zuordnung von einem der 5 Wichtungswerte zu jedem Eingang:

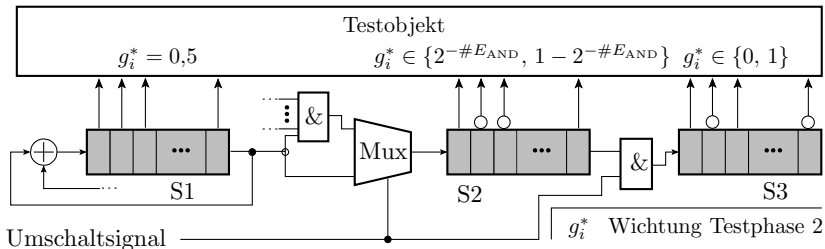
Test	1	2	3	4	5	6	7	8	9	10	11	$g(x_i)$
x_1	1	X	0	1	0	0	X	1	0	1	1	0,5
x_2	1	1	X	1	1	X	1	X	X	X	X	1
x_3	0	0	1	0	0	X	0	X	1	0	0	0,125
x_4	1	0	X	0	X	0	1	X	1	X	0	0,5
x_5	1	1	1	X	0	1	1	1	X	0	1	0,875

$$g(x_i) = \begin{cases} 0 & x_i \in \{0, X\} \\ 2^{-\#E_{\text{AND}}} & \#N \gg \#E \\ 0,5 & \text{sonst} \\ 1 - 2^{-\#E_{\text{AND}}} & \#N \ll \#E \\ 1 & x_i \in \{1, X\} \end{cases}$$

($\#N$ – Anzahl der Nullen; $\#E$ – Anzahl der Einsen für Eingang x_i in den Testeingaben; $\#E_{\text{AND}} \in \{2, 3, 4, \dots\}$ – Anzahl der [N]AND-verknüpften Zufallsfolgen zur Wichtungserzeugung).

- Die zweite und weitere Wichtungsberechnungen berücksichtigen nur noch die Testeingaben bis dahin nicht nachgewiesener Fehler.

Implementierung als Selbsttest



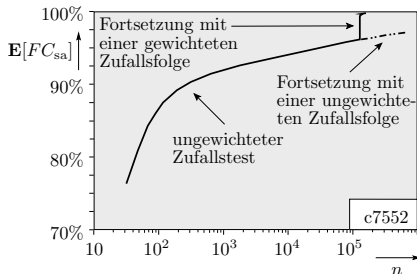
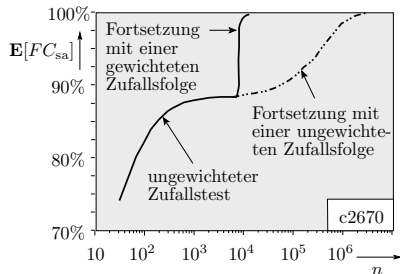
- Testphase 1: Erzeugung ungewichteter Pseudo-Zufallseingaben mit LFSR S1. Serielle Weitergabe an die Schiebereg. S2 und S3.
- Testphase 2: Verringerung der Wichtung
 - in S2 durch UND-Verknüpfung von $\#E_{AND}$ Ausgabefolgen von S1 und für S2 und
 - in S3 durch »UND 0«.

Erzeugung $g(x_i) = 1 - 2^{-\#E_{AND}}$, $g(x_i) = 1$ durch Inverierung.

Kaum aufwändiger als BIST mit ungewichtetem Zufallstest.

Benchmark-Experimente

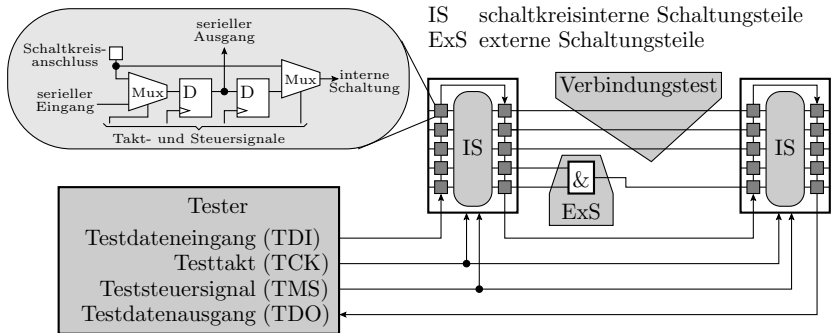
- Beispielentwurf für die größten ISCAS-Benchmarks c2670, c7552.
- Test mit 10^4 bzw. 10^5 ungewichteten Zufallsmustern, die 90% bzw. 95% der Haftfehler nachweisen.
- Gezielte Testberechnung für die restlichen Haftfehler.
- Individuelle Wichtung aller Eingabebits zur Maximierung der mittleren Auftretshäufigkeit der berechneten Testeingaben.
- Weitere 10^4 bzw. 10^5 gewichtet Zufallstests weisen alle verbleibenden nichtredundanten Modellfehler nach.





Testbus

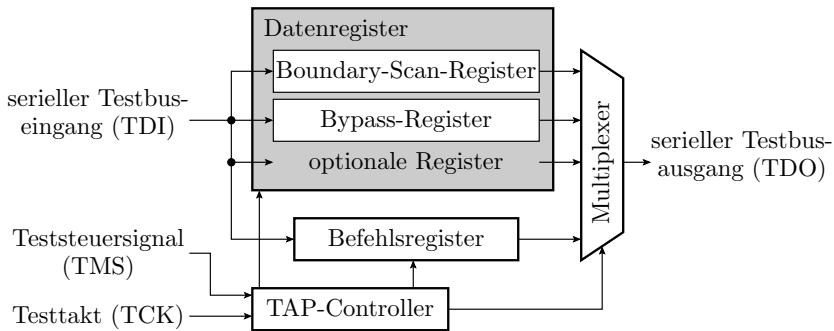
Testbus – JTAG (IEEE 1149.1)



- Steuerung integrierter Testhilfen in Schaltkreisen und auf Baugruppen über die 4 Bussignale: TDI, TDO, TCK und TMS.
- Ersatz der mechanischen Nadeln für den Verbindungs-, Bestückungs- und isolierten Test integrierte Teststrukturen.
- Laden und Lesen von Programm- und Konfigurationsdaten, ...

[Zelle; Verschaltung TDI, TDO, TCK und TMS]

JTAG-Testbusarchitektur der Schaltkreise



Eine Boundary-Scan-Implementierung umfasst:

- den TAP- (Test Access Port) Controller
- ein Befehlsregister
- mehrere Testdatenregister (mindestens das Boundary-Scan- und das Bypass-Register).

[Vendor-, ID-, BIST-, Programmier-Register, ..., Auswahl mit TAP]

JTAG-Busprotokoll

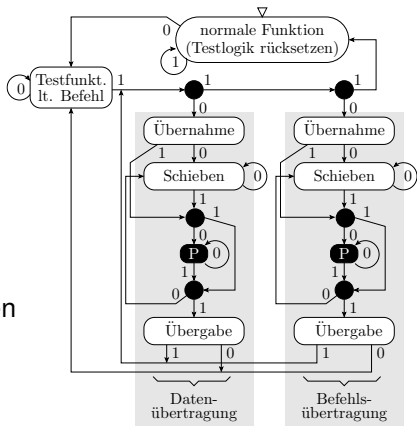
- Automat mit 16 Zuständen
- Kantenauswahl über TMS- (Test Mode Select)

Typischer Testbeginn:

- Befehlsregister lesen (Kontrolle JTAG-Kette⁶),
- Bauteilnummern lesen (Bestückungskontrolle),
- Einen Teil der Schaltkreise auf Bypass setzen. Für die anderen ein Datenregister auswählen.

Verbindungstest:

- BS-Register auswählen,
- Wiederhole alle Testschritte
 - BS-Register Übernahme, Schieben, Übergabe



⁶Übernahme 01 in Bit 0 und Bit 1 zur Kontrolle auf Unterbrechungen der Schiebekette.



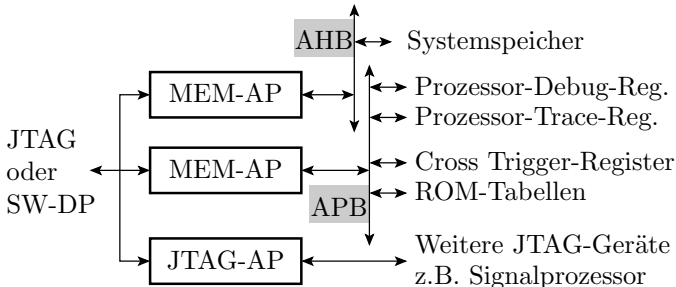
Über das Befehlsregister lässt sich eine

- beliebige Anzahl von Datenregistern adressieren und
- eine beliebige Anzahl von »Testfunktionen laut Befehl« steuern.

Typische zusätzlich unterstützte Testfunktionen:

- lesbare Hersteller- und Bauteilidentifikationsregister für den Bestückungstests.
- Steuerung von Steuerung 0, 1, hochohmig und Lesen der Logikwerte der IC-Anschlüsse für den Verbindungstest.
- Laden und Lesen von Programm- und Konfigurationsdaten.
- Steuerung integrierter Debugger-Funktionen,
- Steuerung von Selbsttests, ...

ARM-Testbusarchitektur



AHB Advanced High Performance Bus

APB Advanced Peripheral Bus

SW-DP Serial-Wire Debug Port

MEM-AP Memory Access Port, serielle Adressüberg., Datentransfer

JTAG-AP JTAG-Master für weitere Cores mit JTAG auf dem Chip

[ARM: verbreitetste Mikroprozessor-Architektur für Embedded (Smartphones, ..)]

[über MEM-AP + AHB: auch Kontrolle über Speicher]

[über MEM-AP + APB: Zugriff Debugger-Register, ROM-Tabellen, ..]



JTAG-Datenregister für den Baugruppentest:

- Boundary-Scan, Bypass und 40-Bit ID-Code

JTAG-Datenregister für den SW-Test:

- 32-Bit Debug-Control und Status-Register (DSCR)
- Instruction-Transfer-Register (ITR): 32 Befehlsbit + ein Statusbit, zur Ausführung von Prozessorbefehlen in einem speziellen Debug-Modus.
- Debug Communications Channel (DCC): 32-Bit-Datenwort + 2 Statusbits für +n bidirektionalen Datentransfer mit Prozessorkern.
- Embedded Trace Module (ETM): 7 Adressbits + 32-Bit-Datenwort + 1 R/W-Bit zur Steuerung von Trace-Operationen⁷.
- Debug-Modul: 7 Adressbits + 32-Bit-Datenwort + 1 R/W Bit. Zugriff auf die Register für Hardware Breakpoints, Watchpoints etc..

ARM Befehle für Zusammenarbeit Programm Debugger: HALT (Übergang in Debug-Modus, in dem über das ITR Befehle eingefügt werden können) und RESTART zum Verlassen des Debug-Modus.

⁷Trace-Aufzeichnung entweder in einen eingebetten Trace-Buffer (ETB) auf dem Chip oder Ausgabe über einen High-Speed-Port.