Test und Verlässlichkeit Foliensatz 4: Test und Überwachung

Prof. G. Kemnitz

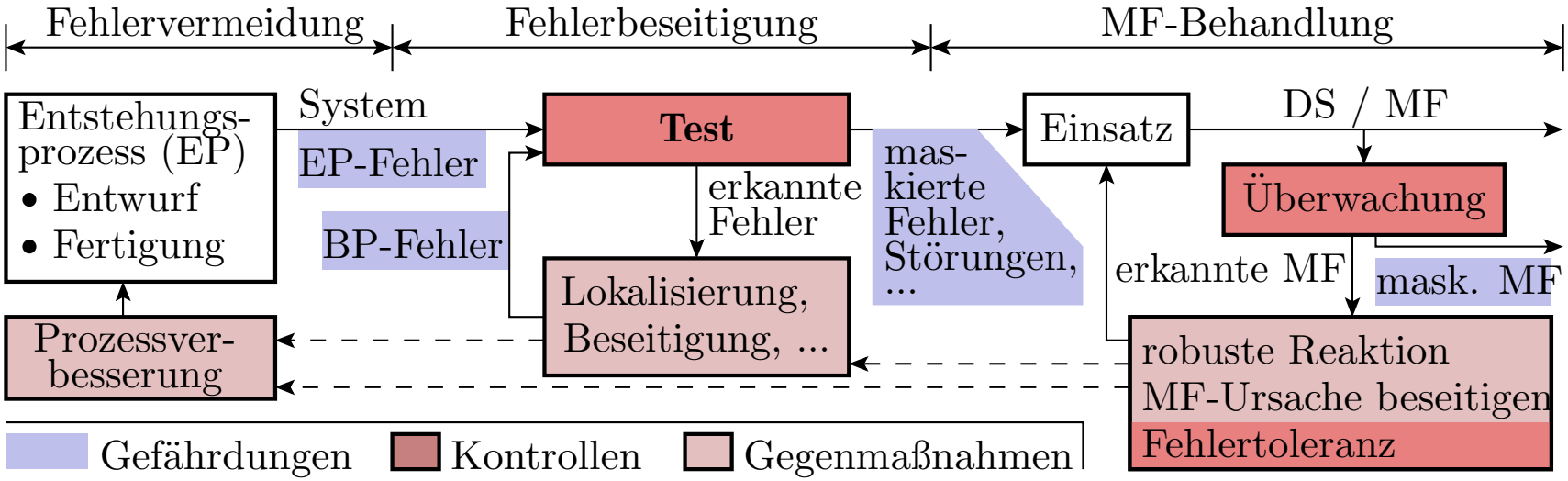
Institut für Informatik, TU Clausthal (TV_F4)

6. März 2023

Inhalt Foliensatz TV_F4: Tests und Kontrollen

Vorlesung	12	13	14
Folie	3	74	84

Test und Verlässlichkeit



Verlässlichkeit wird durch Iterationen aus Kontrollen, der Beseitigung erkannter Gefährdungen und Erfolgskontrollen gesichert. Mit der unterstellten Fehlerkultur *»Beseitigung alle erkannten Gefährdungen (MF, Fehler, ...)«* hängt die Verlässlichkeit hauptsächlich von den Tests und Kontrollen auf allen drei Ebenen ab. Fortsetzungsthemen:

- Vielfalt der Testmöglichkeiten
- Überwachungstechniken mehr im Detail und
- Fehlertoleranz.

Test

Testarten und Kenngrößen

Einteilung der Testarten:

- Statische Tests: direkte Kontrolle von Merkmalen.
- Dynamische Tests: Ausprobieren der Systemfunktion.

Kenngrößen:

- Fehlerüberdeckung, Anteil der nachweisbaren Fehler:

$$FC = \left.\frac{\#DF}{\#F}\right|_{\mathrm{ACR}} \tag{1.34}$$

- Phantom-MF-Rate während des Tests, Anteil der Testausgaben, die als fehlerhaft erkannt werden, aber in Wirklichkeit korrekt sind:

$$\zeta_{\mathrm{PhanT}} = \left.\frac{\#PM}{n}\right|_{\mathrm{ACR}} \tag{1.35}$$

$\#DF$	Anzahl der erkennbaren Fehler (number of **d**etectable **f**aults).
$\#F$	Anzahl der Fehler (number of **f**aults).
ζ_{PhanT}	Phantom-MF-Rate des Tests (**phan**tom MF rate during **t**est).
$\#PM$	Anzahl der **P**hantom-**M**F, d.h. der korrekten DS, die als MF klassifiziert werden.
n	Anzahl der Tests.
ACR	Geeignete Zählwertgrößen (**a**ppropriate **c**ounting **r**anges).

Aus Sicht des Verlässlichkeitsgewinns (siehe Abschn. 1.4.5):

- Vortests, die zusammen mit einer Anzahl von $n_{\min}$ dynamischen Tests einen Anteil FC_{PT} erkennen und
- einem ausreichend langen Zufallstest zur Absenkung der MF-Rate durch Fehler auf einen akzeptablen Wert:

$$\mu_{\mathrm{FNE}}(n) = \mu_{\mathrm{FNE}}(n_0) \cdot \left(\frac{n}{n_0}\right)^{-k} \tag{1.42}$$

$$\zeta_{\mathrm{F}}(n) = \frac{k \cdot \mu_{\mathrm{FNE}}(n)}{n} \tag{1.43}$$

Die Vortests umfassen

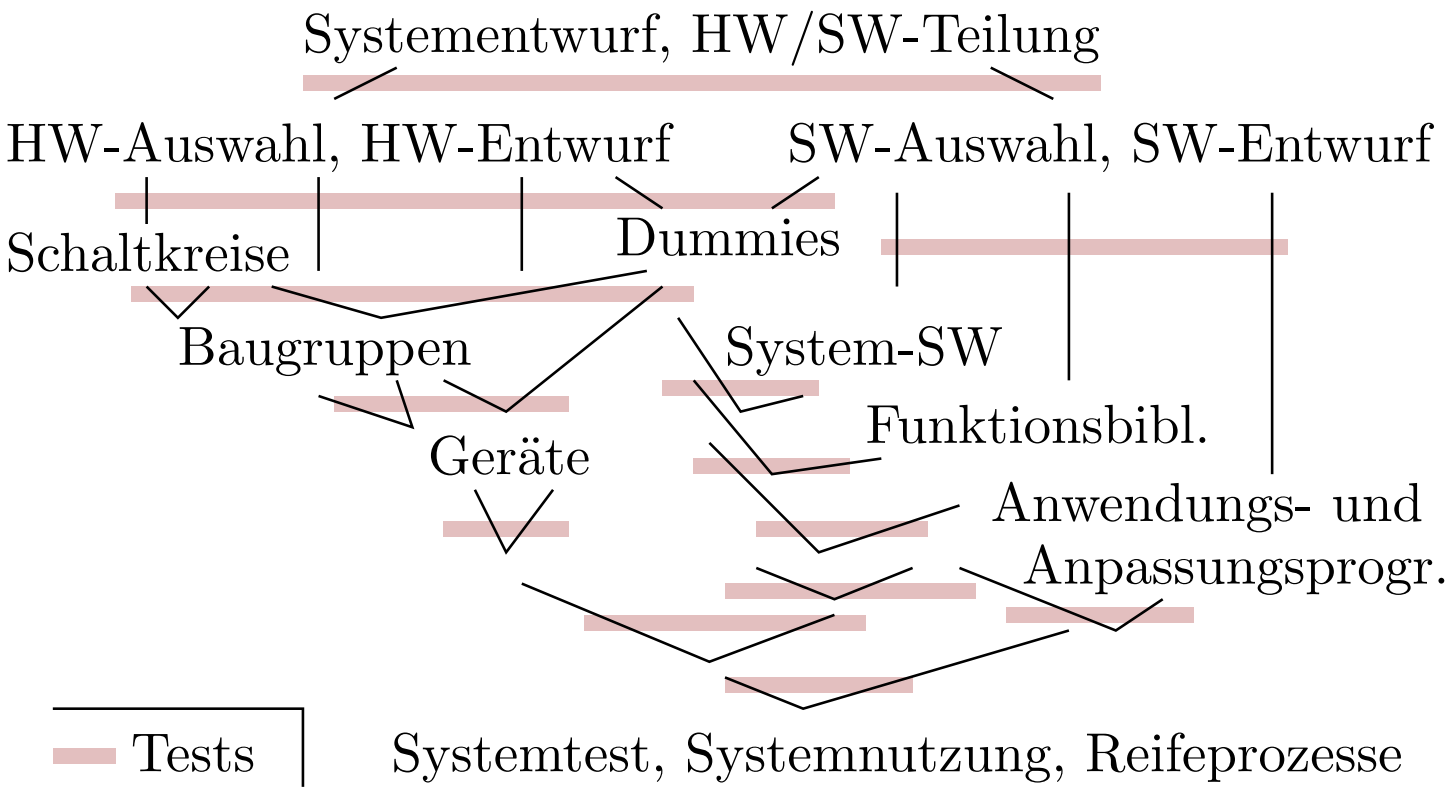
- statische Tests, Grobtests, fehlerorientiert ausgewählte Tests, ...
- nach Entwurfs- und Fertigungsschritten,
- an Komponenten, Teilsystemen und dem Gesamtsystem.

Dieser Abschnitt behandelt als speziellere Tests:

- Inspektion, insbesondere von Entwurfsergebnissen,
- Funktionstests für Teil- und Gesamtsysteme und
- Fertigungstest für Baugruppen.

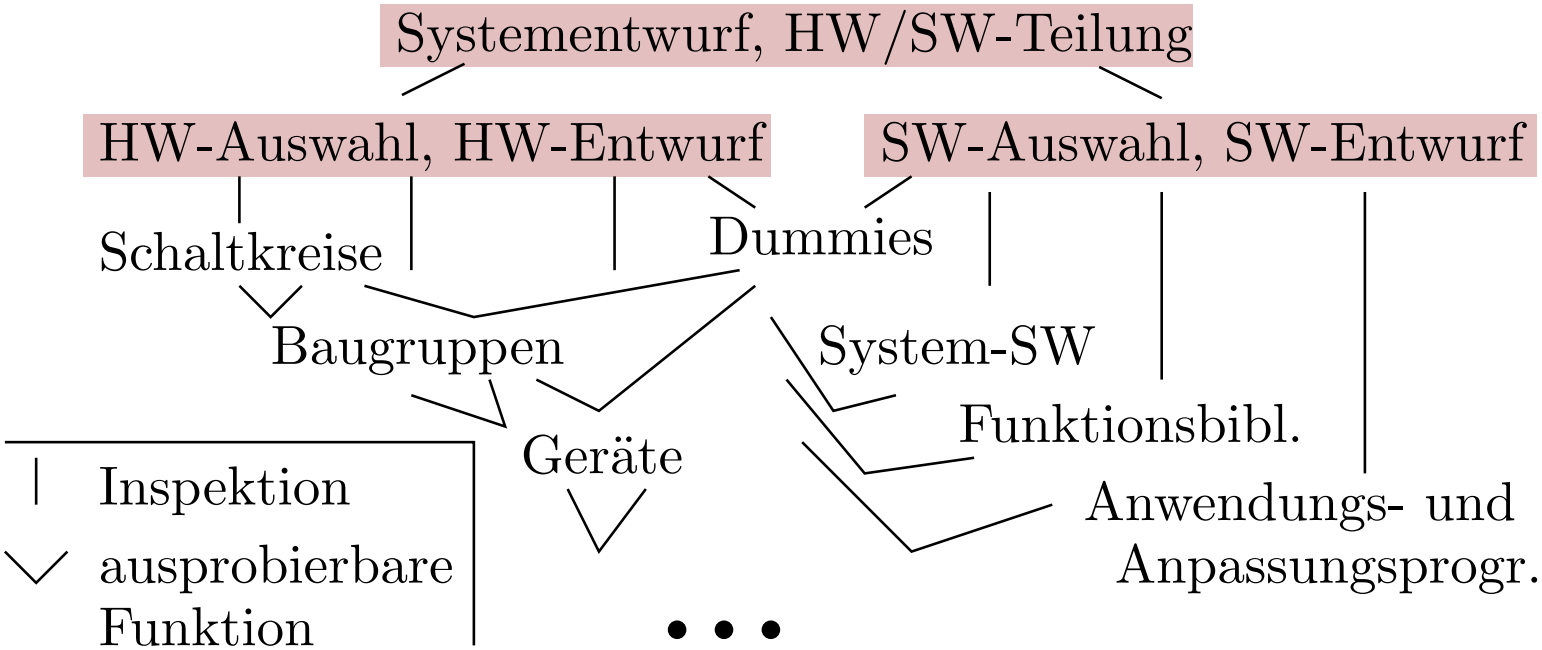
μ_{FNE} zu erwartende Anzahl der nicht beseitigten (**n**ot **e**liminated) Fehler.

Vielfalt der Testaufgaben

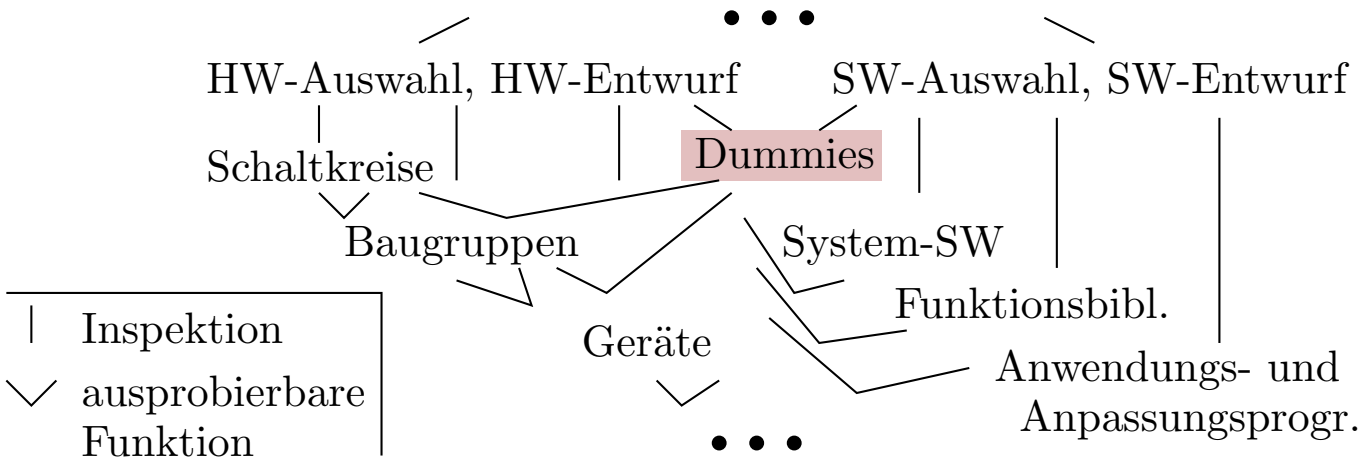


- Der Entstehungsprozess eines IT-Systems, in dem auch die Mehrheit der Fehler entstehen, besteht aus vielen Schritten. Nach Jedem Schritt wird getestet.

1. Test

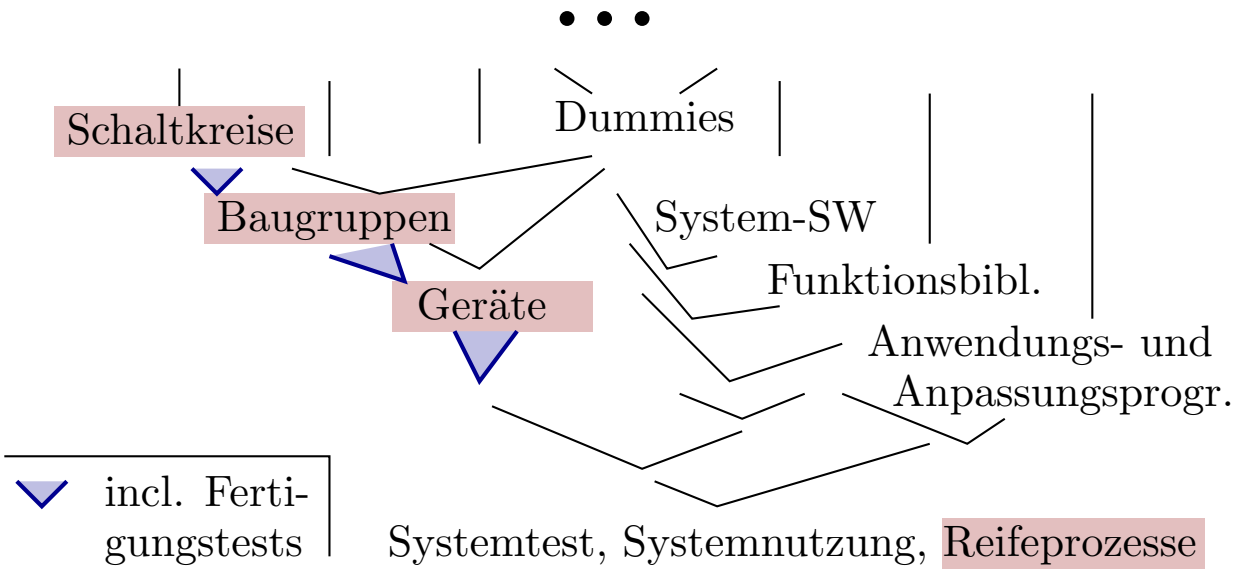


- Der Entwurf eines IT-Systems beginnt mit der Zusammenstellung der Anforderungen, grundlegenden Entscheidungen zur Systemarchitektur incl. HW/SW-Aufteilung.
- Zur Aufwands- und Fehlervermeidung wird immer versucht, existierende Komponenten mitzunutzen: Schaltkreise, Baugruppen, Geräte, System-Software, SW-Bibliotheken, ...
- Zur Ermöglichung dynamischer Test strebt der System- und Komponentenentwurf ausprobierbare Funktionseinheiten an, in frühen Entwurfsphasen auch Demonstratoren oder Simulationen.



- Ohne ausprobierbare Funktionseinheiten beschränken sich die Kontrollmöglichkeiten überwiegend auf Inspektionen, d.h. die Sichtung der entstandenen Dokumente.
- Dummies sind Nachbildungen noch nicht verfügbarerer Funktionseinheiten (HW oder SW), um bereits erste Tests an übergeordneten Funktionseinheiten durchführen zu können.
- Funktionseinheiten incl. Dummies werden vor Integration in übergeordnete Einheiten seperat getestet und auch jede daraus zusammengesetzte Einheit wird getestet (siehe Abschn. 1.4.7).
- Dynamisch testbare Funktionseinheit werden auch statischen Tests unterzogen, z.B. Syntax, Datentypen, ...

1. Test



- HW benötigt zusätzlich zu den Entwurfstests Fertigungstests für jedes gefertigte Produkt.
- Der Systemtest des Gesamtsystems zerfällt in Vortests und einer hinreichenden Anzahl von Zufallstests zur Sicherug der Zuverlässigkeit (siehe Abschn. 1.4.5).
- Große Systeme müssen in der Einsatzphase weiterreifen, in dem die Info über beobachtetet Fehlfunktionen bei genutzten Service-Leistung gesammelt, die zugrundeliegenden Fehler gesucht und abgestellt werden (siehe Abschn. 1.4.6).

Inspektion

Inspektion (Review)

Inspektion, Sichtprüfungen (von lat. inspicere = besichtigen, betrachten). Angewendet auf:

- Dokumentationen (Spezifikation, Nutzerdokumentation, ...),
- Programmcode, Testausgaben,
- Schaltungsbeschreibungen, Konstruktionspläne, ...

besonders in frühen Entwurfsphasen, bevor die Zielfunktionen in simulierbarer oder ausprobierbarer Form beschrieben sind oder bei Testausgaben, bevor Sollwerte festgelegt oder Kontrollen programmiert sind.

Eigenenschaften von Inspektionen als Kontrollen:

- großer manueller Arbeitsaufwand,
- geringere Güte als maschinelle Kontrollen.
- Nachweis auch von nicht funktionalen Fehlern: Verstößen gegen Vereinbarungen und Standards, Antipattern*.
- Know-How-Weitergabe als positiver Zusatzeffekt.

* Beschreibungselemente, die die Übersichtlichkeit beeinträchtigen, die Kontrolle erschweren und die Fehlerentstehung begünstigen.

Kenngrößen einer Inspektion

Kenngrößen wie bei jedem anderen Test:

- Fehlerüberdeckung:

$$FC = \left.\frac{\#DF}{\#F}\right|_{\mathrm{ACR}} \tag{1.34}$$

- Phantom-MF-Rate des Tests:

$$\zeta_{\mathrm{PhanT}} = \left.\frac{\#PM}{n}\right|_{\mathrm{ACR}} \tag{1.35}$$

Weitere Kenngrößen zur Bewertung von Inspektionsprozessen [3]:

- Effizienz (EFC): Gefundene Abweichungen pro Mitarbeiterstunde.
- Effektivität (EFT): Gefundene Abweichungen je 1000 NLOC.

$\#DF$	Anzahl der erkennbaren Fehler (number of **d**etectable **f**aults).
$\#F$	Anzahl der Fehler (number of **f**aults).
ζ_{PhanT}	Phantom-MF-Rate des Tests (**phan**tom MF rate during **t**est).
$\#PM$	Anzahl der **P**hantom-**M**F, d.h. der korrekten DS, die als MF klassifiziert werden.
n	Anzahl der Tests.
ACR	Geeignete Zählwertgrößen (**a**ppropriate **c**ounting **r**anges).
NLOC	Netto Lines of Code, Anzahl der Code-Zeilen ohne Kommentar und Leerzeilen.

Beispiel 4.1: Inspektion

Programmgröße: 10.000 NLOC, Arbeitsaufwand: 200 Stunden, 228 gefundene Fehler, davon 156 funktionale. Geschätzte Gesamtfehleranzahl (vor der Inspektion): 300, davon 200 funktionale. Wie groß sind

a) Inspektionsfehlerüberdeckung FC?

b) Effizienz?

c) Effektivität?

	gesamt	funktionale Fehler	sonstige Fehler
$FC = \frac{\#DF}{\#F}$	$\frac{228}{300}$	$\frac{156}{200}$	$\frac{72}{100}$
Effizienz (EFC)	$\frac{228\text{ Fehler}}{200\text{ h}}$	$\frac{156\text{ Fehler}}{200\text{ h}}$	$\frac{72\text{ Fehler}}{200\text{ h}}$
Effektivität (EFT)	$\frac{228\text{ Fehler}}{10.000\text{ NLOC}}$	$\frac{156\text{ Fehler}}{10.000\text{ NLOC}}$	$\frac{72\text{ Fehler}}{10.000\text{ NLOC}}$

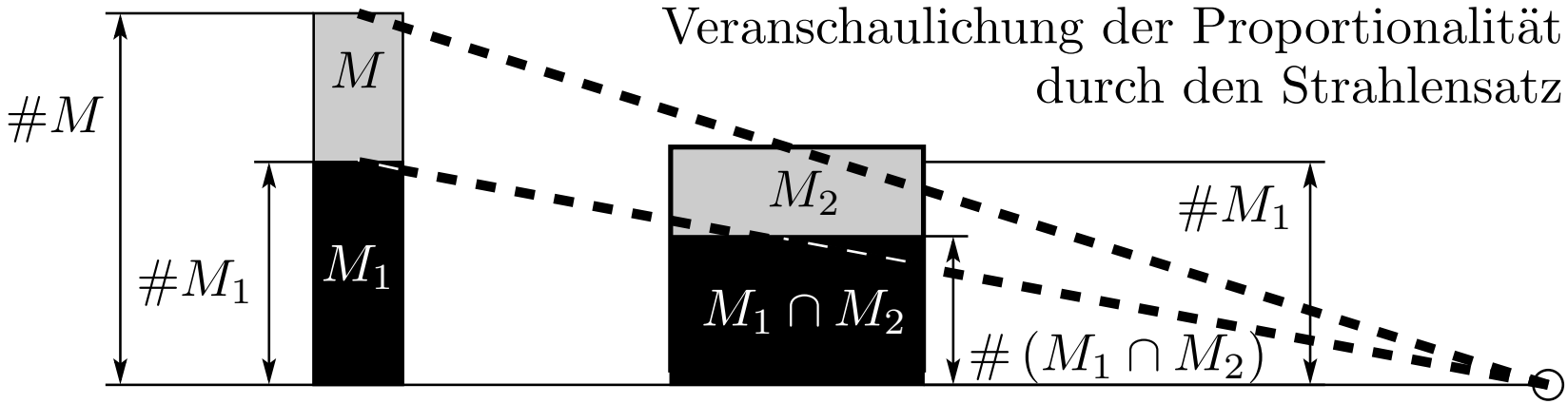
- Die Fehlerüberdeckung FC basiert auf schlecht überprüfbaren Schätzwerten für die Gesamtfehleranzahl.
- Die Angaben zur Effizienz und Effektivität sind objektiver und bewerten die Inspektionstechnik.

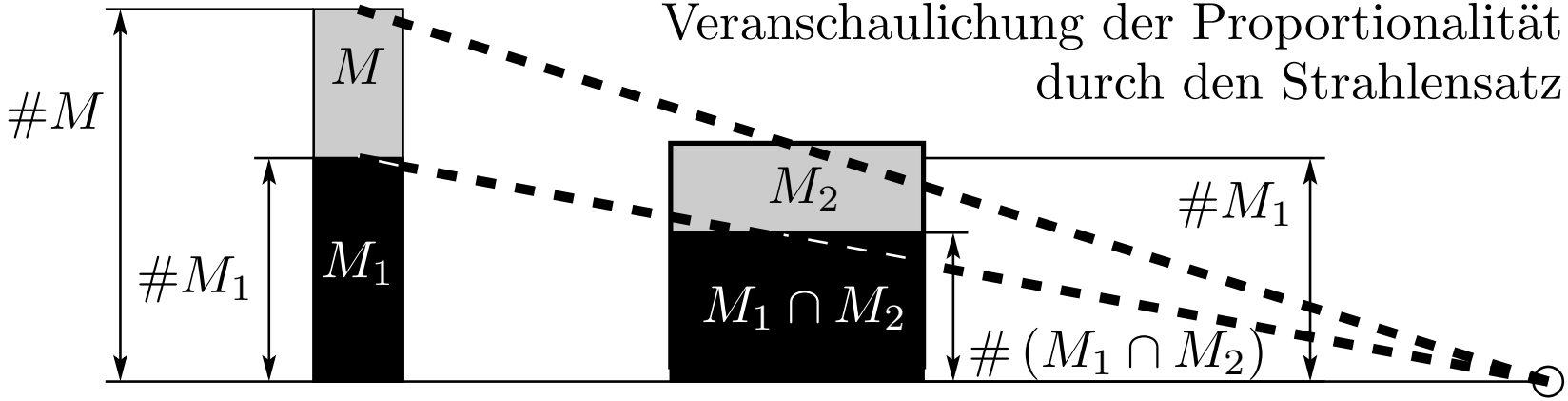
Gesamtfehleranzahl nach »Capture-Recapture«

Abgeleitet von einem Schätzer für die Größe von Tierpopulationen (z.B. von Vögeln in einem Gebiet) [2, 7, 4].

- Aus einer Menge M unbekannter Größe wird eine Menge M_1 von Tieren eingefangen, gekennzeichnet und freigelassen.
- Nach Vermischung der Population Menge M_2 von Tieren einfangen. Gekennzeichnete Tiere werden gezählt.

Bei tierunabhängiger Einfangwahrscheinlichkeit ergibt sich der Anteil der Tiere, die beim zweiten Einfangen gekennzeichnet sind, über den Strahlensatz:





$$\frac{\#M_1}{\#M} \approx \frac{\#(M_1 \cap M_2)}{\#M_2}$$

(#... – Größe der Mengen, hier Anzahl der Tiere; M – Menge aller Tiere, M_1, M_2 – beim ersten bzw. zweiten mal eingefangene Tiere; $M_1 \cap M_2$ – Menge der beide Male eingefangenen Tiere). Geschätzte Größe der Tierpopulation:

$$\#M \approx \frac{\#M_1 \cdot \#M_2}{\#(M_1 \cap M_2)}$$

Fehler statt Tiere

Zwei Inspekteure i finden jeweils eine Menge von F_i Fehlern:

$$\#F = \frac{\#F_1 \cdot \#F_2}{\#(F_1 \cap F_2)} \tag{1}$$

Die geschätzte Fehlerüberdeckung ist das Verhältnis der Anzahl der insgesamt von beiden Inspekteuren gefundenen Fehler $\#(F_1 \cup FF_2)$ zur geschätzten Geamtfehleranzahl $\#F$:

$$FC = \frac{\#(F_1 \cup F_2)}{\#F} = \frac{\#(F_1 \cap F_2) \cdot \#(F_1 \cup F_2)}{\#F_1 \cdot \#F_2} \tag{2}$$

Gebunden an die Annahmen:

- Inspekteure erkennen die Fehler unabhängig voneinander.
- Alle Fehler haben dieselbe Erkennungswahrscheinlichkeit.

$\#F$	geschätzende Gesamtfehleranzahl.
$\#F_i$	Anzahl der von Inspektor 1 bzw. Inspektor 2 gefundenen Fehler.
$F_1 \cap F_2$	Anzahl von beiden Inspekteuren getrennt gefundene gleiche Fehler.
FC	Fehlerüberdeckung (**f**ault **c**overage), Anteil der nachweisbaren Fehler.

Vertrauenswürdigkeit der Schätzung

Die Gesamtfehlerzahl nach Gl. 4.1 ist ein Istwerte, um den der zu schätzende Erwartungswert schwankt, bei Normalverteilung:

$$\mathrm{sr}\left(\mu\right) = x_{\mathrm{AV}} \mp \sigma \cdot \Phi^{-1}\left(1 - \tfrac{\alpha}{2}\right) \tag{3.50}$$

Die Zählwerte $\#F_1$, $\#F_2$ und $\#\left(F_1 \cap F_2\right)$ haben die relative Varianzen:

$$\frac{\mathrm{Var}[X]}{\mathbb{E}[X]^2} = \left(\frac{1}{x_{\mathrm{AV}}} - \frac{1}{x_{\mathrm{mcv}}}\right) \tag{3.52}$$

Bei der Multiplikation und Division von unabhängigen Zufallsgrößen addieren sich abschätzungsweise die relativen die Varianzen:

$$\frac{\mathrm{Var}[\#F]}{\#F^2} = \frac{1}{\#F_1} \cancel{- \frac{1}{\max(\#F_1)}} + \frac{1}{\#F_2} \cancel{- \frac{1}{\max(\#F_2)}} + \frac{1}{\#(F_1 \cap F_2)} - \frac{1}{\min(\#F_1, \#F_2)}$$

$$\frac{\mathrm{sr}(\mathbb{E}[\#F])}{\#F} = 1 \mp \sqrt{\frac{1}{\#F_1} + \frac{1}{\#F_2} + \frac{1}{\#(F_1 \cap F_2)} - \frac{1}{\min(\#F_1, \#F_2)}} \cdot \Phi^{-1}\left(1 - \tfrac{\alpha}{2}\right) \tag{3}$$

$\mathrm{sr}\left(\ldots\right)$	symmetrischer Bereich der wahrscheinlichen Werte.
x_{AV}	Istwert (actual value).
α	Irrtumswahrscheinlichkeit Werte außerhalb des geschätzten Bereichs.
x_{mcv}	maximal möglicher Zählwert (**m**aximum possible **c**ount **v**alue).
$\#F_i$	Anzahl der von Inspektor 1 bzw. Inspektor 2 gefundenen Fehler.
$F_1 \cap F_2$	Anzahl von beiden Inspekteuren getrennt gefundene gleiche Fehler.

Hinzu kommen systematische Schätzfehler:

- Capture-Recaptur unterstellt für alle Fehler dieselbe Erkennungswahrscheinlichkeit. In der Praxis reicht die Nachweiswahrscheinlichkeit von kaum übersehbar bis fast nicht erkennbar.
- Capture-Recaptur verbietet Informationsaustausch zwischen den Inspekteuren. Falls es doch einen Informationsaustausch gibt, vergrößert der die Menge der gleichen gefundenen Fehler $F_1 \cap F_2$ gegenüber einer unabhängen Suche.
- Wenn die Inspekteure ihre Fehlerlisten voneinander abschreiben $F_1 = F_2$, ergibt sich als Schätzwert für die Inspektionsfehlerüberdeckung 100%.

Beispiel 4.2: Inspektionsergebnisse für ein Programm

- Inspekteur 1: 228 gefundene Fehler.
- Inspekteur 2: 237 gefundene Fehler.
- Schnittmenge: 105 Fehler.

$\#F_1 = 228; \#F_2 = 237; \#\left(F_1 \cap F_2\right) = 105.$

a) Gesamtfehleranzahl und der Inspektionsfehlerüberdeckung nach Gl. 4.1 und 4.2?

b) Bereich der zu erwartenden Gesamtfehleranzahl nach Gl. 4.3 für $\alpha < 4\%$?

c) Kontrolle Aufgabenteil b über Berechnung der Ableitungen $\frac{\partial \mathbb{E}[\#F]}{\partial \#F_1}, \frac{\partial \mathbb{E}[\#F]}{\partial \#F_2}$ und $\frac{\partial \mathbb{E}[\#F]}{\partial \#(F_1 \cap F_2)}$ und der Varianz über Gl. 3.21?

d) Abschätzung des wahrscheinlichen Bereichs der zu erwartenden Fehlerüberdeckung über den Lösungsweg von Aufgabenteil c?

$\#F_1 = 228; \#F_2 = 237; \#(F_1 \cap F_2) = 105.$

a) Gesamtfehleranzahl und der Inspektionsfehlerüberdeckung nach Gl. 4.1 und 4.2?

$$\#F = \frac{\#F_1 \cdot \#F_2}{\#(F_1 \cap F_2)} \tag{4.1}$$

$$FC = \frac{\#(F_1 \cap F_2) \cdot \#(F_1 \cup F_2)}{\#F_1 \cdot \#F_2} \tag{4.2}$$

$$\#F = \frac{228 \cdot 237}{105} = 515$$

$$FC = \frac{\#F_1 + \#F_2 - \#(F_1 \cap F_2)}{\#F} = \frac{228 + 237 - 105}{515} = 70\%$$

$\#F$	geschätzende Gesamtfehleranzahl.
$\#F_i$	Anzahl der von Inspektor 1 bzw. Inspektor 2 gefundenen Fehler.
$F_1 \cap F_2$	Anzahl von beiden Inspekteuren getrennt gefundene gleiche Fehler.
FC	Fehlerüberdeckung (**f**ault **c**overage), Anteil der nachweisbaren Fehler.

$\#F_1 = 228; \#F_2 = 237; \#(F_1 \cap F_2) = 105.$

b) Bereich der zu erwartenden Gesamtfehleranzahl nach Gl. 4.3 für $\alpha < 4\%$?

$$\frac{\mathrm{sr}(\mathbb{E}[\#F])}{\#F}) \approx 1 \mp \sqrt{\frac{1}{\#F_1} + \frac{1}{\#F_2} + \frac{1}{\#(F_1 \cap F_2)} - \frac{1}{\min(\#F_1, \#F2)}} \cdot \Phi^{-1}\left(1 - \frac{\alpha}{2}\right) \qquad (4.3)$$

Mit dem Schätzwert $\#F = 515$ aus Aufgabenteil b.

α	2,27%	0,13%	0	2%	1%	0,5%	0,2%	0,1%
$\Phi^{-1}(1-\alpha)$	2	3	4	2,05	2,33	2,57	2,88	3,10

$$\begin{aligned}
\mathrm{sr}(\mathbb{E}[\#F]) &= 515 \cdot \left(1 \mp \sqrt{\cancel{\frac{1}{228}} + \frac{1}{237} + \frac{1}{105} - \cancel{\frac{1}{228}}} \cdot \Phi^{-1}\left(1 - \frac{\alpha}{2}\right)\right) \\
&= 515 \mp (1 \mp 0{,}117 \cdot 2{,}05) \\
&= (391,\ 639)
\end{aligned}$$

Der Bereich des wahrscheinlichen Erwartungswertes ist Schätzwert ∓ 124 Fehler bzw. 24%.

$\#F_1 = 228; \#F_2 = 237; \#(F_1 \cap F_2) = 105.$

c) Kontrolle Aufgabenteil b über Berechnung der Ableitungen $\frac{\partial \mathbb{E}[\#F]}{\partial \#F_1}, \frac{\partial \mathbb{E}[\#F]}{\partial \#F_2}$ und $\frac{\partial \mathbb{E}[\#F]}{\partial \#(F_1 \cap F_2)}$ und der Varianz über Gl. 3.21?

$$\mathrm{sr}(\mu) = x_{\mathrm{AV}} \mp \sigma \cdot \Phi^{-1}\left(1 - \frac{\alpha}{2}\right) \tag{3.50}$$

$$\mathrm{Var}[f(X_1, X_{2,\ldots})] = \sum_{i=1}^{n} \left(\frac{\partial (f(X, X_{2,\ldots}))}{\partial X_i}\bigg|_{\forall \mathbb{E}} \right)^2 \cdot \mathrm{Var}[X_i] \tag{3.21}$$

$$\frac{\partial \#F}{\partial \#F_1} = \frac{\#F(\#F_1+1, \#F_2, \#(F_1 \cap F_2)) - \#F(\#F_1, \#F_2, \#(F_1 \cap F_2))}{1} = 2{,}26$$

$$\frac{\partial \#F}{\partial \#F_2} = \frac{\#F(\#F_1, \#F_2+1, \#(F_1 \cap F_2)) - \#F(\#F_1, \#F_2, \#(F_1 \cap F_2))}{1} = 2{,}17$$

$$\frac{\partial \#F}{\partial \#(F_1 \cap F_2)} = \frac{\#F(\#F_1, \#F_2, \#(F_1 \cap F_2)+1) - \#F(\#F_1, \#F_2, \#(F_1 \cap F_2))}{1} = -4{,}48$$

$$\mathrm{Var}[F] = (2{,}26)^2 \cdot \mathrm{Var}[\#F_1] + (2{,}17)^2 \cdot \mathrm{Var}[\#F_2] + (4{,}48)^2 \cdot \mathrm{Var}[\#F_1 \cap F_2]$$

$$= 2{,}26 \cdot 228 + 2{,}17 \cdot 237 + 4{,}48 \cdot 105 \cdot \left(1 - \frac{105}{228}\right) = 3414$$

$\#F_1 = 228; \#F_2 = 237; \#(F_1 \cap F_2) = 105.$

$$\mathrm{Var}[f(X_1, X_{2,\ldots})] = \sum_{i=1}^{n} \left(\frac{\partial(f(X, X_{2,\ldots}))}{\partial X_i}\bigg|_{\forall \mathbb{E}} \right)^2 \cdot \mathrm{Var}[X_i] \tag{3.21}$$

$$\begin{aligned} \mathrm{Var}[FC] &= 3414 \\ \mathrm{sd}[FC] &= 60{,}1 \\ \mathrm{sr}(\mathbb{E}[\#FC]) &\approx 515 \mp 60{,}1 \cdot 2{,}05 = (391,\ 638) \end{aligned}$$

Der geschätzter Bereich in Aufgabenteil b war $(391,\ 639)$. Bis auf kleine numerische Unterschiede liefert die Rechnung über Gl. 3.21 mit einer numerischen Bestimmung der Ableitungen dasselbe Ergebnis.

$\#F_1 = 228; \#F_2 = 237; \#(F_1 \cap F_2) = 105.$

d) Abschätzung des wahrscheinlichen Bereichs der zu erwartenden Fehlerüberdeckung über den Lösungsweg von Aufgabenteil c?

$$\mathrm{Var}\left[f\left(X_1,\, X_{2,\ldots}\right)\right] = \sum_{i=1}^{n} \left(\frac{\partial (f(X, X_{2,\ldots}))}{\partial X_i}\bigg|_{\forall \mathbb{E}} \right)^2 \cdot \mathrm{Var}\left[X_i\right] \tag{3.21}$$

$$\begin{aligned}
\frac{\partial FC}{\partial \#F_1} &= \frac{|FC(\#F_1+1, \#F_2, \#(F_1 \cap F_2)) - FC(\#F_1, \#F_2, \#(F_1 \cap F_2))|}{1} = -0{,}108\% \\
\frac{\partial FC}{\partial \#F_2} &= \frac{|FC(\#F_1, \#F_2+1, \#(F_1 \cap F_2)) - FC(\#F_1, \#F_2, \#(F_1 \cap F_2))|}{1} = -0{,}097\% \\
\frac{\partial FC}{\partial \#(F_1 \cap F_2)} &= \frac{|FC(\#F_1, \#F_2, \#(F_1 \cap F_2)+1) - FC(\#F_1, \#F_2, \#(F_1 \cap F_2))|}{1} = 0{,}453\%
\end{aligned}$$

$$\begin{aligned}
\mathrm{Var}\left[FC\right] &= (-0{,}108\%)^2 \cdot \mathrm{Var}\left[\#F_1\right] + (-0{,}097\%)^2 \cdot \mathrm{Var}\left[\#F_1\right] + (0{,}453\%)^2 \cdot \mathrm{Var}\left[\#F_1 \cap F_2\right] \\
&= (-0{,}108\%)^2 \cdot 228 + (-0{,}097\%)^2 \cdot 237 + (0{,}453\%)^2 \cdot 105 \cdot \left(1 - \frac{105}{228}\right) = 0{,}00165
\end{aligned}$$

$$\begin{aligned}
\mathrm{sd}\left[FC\right] &= 0{,}0406 \\
\mathrm{sr}\left(\mathbb{E}\left[FC\right]\right) &= 70\% \cdot (1 \mp 0{,}0406 \cdot 2{,}05) = (64{,}2\%,\, 75{,}8\%)
\end{aligned}$$

Modellfehlerbasierte Abschätzung

Einbau von Kontrollfehlern (Mutationen) in das zu inspezierende Datenmaterial und Abschätzung der zu erwartenden Fehlerüberdeckung aus dem Anteil der nachweisbaren Kontrollfehler nach Gl. 1.34:

$$FC = FC_{\mathrm{M}} = \frac{\#DF_{\mathrm{M}}}{\#F_{\mathrm{M}}} \tag{4}$$

Zu erwartenden Anzahl der tatsächlichen Fehler:

$$\#F = \frac{\#DF}{FC} = \#F_{\mathrm{M}} \cdot \frac{\#DF}{\#DF_{\mathrm{M}}} \tag{5}$$

Die Kontrollfehler sollen vergleichbar gut erkennbar sein wie die zu findenen Fehler und die Inspekteure dürfen sie nicht kennen. Nachträgliche Klassifizierung der gefundenen Fehler in Kontrollfehler, echte funktionale Fehler, echte sonstige Fehler und Phantomfehler.

FC	Fehlerüberdeckung (**f**ault **c**overage), Anteil der nachweisbaren Fehler.
$\#F$	Anzahl der Fehler (number of **f**aults).
$\#DF$	Anzahl der erkennbaren Fehler (number of **d**etectable **f**aults).
$\#F_{\mathrm{M}}$	Anzahl der **M**odell**f**ehler.
$\#DF_{\mathrm{M}}$	Anzahl der nachweisbaren Modellfehler.

Wahrscheinliche Bereiche nach

$$\mathrm{sr}\left(\mu\right) = x_{\mathrm{AV}} \mp \sigma \cdot \Phi^{-1}\left(1 - \tfrac{\alpha}{2}\right) \tag{3.50}$$

$$\frac{\mathrm{Var}[X \cdot Y]}{\mathbb{E}[X]^2 \cdot \mathbb{E}[Y]^2} = \frac{\mathrm{Var}[X]}{\mathbb{E}[X]^2} + \frac{\mathrm{Var}[Y]}{\mathbb{E}[Y]^2} \tag{3.17}$$

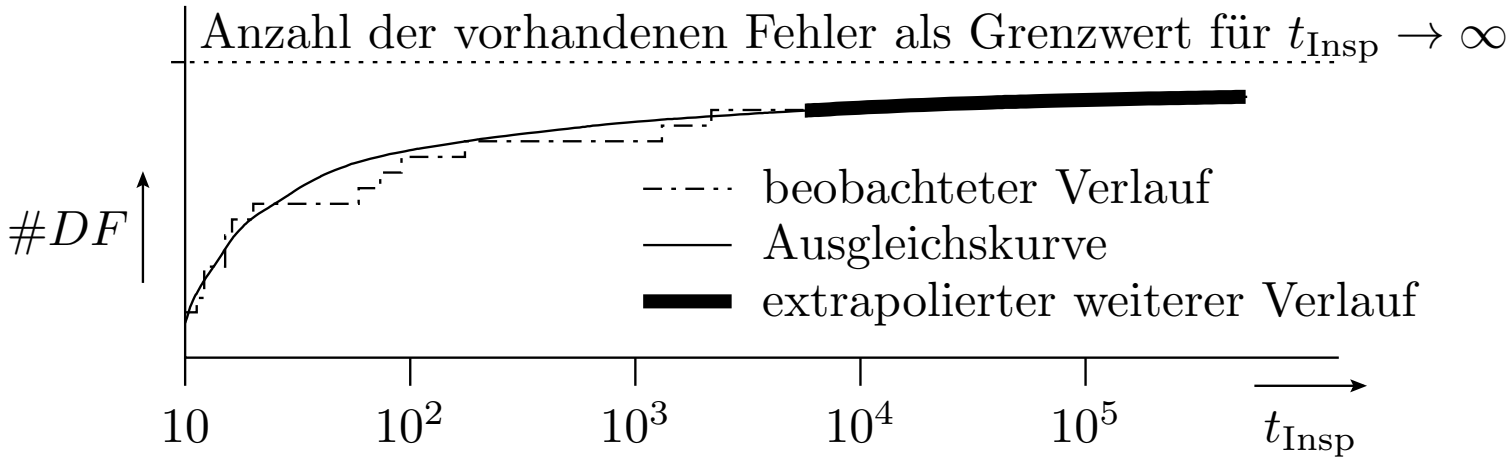
$$\frac{\mathrm{Var}[X]}{\mathbb{E}[X]^2} = \left(\frac{1}{x_{\mathrm{AV}}} - \frac{1}{x_{\mathrm{mcv}}}\right) \tag{3.52}$$

$$\mathrm{sr}\left[\mathbb{E}\left[FC\right]\right] = FC \cdot \left(1 \mp \sqrt{\tfrac{1}{\#DF_{\mathrm{M}}}} \cdot \Phi^{-1}\left(1 - \tfrac{\alpha}{2}\right)\right) \tag{6}$$

$$\mathrm{sr}\left[\mathbb{E}\left[\#F\right]\right] = \#F \cdot \left(1 \mp \sqrt{\tfrac{1}{\#DF_{\mathrm{M}}} + \tfrac{1}{\#DF}} \cdot \Phi^{-1}\left(1 - \tfrac{\alpha}{2}\right)\right) \tag{7}$$

- Bei gleicher Zählwertgröße genauere Schätzung, weil 1 bzw. 2 Zählwerte unter der Wurzen für die Standardabweichung weniger.
- Vermeidung systematische Fehler durch Informationsaustausch zwischen den Inspekteuren und Unterschieden in den Nachweiswahrscheinlichkeiten der zu findenen Fehler.
- Einbau und Beseitigung der Kontrollfehler möglichst automatisiert.
- Brauchbare Abschätzungen verlangt ca. 100 Kontrollfehler und mindestens ähnlich viele Fehler in den zu inspezierenden Daten.

Verteilung der Nachweiszeit

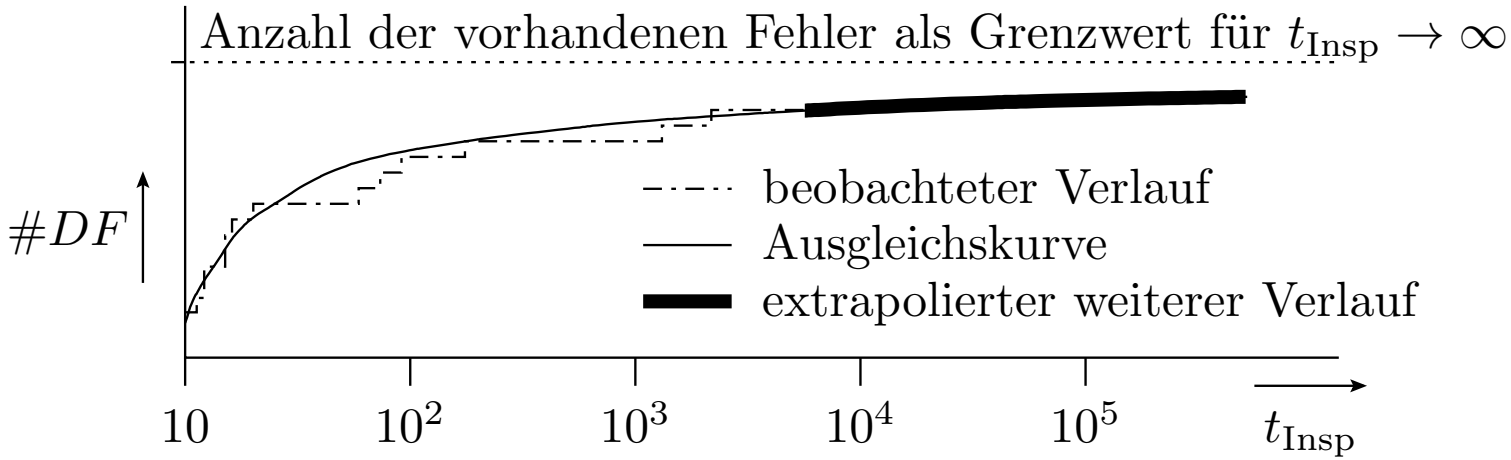


Auch für Inspektionen gilt in der Regel das Pareto-Prinzip. In einem kleinen Teil der Inspektionszeit wird die Mehrheit der Fehler gefunden. Deutet auf eine Pareto-Verteilung der Nachweiszeit (vergl. Gl. 3.74):

$$F_T(t_{\text{Insp}}) = \mathbb{P}\left(T \leq t_{\text{Insp}}\right) = FC\left(t_{\text{Insp}}\right) = \begin{cases} 0 & t_{\text{Insp}} \leq t_{\min} \\ 1 - \left(\frac{t_{\min}}{t_{\text{Insp}}}\right)^k & \text{sonst} \end{cases} \quad (8)$$

Schätzungen von $FC\left(t\right)$ für wesentlich längere Inspektionszeiten t aus dem Verlauf für wesentlich kürzere wie im Bild unsicher.

Inspektionsdauer und Effizienz



Die Effizienz EFT als »gefundene Abweichungen pro Mitarbeiterstunde« ist proportional zum Anstieg und damit zur Dichte der Nachweiszeit:

$$\frac{EFC(t_{\mathrm{Insp}})}{\#DF(t_{\mathrm{min}})} = f_T(t_{\mathrm{Insp}}) = \frac{\mathrm{d}F_T(t)}{\mathrm{d}t} = \frac{k \cdot t_{\mathrm{min}}^k}{t_{\mathrm{Insp}}^{k+1}} \tag{9}$$

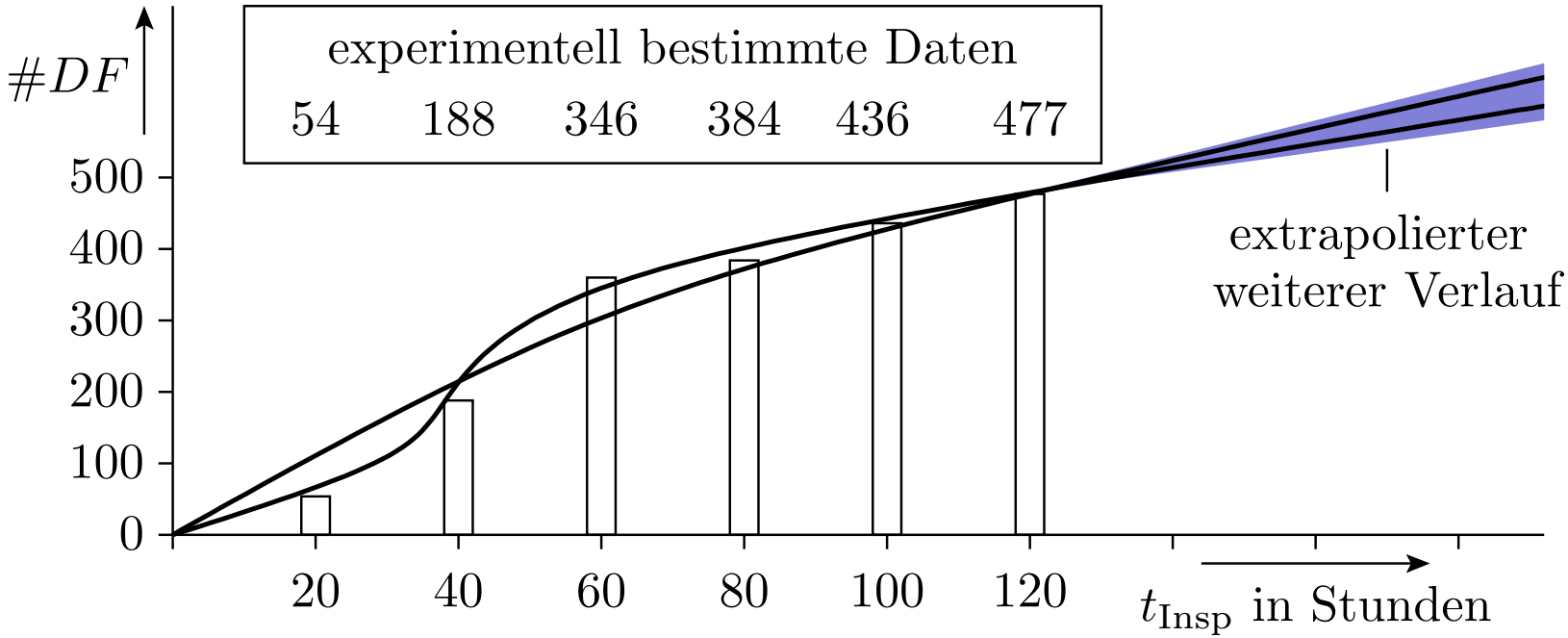
Mehr als umgekehrt proportionale Abnahme mit Inspektionsdauer t_{Insp}.

EFC	Effizienz Inspektion in gefundene Fehler je Mitarbeiterstunde.
$\#DF$	Anzahl der erkennbaren Fehler (number of **d**etectable **f**aults).

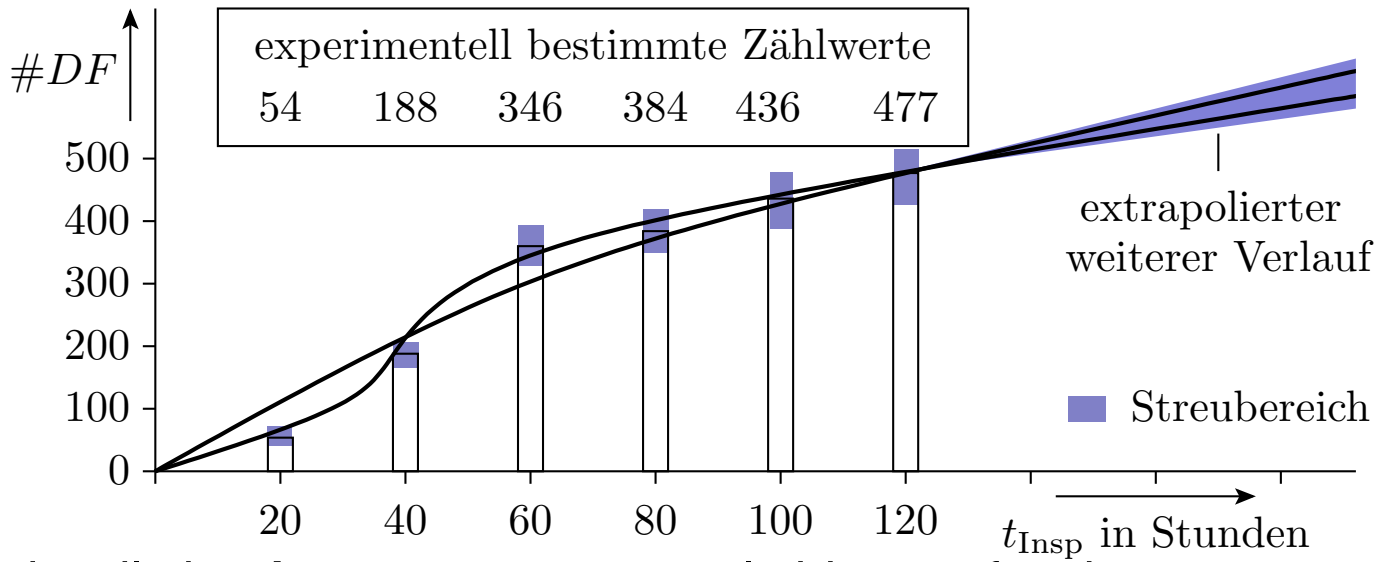
Experiment mit einem Inspekteur

Inspektion des Buchmanuskripts* plus Beispielprogramme:

- Anzahl der gefunden Fehler in Abhängigkeit von der Inspektionsdauer.



$\#DF$	Anzahl der erkennbaren Fehler (number of **d**etectable **f**aults).
t_{Insp}	Inspektionsdauer.
*	Bachelor-Arbeit von Yu Hong.



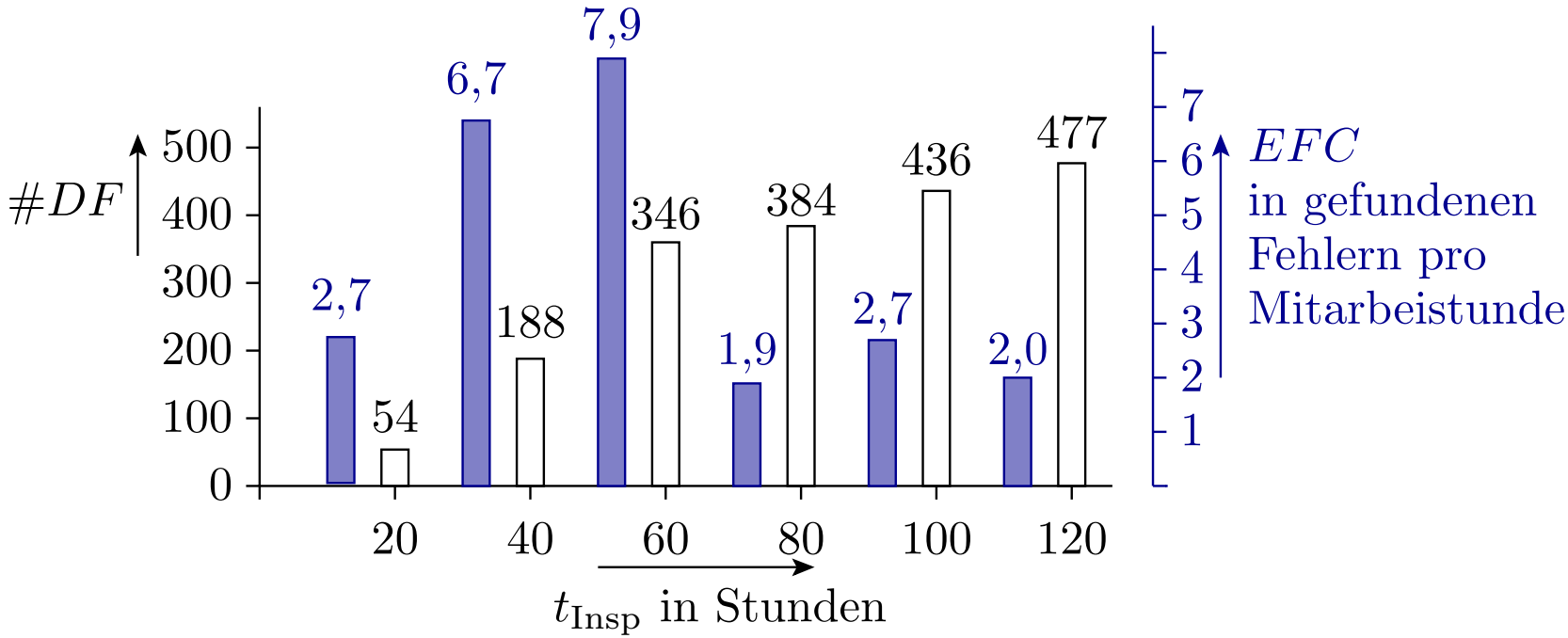
Unterschiedliche Approximationsmöglichkeiten für die weitere Abnahme der zu erwartenden Anzahl der nicht gefundenen Fehler, z.B. bei pareto-verteilter Nachweiszeit (Gl. 4.8):

$$\mathbb{E}\left[\#DF\left(t_{\text{Insp}}\right)\right] = \mathbb{E}\left[\#DF\left(t_0\right)\right] \cdot \left(\frac{t_{\text{Insp}}}{t_0}\right)^{-k}$$

$\#DF$	Anzahl der erkennbaren Fehler (number of **d**etectable **f**aults).
t_{Insp}	Inspektionsdauer.
t_0	Inspektionsdauer mit bekannter Anzahl nachweisbarerer Fehler.

Untschiede zwischen Inspektion und Zufallstest

Bei einem Zufallstest nimmt die Effizienz (gefundene Abweichungen pro Mitarbeiterstunde) mit der Testdauer ab, weil nicht erkannte Fehler tendenziell schlechter als erkannte Fehler nachweisbar sind.



Die Beispielinspektion hatte offenbar eine »Anlernphase«, in der die Effizienz mit der Inspektionsdauer zugenommen hat.

EFC Effizienz Inspektion in gefundene Fehler je Mitarbeiterstunde.

- Beim dritten und vierten mal »Lesen des Buchs und der Aufgabentexte« nahm im Experiment nicht nur die Effizienz, sondern auch die Zeit dafür deutlich ab, obwohl ein erheblicher Anteil (ca. 25%) der Fehler noch nicht gefunden war.

Anzahl, wie oft gelesen	1	2	3	4
Anzahl der gefundenen Fehler	251	126	79	4
Zeitaufwand	50 h	70 h		

- Ein Mensch als Inspekteur ermüdet offenbar nach einiger Zeit und wird blind für Fehler, ...

These

Ein gute Inspektionstechnologie vermeidet die uneffizienten Einarbeitungs- und Ermüdungsphasen.

Inspektionstechniken

- Arbeit »geschickt« auf mehrere Inspekteure mit unterschiedlichen Rollen verteilen.
- Know-How-Weitergabe (Inspekteur ungleich Autor).
- Diversität ausnutzen »Vier Augen sehen mehr als zwei«.

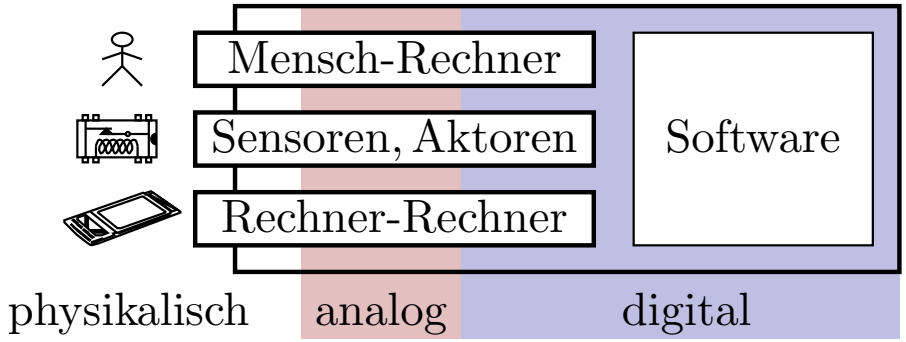
Einteilung der Inspektionstechniken

- Review in Kommentartechnik: Korrekturlesen und Dokument mit Anmerkungen versehen.
- Informales Review in Sitzungstechnik: Lösungsbesprechung in der Gruppe, Vier-Augen-Prinzip. Nimmt die Monotonie, steigert die Aufmerksamkeit, fördert den Wissensaustausch.
- Formales Review in Sitzungstechnik: Festlegen von Rollen (Leser, Moderator, Autor, Inspekteure) und Abläufen, ...

[Lesegeschwindigkeit, Rollendisziplin, Vermeidung Langeweile, überhitzte Emotionen; Reifen Gruppennormen, Sägezahnverlauf]

Funktionstest

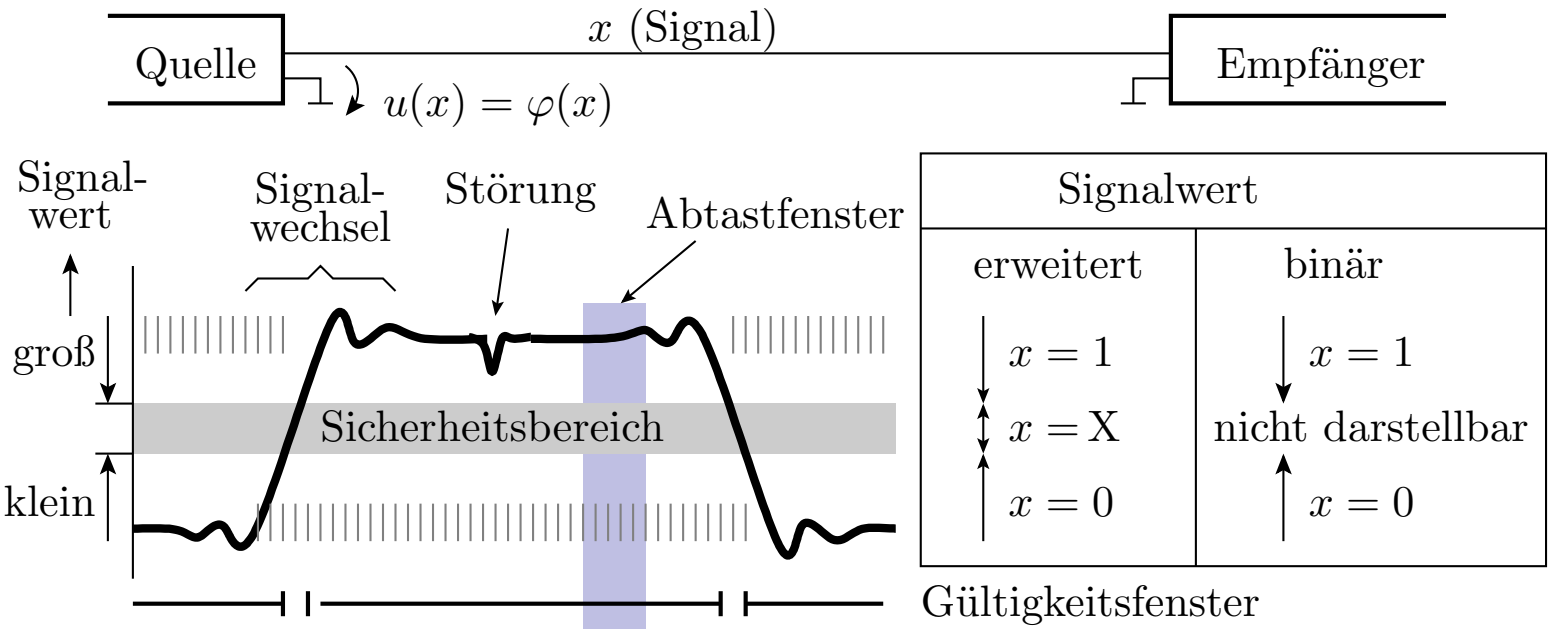
Systemtypen



Ein IT-System kommuniziert mit der Welt über analoge Signale, arbeitet aber intern überwiegend digital.

- Analoge Verarbeitung:
  - Nur an den Schnittstellen zur physikalischen Umgebung.
  - Überschaubare Funktionsvielfalt: Wandlung physikalisch $\Leftrightarrow$ Elektrisch, Verstärkung, Bandbegrenzung, Wandlung analog $\Leftrightarrow$ digital.
- Kompliziertere Verarbeitungsfunktionen werden digital realisiert,
- noch kompliziertere Funktionen in SW.

Digitalisierung



Digitalisierung: Informationstdarstellung durch Bits

- physikalische Werteunterscheidungt nur groß, klein und ungültig,
- Abtastung im Gültigkeitsfenster.
- Fehlertolerant* gegen Verfälschungen durch Rauschen, induktives und kapazitives Übersprechen, Fertigungsstreuungen, Alterung, ...

* Fehlertoleranz: Oberbegriff für die Tolerierung von Bedrohungen aller Art (Fehlfunktionen, Störungen, Fehler, Ausfälle).

Vorteile digitaler Verarbeitung

- Fehlertoleranz gegen Störungen, Fertigungsstreuungen, ...

Trotz der viel größeren Anzahl von Signalen für die Darstellung und Berechnungsschritten (z.B. für eine Addition von zwei Werten):

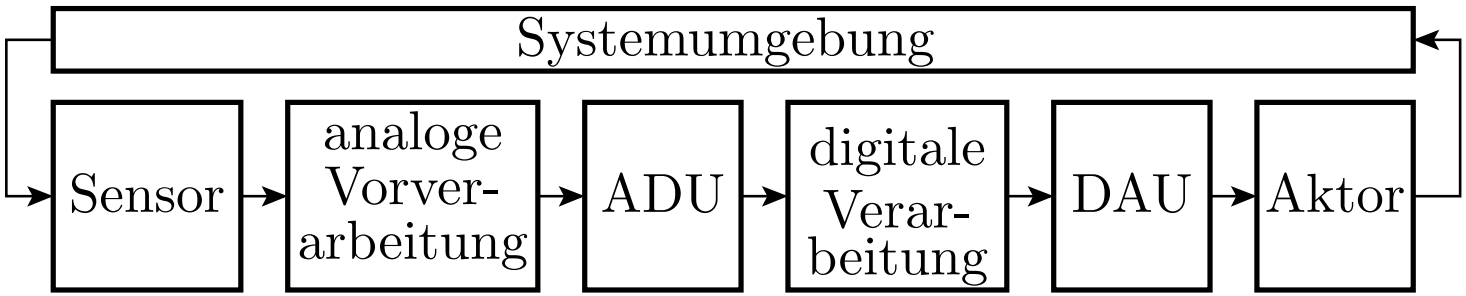
- geringere Fertigungskosten und Entwurfskosten*,
- kleiner, schneller, genauer zuverlässiger.

Zusätzlicher funktionale Gestaltungsspielraum:

- Datenspeicherung,
- sequentielle und SW-gesteuerte Abarbeitung, ...
- Fehlerbeseitigung durch Programmänderung,
- Einschalttest, einprogrammierbare Testhilfen, Überwachungs- und Fehlerbehandlungsfunktionen.
- keine Akkumulation Rauschen und anderer Störungen bei der Verarbeitung und Übertragung, wie bei analogen Signalen,
- ...

* Komplexe Digitalschaltungen werden heute wie Software entworfen.

Typische Verarbeitungskette

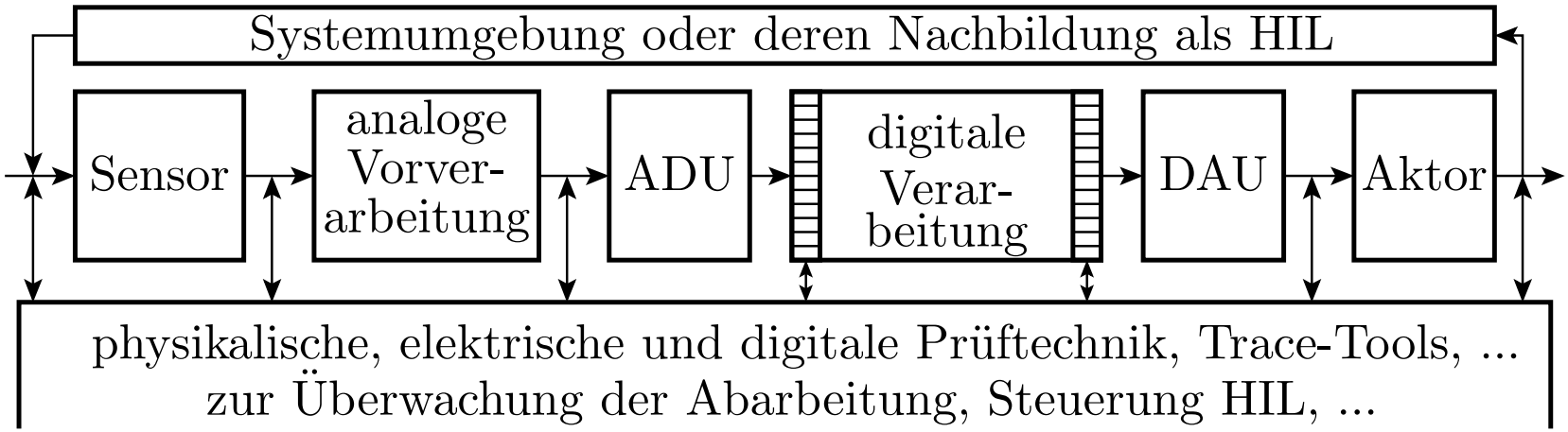


- Sensor, analoge Vorverarbeitung, analog-digital-Wandlung,
- digitale Verarbeitung
  - in der Regel mit Rechnern, durch Software,
  - die Hardware stellt für die SW die Grundfunktionen, zeitkritische Funktionen, ... bereit.
  - Zwischen der HW und der Anwendungs-SW liegt in der Regel noch Schichten: Hardware-Abstraktion, Betriebssystem, ...
- digital-analog-Wandlung,
- analoge Nachbearbeitung, Aktor.

Die oberste Testebene ist das Ausprobieren in der Systemumgebung:

- Bereitstellung physikalischer Eingaben und
- Kontrolle der Ausgaben.

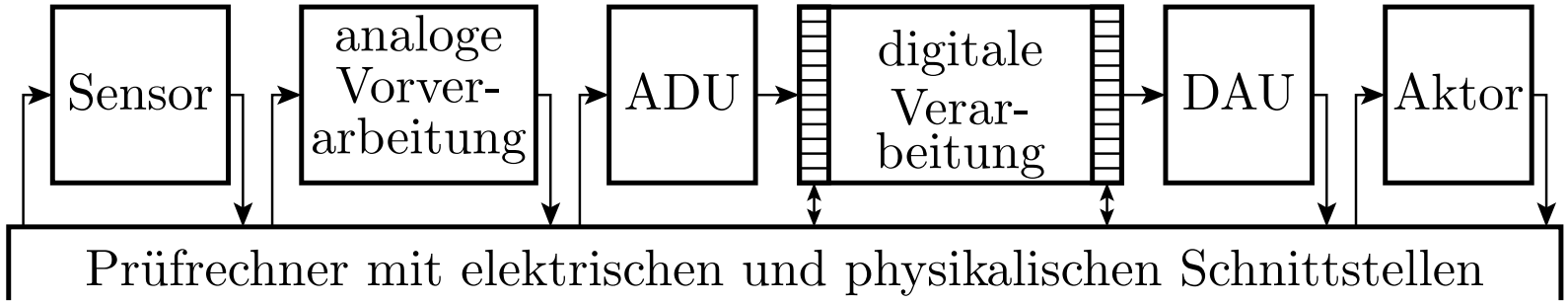
Systemtest, HIL



Das Ausprobieren eines ungetesteten Systems (z.B. Motorsteuerung) in der realen Umgebung kann gefährlich sein. Alternative HIL (Hardware in the Loop, Attrappe für die Systemumgebung).

Die Überwachung der Tests und die Fehlersuche erfordert in der Regel zusätzliche Mittel und Testhilfen zur Beobachtung systeminterner Signale, Daten und Verarbeitungsabläufe.

Isolierte Tests



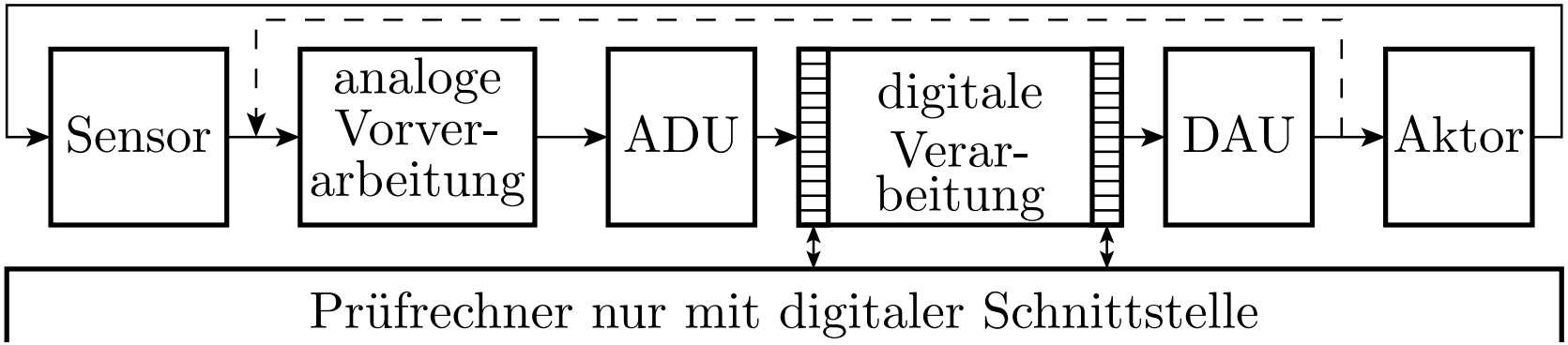
Gründliche isolierter Funktionstests nicht digitale Komponenten (Sensoren, Vorverarbeitung, ADU, DAU und Aktoren):

- durch Kontrolle der Kennlinien, Frequenzgänge, Verzerrungen, ...
- mit analoger Messtechnik

vorzugsweise isoliert von der digitalen Verarbeitung. Erfordert:

- Kontaktierungsmöglichkeiten der Ein- und Ausgänge zum Anschluss geeigneter Prüftechnik,
- Unterbrechungsmöglichkeit des normalen Signalflusses an Testpunkten, ...
- Der Zugriffe auf die bitparallelen digitalisierten Signale erfolgt in der Regel über Testbus seriell (siehe Folie 4.14 *Boundary-Scan*).

Loop-Test



Statt externer Prüftechnik Nutzung der digital-analog- (DAC) und analog-digital-Wandler (ADC) für die Bereitstellung und Auswertung der analogen Testsignale. Erfordert Testmodi mit Signalfluss:

- DAC, (analoge Vorverarbeitung,) ADC und
- DAC analoge Vorverarbeitung, Aktor, Sensor, ADC.

Kontrollen Übertragungsfunktion, Frequenzgang, Verzerrung, Rauschen, ... der entstehenden Loops in den Testmodi erfordert:

- ausreichend schnelle und genaue ADC und DAC,
- Umschalter im Signalfluss, die die Signale nicht verfälschen.
- Auch nutzbar für Selbst-, Einschalt- und Wartungstests.

Prüfgerechter Entwurf

- Anschluss von Prüftechnik zur Überwachung von Systemtests,
- HIL (Hardware-in-tho-Loop),
- isolierte Tests analoger Komponenten,
- Loop-Tests, ...

verlangen Vorkehrungen, die ab Beginn des Entwurfsprozesses mit berücksichtigt werden müssen. Das gilt auch, wie im weiteren gezeigt, für den Test der digitalten Verarbeitung, SW-Tests und Selbsttests.

Prüfgerechter Entwurf

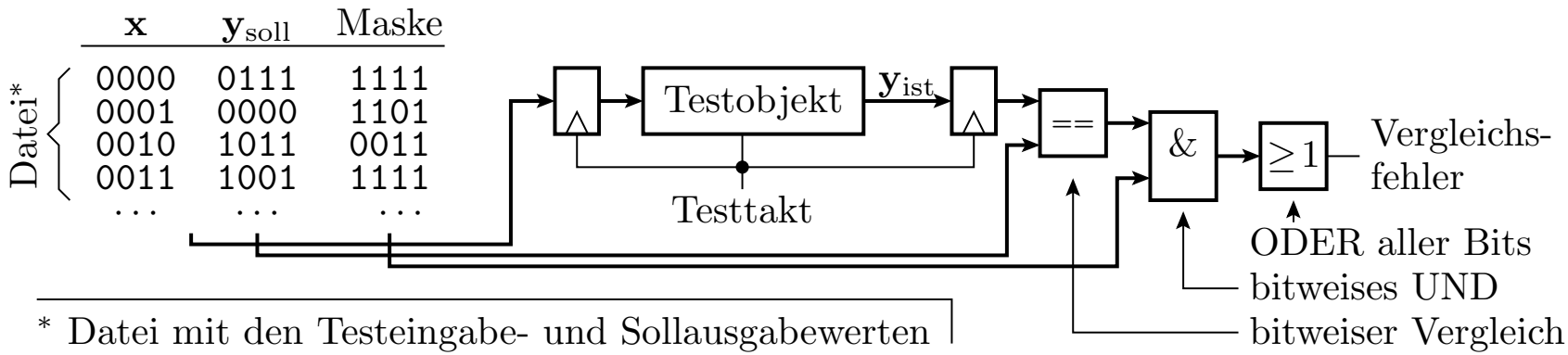
Sammlung von Maßnahmen, um Tests entwerfen und durchführen sowie die Testergebnisse auswerten zu können.

Die Art der Testdurchführung muss schon sehr früh in den Entwurfsprozess für ein neues System einfließen.

Ohne prüfgerechten Entwurf kein ausreichender Test, keine ausreichende Verlässlichkeit und damit nicht nutzbar.

Digitale Schaltung

Test digitaler Bausteine



Wiederhole für jeden Taktschritt des Tests:

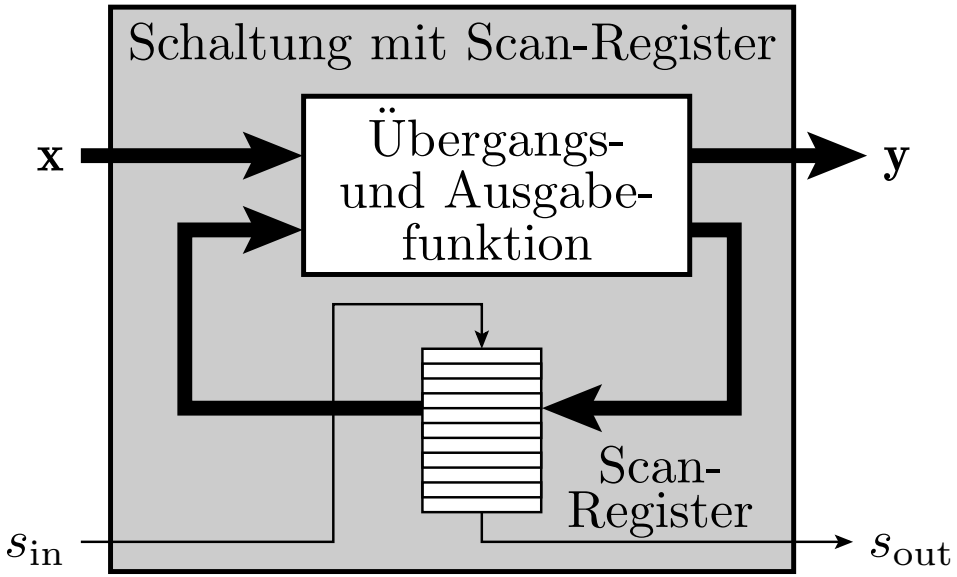
- Bereitstellung logischer Eingabewerte und
- Abtasten und Auswertung der vorherigen Ausgaben.

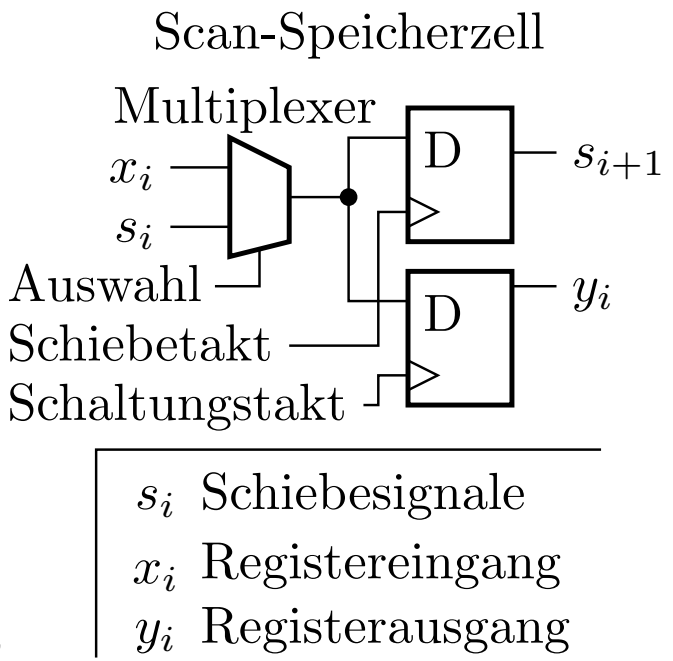
Auswertung vorzugsweise Vergleich mit Sollwerten unter Ausmaskierung von Ausgabebits ohne definierten Sollwert.

Erweiterungen:

- Speicherelemente im Testobjekt sind vor dem Test zu initialisieren.
- Kontrolle der Signalverzögungen durch Ergebnisabtastung mit mehrfacher Aufzeichnungsfrequenz.
- Tester auch als Kombination Signalgenerator Logikanalysator.

Scan-Verfahren

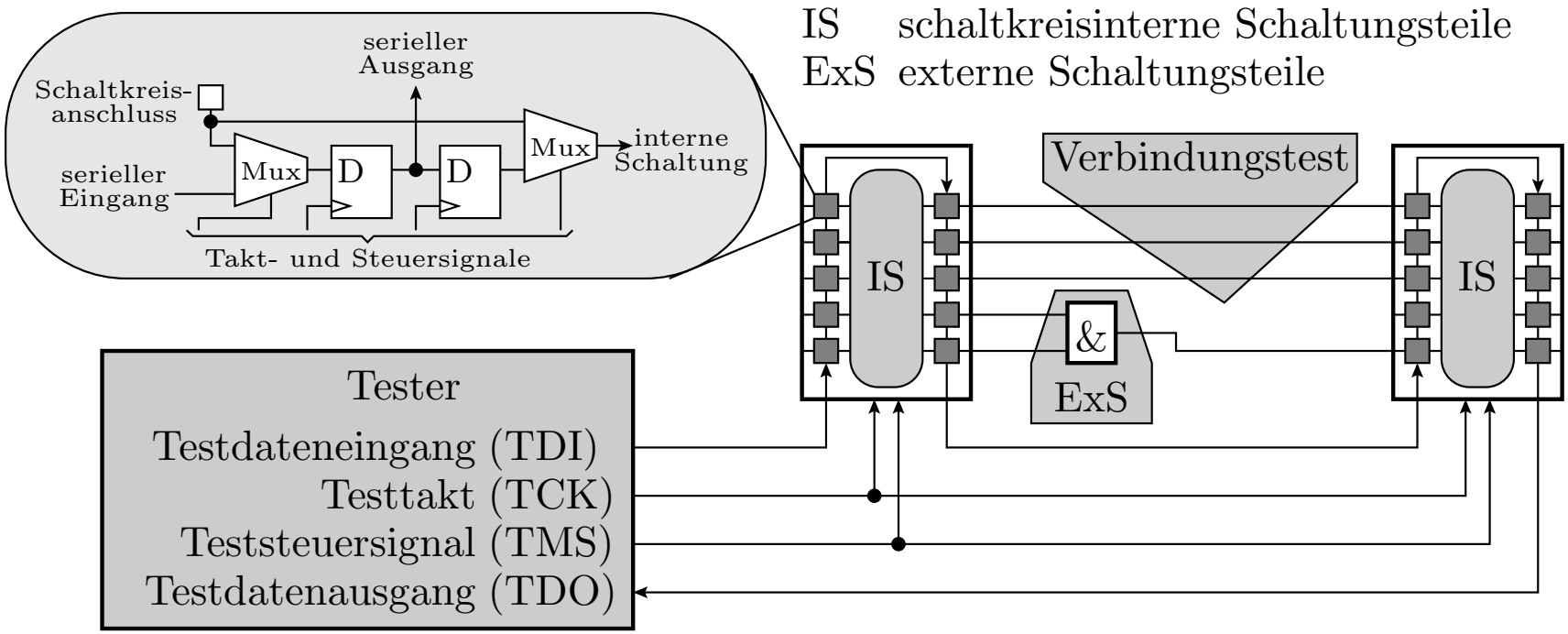




Vereinfachung von Fehlernachweis und -lokalisierung durch Lese- und Schreibzugriff auf interene Signale und Speicher. Datentransfer vorzugsweise seriell. Im Bild ist jede Speicherzelle um einen Multiplexer und eine Schiebezelle erweitert zur Bereitstellung der Funktionen:

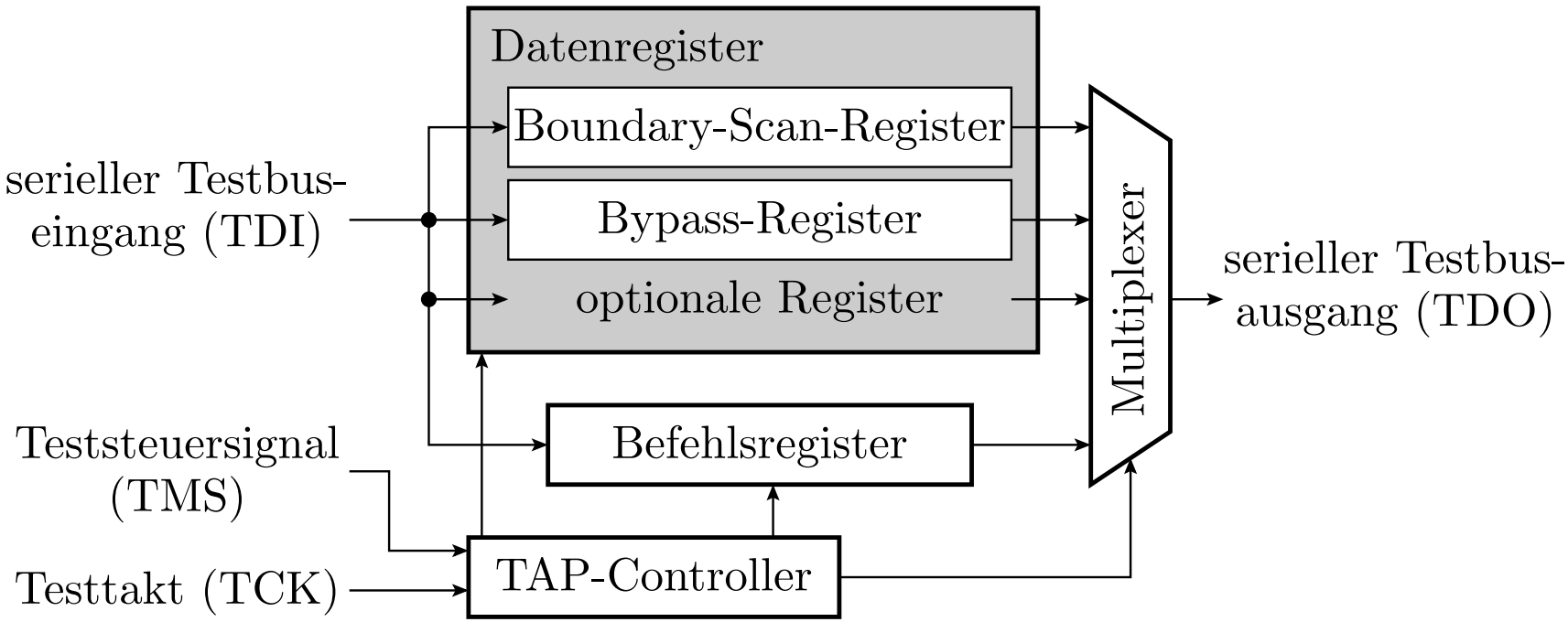
- Capture: Übernahme aus der Schaltung in die Schiebezellen,
- Shift: serielles Auslesen und neu beschreiben und
- Update: Übergabe seriell eingelesene Daten an Schaltung.

Boundary-Scan



- Scan-Registern an den Anschlüssen digitaler Schaltkreise.
- Ermöglicht den isolierten Test der schaltkreisinternen Funktion und Schaltkreisumgebung auf Baugruppen (siehe Abschn. 4.1.5).
- Der standardisierte JTAG-Testbus besitzt 4 Bussignale: TDI, TDO, TCK und TMS zur Verkettung der Testbusse vieler Schaltkreise.

JTAG-Testbusarchitektur der Schaltkreise



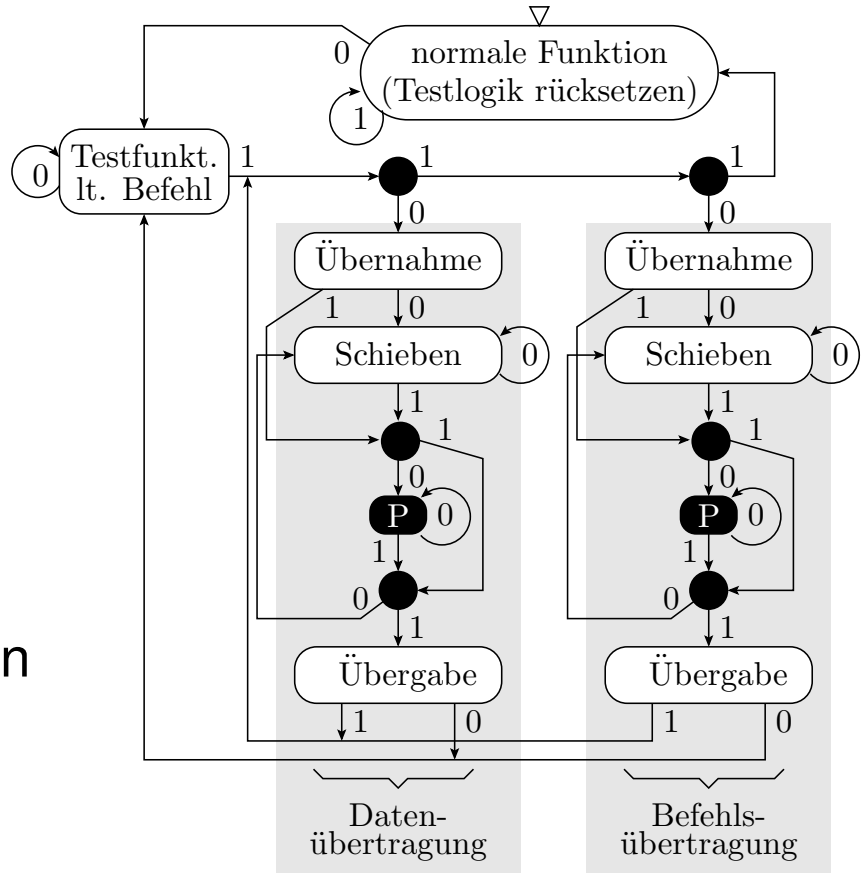
Eine Boundary-Scan-Implementierung umfasst:

- 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 (**T**est **M**ode **S**elect)



Typischer Testbeginn:

- Befehlsregister lesen (Kontrolle JTAG-Kette*),
- 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

* Übernahme 01 in Bit0 und Bit1 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, ...

Software

Tesrahmen

Zu testende Programmbausteine werden für den Test in einem Programmrahmen eingebettet, der Testeingaben bereitstellt und Testausgaben auswertet und mit dem er gemeinsam in ein ausführbaren Programm übersetzt wird.

Im einfachsten Fall ist der Testrahmen ein ausführbares Programm und das Testobjekt ein aufgerufenes Unterprogramm.

Nützliche Funktionen zur Unterstützung von Test und Fehlersuche (bereitzustellen von Systemsoftware und Hardware):

- Schrittbetrieb, Haltepunkte,
- Lesen und Verändern von Variablen im Haltezustand,
- Trace-Aufzeichung von Variablen und Ausgabesignalen, ...

Ein Testobjekt und sein Testrahmen

Beispieltestsobjekt: Unterprogramm zur Quadrierung:

```
uint32_t quad(int16_t a){ // Quadratberechnung
  return (uint32_t)a * a; // Warum mit Typcast?
};
```

Testbeispiele sind Tupel aus Eingaben und Sollausgaben. Man kann dafür einen neuen Datentyp definieren:

```
typedef struct{
  int16_t x;                    //Eingabe
  uint32_t y;                   //Sollausgabe
} test_t;
```

Ein »struct« ist eine Zusammenfassung aus bereits definierten Datentypen.

Testsatz

Eine Testsatz als Menge von Tests ist im einfachsten Fall ein initalisiertes Feld von Testbeispielen

```
test_t testsatz[] ={{<Tupel1>},{...},...};
```

Testbeispiele für die Quadratberechnung:

```
test_t testsatz[] ={        //Eingabe   Sollwert
  {0, 0},                   // 0x0000   0x00000000
  {1, 1},                   // 0x0001   0x00000001
  {9, 81},                  // 0x0009   0x00000051
  {-5, 25},                 // 0xFFFB   0x00000019
  {463, 214369},            // 0x01CF   0x00034561
  {0x7FFF, 1073676289}      // 0x7FFF   0x3FFF0001
};
```

Welche Testbeispiele führt das Programm falsch aus? Ausprobieren!

Testrahmen

Programm, das in einer Schleife alle Testbeispiele abarbeitet und die Ergebnisse kontrolliert oder zur Kontrolle ausgibt:

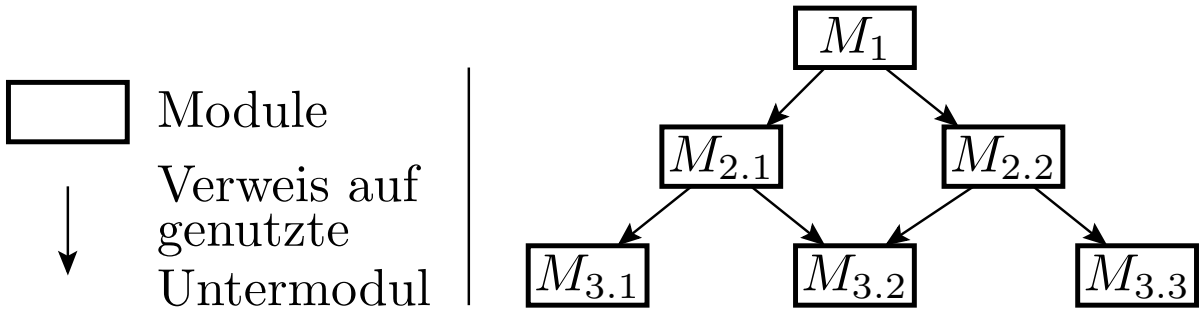
```
int main(){
 uint8_t idx, err_ct=0;
 uint32_t erg;
 for (idx=0; idx<6;idx++){
 erg = quad(testsatz[idx].x);//Istwert
 if (erg != testsatz[idx].y) //Soll/Ist-Vergl.
  err_ct++;                  //Fehlfkt. zählen
 }
}
```

Testdurchführung mit dem im Simulator im Debug-Modus:

- Unterbrechungspunkt vor dem Fehlerzähler.
- Bei jeder Fehlfunktion Halt am Unterbrechungspunkt.

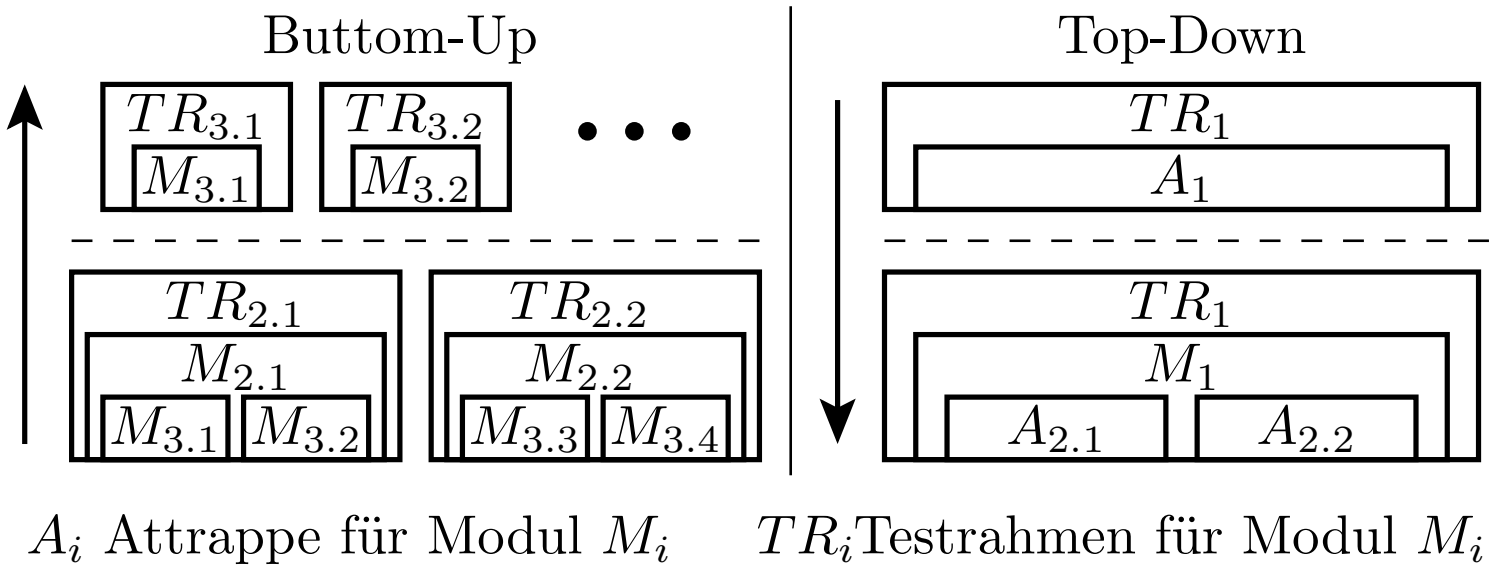
Modularisierung und Testrahmen

Jeder Code-Baustein muss ausprobiert werden:



Strategien für die Entwurfs- und Testreihenfolge:

- Bottom-Up: Beginn mit dem Entwurf und Test der untersten Module. Test der übergeordneten Module mit den bereits getesteten Untermodulen.
- Top-Down: Beginn mit dem Entwurf übergeordneter Module und Test mit Attrappen für die Untermodule. Schrittweise Ersatz der Attrappen durch getestete Untermodule.

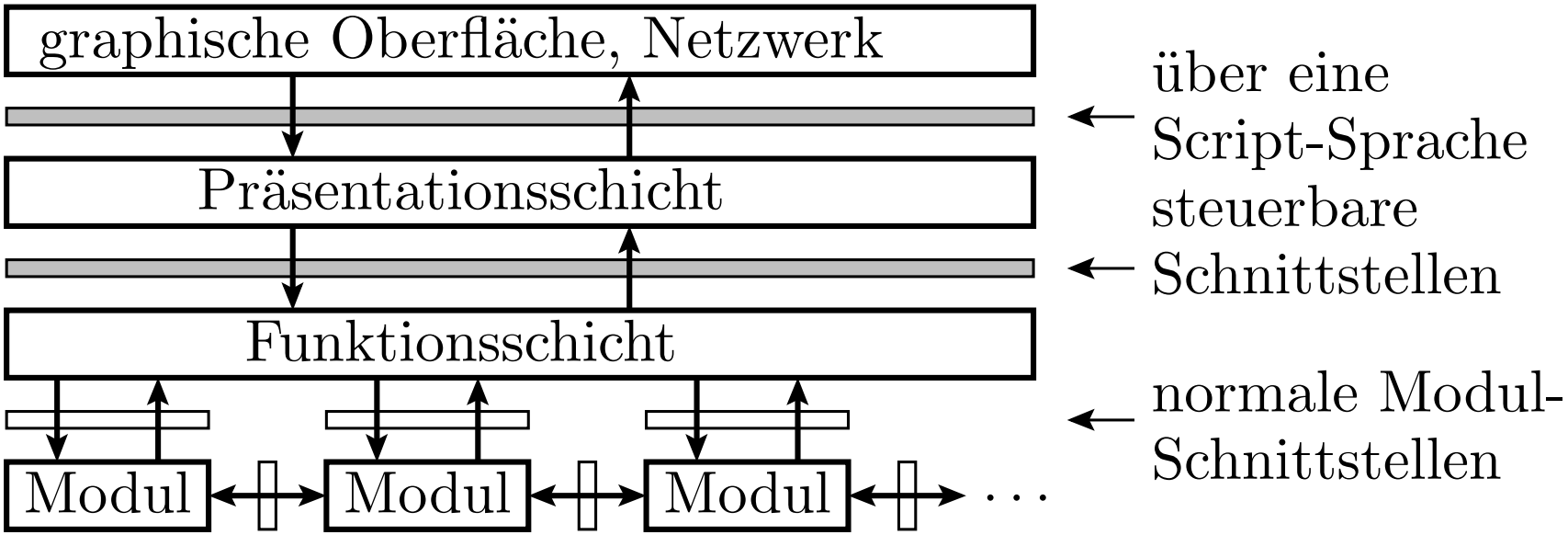


Praktisches Vorgehen:

- erst beispielbasierte Tests mit Ergebnisausgabe, um das Testobjekt zu untersuchen,
- dann Erweiterungen auf zielgerichtete Kontrolle zuzusichernder Eigenschaften,
- Ergänzung Fehlerbehandlung im Testobjekt und Tests dafür,
- dann Fussifizierung, um ungewollte Eigenarten aufzudecken.

Je mehr Attrappen der Test erfordert, um so schlechter ist der Code.

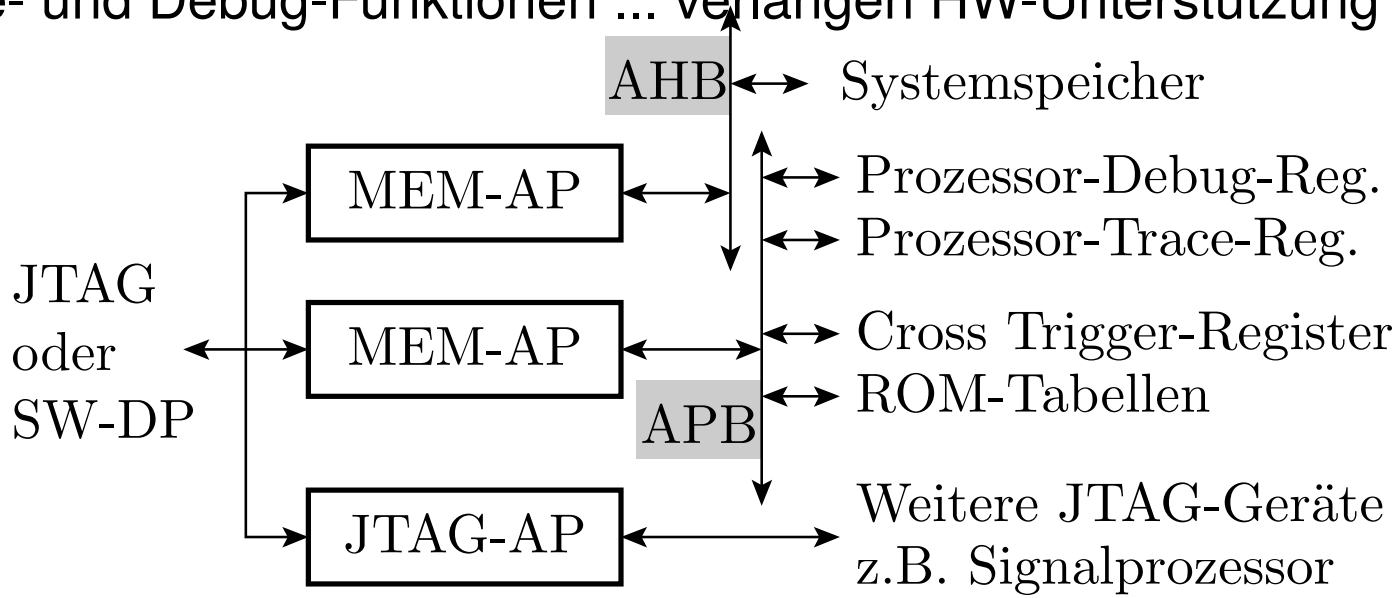
Schichtenmodell



- Alle Funktionsbausteine sind einer Schicht zugeordnet.
- Einer höhere Schicht - z.B. Benutzeranwendung wie Excel oder Word - kann nur Prozeduren der Schicht darunter über eine wohl definierte Schnittstelle (API) nutzen.
- Ein Kommunikationsprogramm kann z.B. nicht direkt, sondern nur über eine Betriessystemfunktion auf den COM-Port zugreifen.
- Die Schichten bieten oft Script-Sprachen, um Tools (Funktionen der Schicht) zu steuern und damit auch Testskripte zu schreiben.

ARM-Testbusarchitektur

Einige Trace- und Debug-Funktionen ... verlangen HW-Unterstützung



AHB	Advanced High Performance Bus
APB	Advanced Peripherial 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 Befehlbit + 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-Operationen[1].
- 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.

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

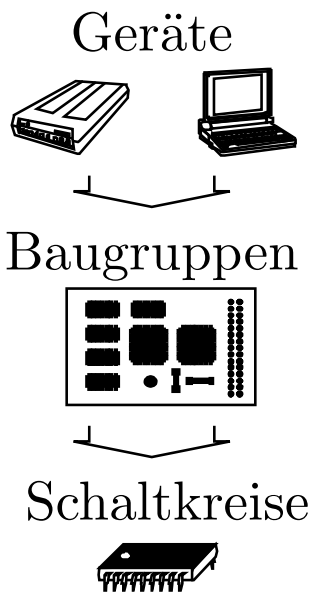
Leiterplatten

Hierarchie und Test

Rechnerhardware besteht aus tauschbaren Komponenten:

- tauschbare Teilsysteme.
- tauschbare Leiterplatten,
- austauschbare Bauteile.

Die Komponenten werden vor Einbau in das übergeordnete System gründlich getestet.

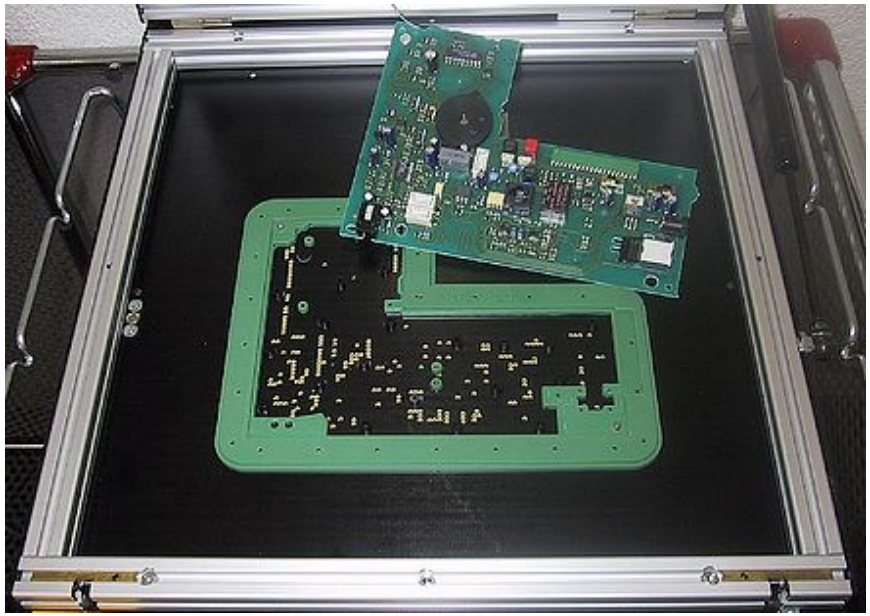


Wiederholungsfragen

1. Warum tauschbare Komponenten bzw. unter welcher Bedingung nicht erforderlich (nicht zweckmäßig)?
2. Warum ist ein gründlicher Komponententest zu fordern?
3. Warum beginnt ein BG Test mit einem statischen Bestückungs- und Verbindungstest?

Bestückungs- und Verbindungstests

Hauptfehler auf Baugruppen sind Kurzschlüsse und Unterbrechungen. Nachweis durch Widerstandsmessungen zwischen und entlang der Verbindungen.
In der Serienfertigung erfolgt die Kontaktierung mit einem mit Unterdruck angesaugten Nadeladapter.



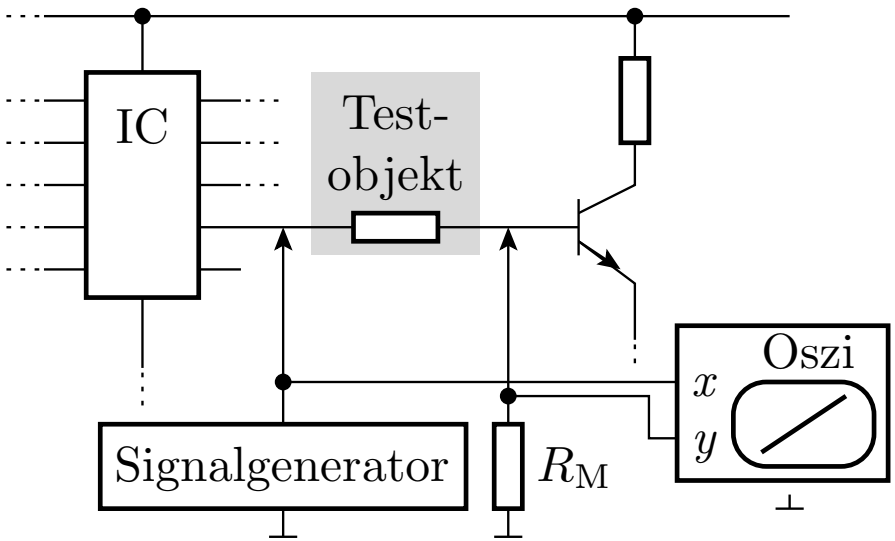
Die Nadeln sind mit reine Relais-Matrix zur Verbindung mit den Prüfgeräten angeschlossen. Auch Bestückungsfehler lassen sich überwiegend mit Zweipunktmessungen von Strom-Spannungsbeziehungen erkennen. Fehlerlokalisierung im Vergleich zu der für einen dynamischen Test, der versagt, sehr einfach.

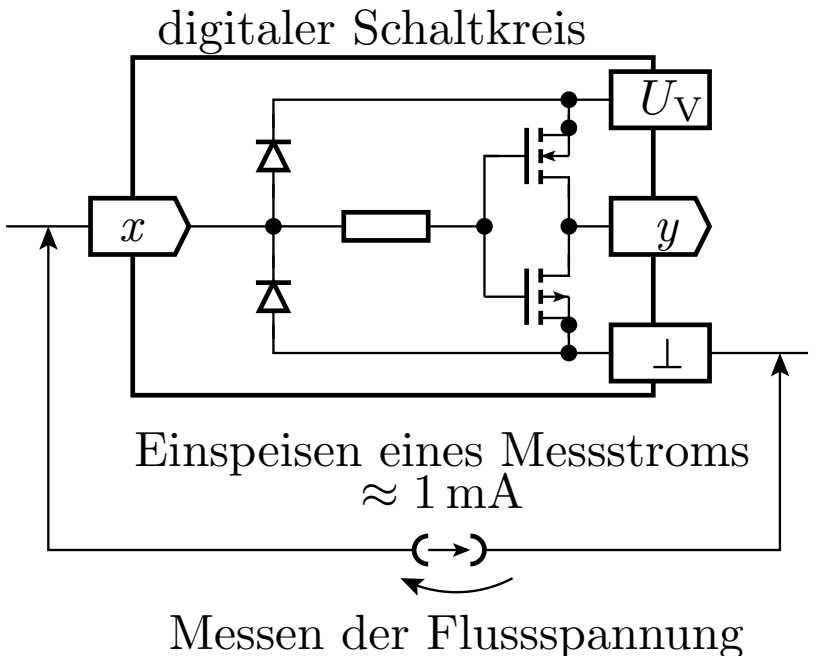
Bestückungstests

Kontrolle auf Bestückungsfehler durch Überprüfung ausgewählter Zweipunktmerkmale:

- Widerstandswerte,
- Kapazitäten,
- Flussspannungen von Dioden, ...

Zu Kontrolle, dass Schaltkreisanschlüsse mit dem Kontakt-Pad verbunden sind, werden die Schutzdioden zur Versorgungsspannung und Masse ausgemessen.

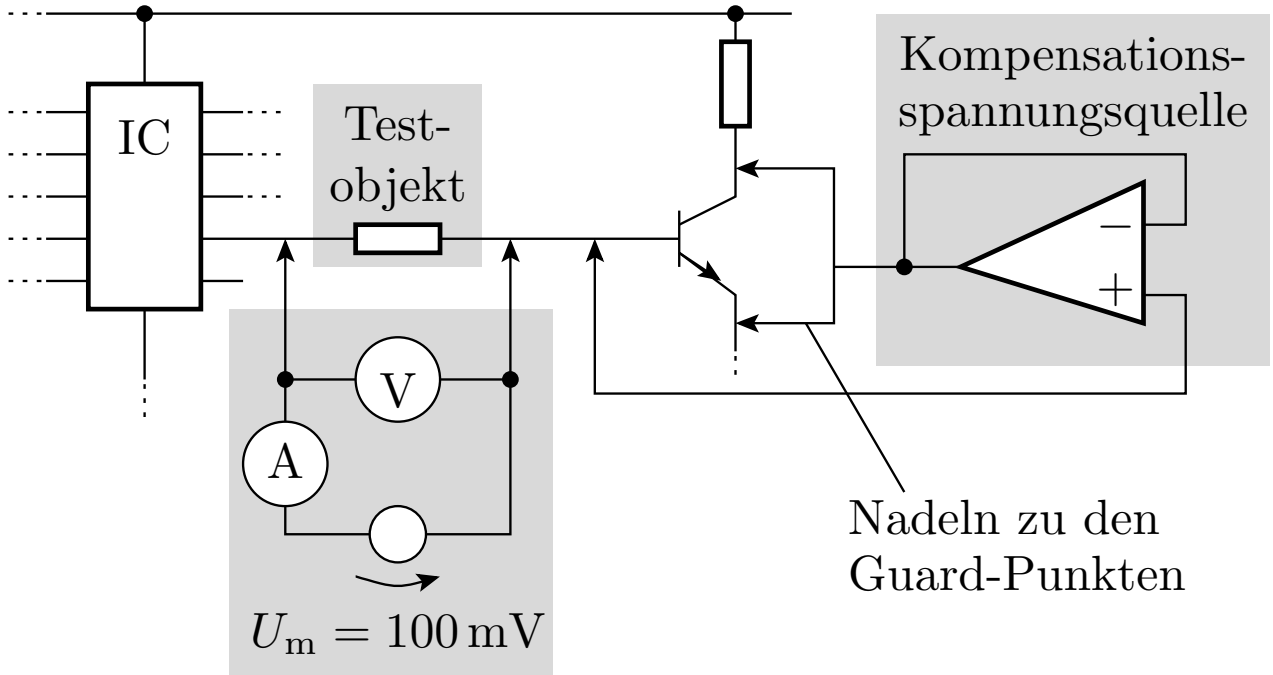


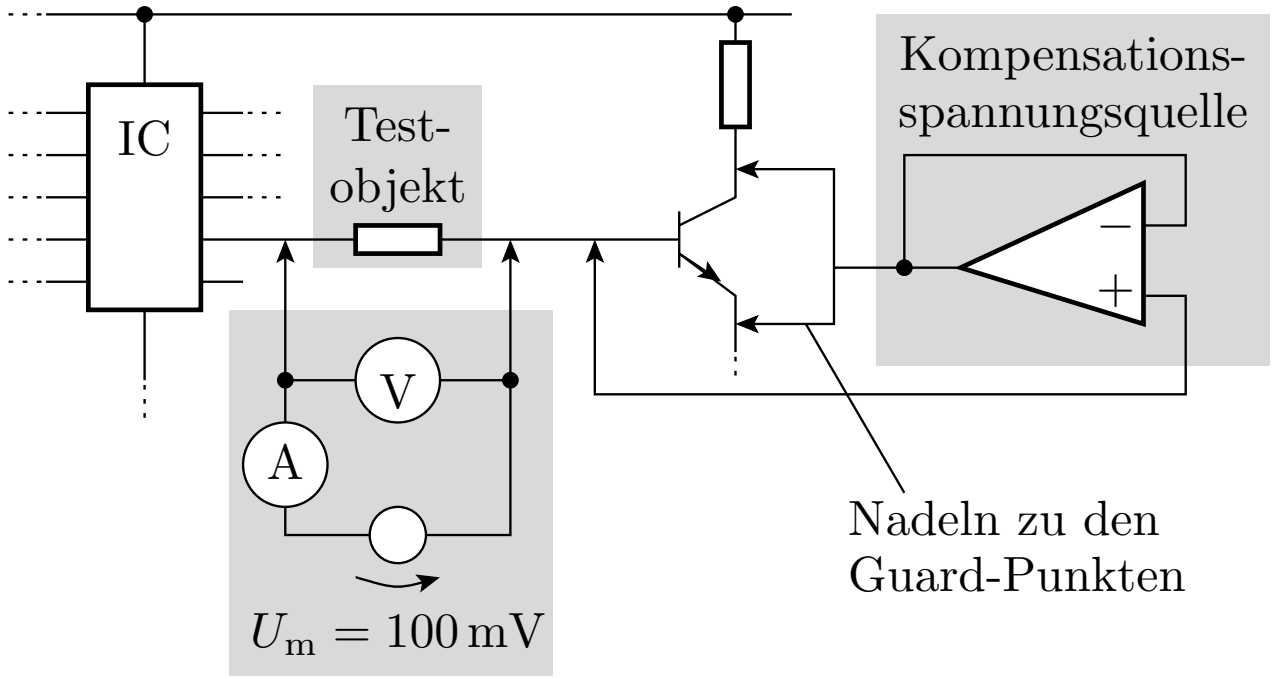


Analoger In-Circuit Test

Die Strom-Spannungs-Beziehung zwischen zwei Punkten hängt nicht nur vom Bauteil zwischen den Nadeln, sondern von allen Strompfaden, im Beispiel durch Transistor und Schaltkreis ab. Problematisch:

- die Toleranzbereiche der Sollwerte mit allen Bauteilstreuungen,
- die Erkennungssicherheit für Fehlbestückungen, z.B. bei sehr kleinen Kapazitäten.

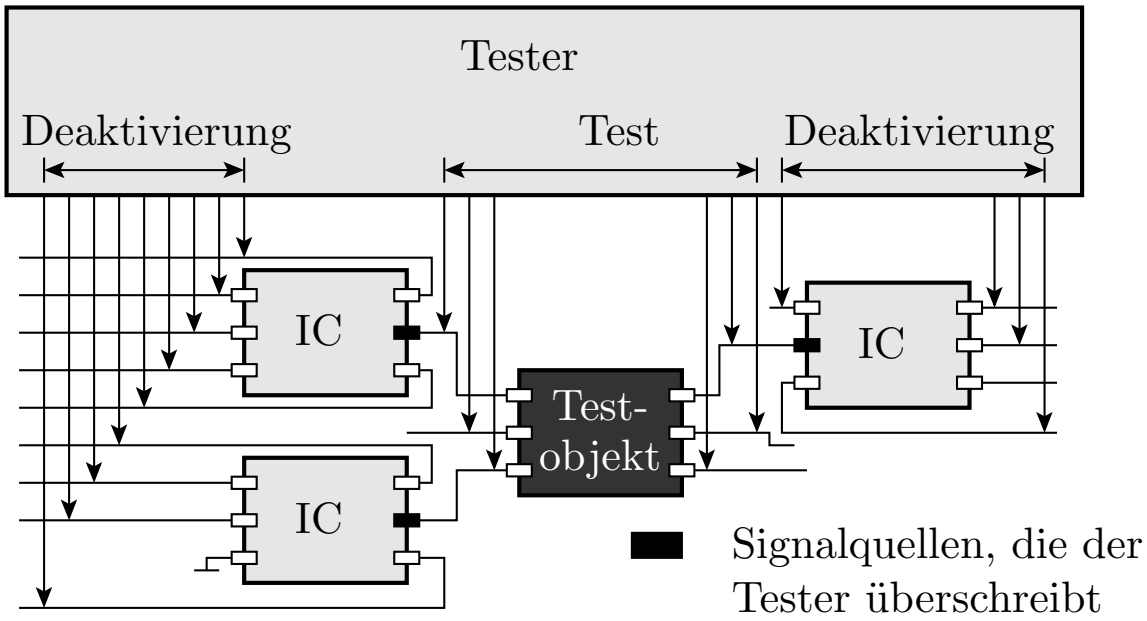




Unterdrückung von Parallelströmen zum Testobjekt durch Kompensation der Spannungsabfälle über den wegführenden Bauteilen auf einer Testobjektseite auf null über »Guard-Punkte«:

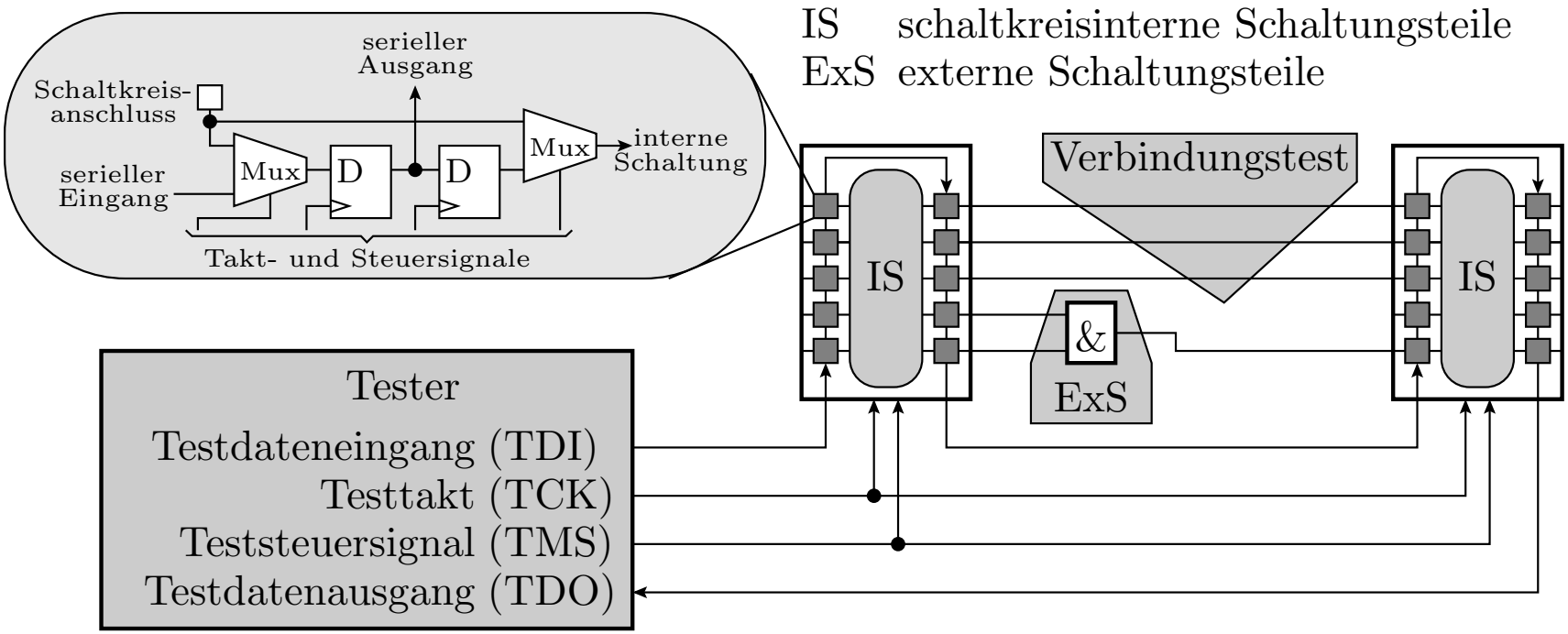
- Erlaubt einen isolierten Zweipoltest.
- Vereinfacht die Testprogrammerstellung, insbesonderen die Sollwert- und Sollwerttoleranzfestlegung.
- Mindert die Häufigkeit von Fehlklassifikationen.

Digitaler In-Circuit-Test



- Kontaktierung der Baugruppe über ein Nadelbetadapter.
- Isolierter Test der Schaltkreise durch Überschreiben der digitalen Schaltkreiseeingaben mit stromstarken Treibern.
- Im Gegensatz zum analogen ICT unter Spannung.
- Andere Schaltkreise werden möglichst deaktiviert (Anschlüsse hochohmig).

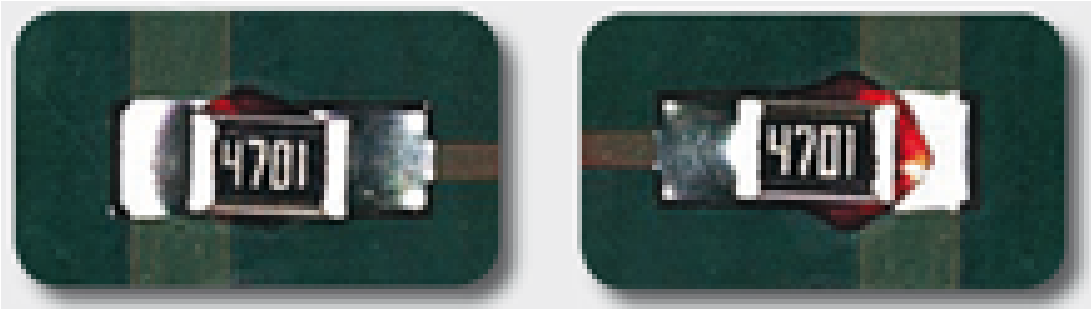
Boundary-Scan



Scan-Register zum Lesen und Überscheiben der logischen Werte an den Schaltkreisanschlüssen über einen seriellen Bus (siehe Folie 4.14 *Boundary-Scan*). Ersatz der mechanischen Nadeln durch »silicon nails«. Alternative zu den teuren, für jede Baugruppe speziell anzufertigenden Nadeladaptern.

Optische Inspektion

Es gibt Bestückungsfehler, die sind optisch, aber nicht elektrisch erkennbar. Bild links korrekt bestückter SMD-Widerstand, rechts Lötfläche durch Kleber verschmutzt. Elektrisch leitende aber keine feste Lötverbindung:



Nachweis nur durch visuelle Kontrolle möglich. Nach Ausfall der Baugruppe z.B. durch Vibration in einem Fahrzeug ist sofort erkennbar, dass es sich um einen optisch nachweisbaren Fertigungsfehler handelt.

Wenn ein solcher Fertigungsfehler Schaden verursacht, z.B. einen Unfall, greift die Produkthaftung, d.h. der Hersteller muss für den entstandenen Schaden aufkommen.

Kontrollfragen

- Zählen Sie Maßnahmen des prüfgerechten Entwurfs für den Baugruppentest auf.
- Was bedeutet isolierter Test von Komponenten und wie wird dieses Prinzip mit dem In-Circuit-Test umgesetzt?
- Warum ist für SMD-bestückte Baugruppen von Steuergeräten für KFZ eine optische Inspektion zwingend?
- Welche prüftechnische Problem des Baugruppentests löst Boundary-Scan?

Zusammenfassung

Inspektion

Sichtprüfungen, anwendbar auf Entwurfsbeschreibungen, Programmcode, Testausgaben, ... Kenngrößen

- Test: Fehlerüberdeckung, Phantom-MF-Rate,
- Technologie: Effizienz (gefundene Abweichungen pro Mitarbeiterstunde) und Effektivität (gefundene Abweichungen je NLOC).

Capture-Recapture

Schätzer für die zu erwartende Fehleranzahl und Fehlerüberdeckung:

$$\#F = \frac{\#F_1 \cdot \#F_2}{\#(F_1 \cap F_2)} \tag{4.1}$$

$$FC = \frac{\#(F_1 \cap F_2) \cdot \#(F_1 \cup F_2)}{\#F_1 \cdot \#F_2} \tag{4.2}$$

Wahrscheinlicher Bereich der geschätzten Fehleranzahl:

$$\frac{\mathrm{sr}(\mathbb{E}[\#F])}{\#F}) \approx 1 \mp \sqrt{\frac{1}{\#F_1} + \frac{1}{\#F_2} + \frac{1}{\#(F_1 \cap F_2)} - \frac{1}{\min(\#F_1, \#F2)}} \cdot \Phi^{-1}\left(1 - \frac{\alpha}{2}\right) \tag{4.3}$$

Abschätzung des wahrscheinlichen Bereichs der FC am Beispiel.

Modellfehlerbasierte Abschätzung

Schätzer für die zu erwartende Fehlerüberdeckung und Fehleranzahl:

$$FC = FC_{\mathrm{M}} = \frac{\#DF_{\mathrm{M}}}{\#F_{\mathrm{M}}} \tag{4.4}$$

$$\#F = \frac{\#DF}{FC} = \#F_{\mathrm{M}} \cdot \frac{\#DF}{\#DF_{\mathrm{M}}} \tag{4.5}$$

Wahrscheinliche Bereiche:

$$\mathrm{sr}\left[\mathbb{E}\left[FC\right]\right] = FC \cdot \left(1 \mp \sqrt{\frac{1}{\#DF_{\mathrm{M}}}} \cdot \Phi^{-1}\left(1 - \frac{\alpha}{2}\right)\right) \tag{4.6}$$

$$\mathrm{sr}\left[\mathbb{E}\left[\#F\right]\right] = \#F \cdot \left(1 \mp \sqrt{\frac{1}{\#DF_{\mathrm{M}}} + \frac{1}{\#DF}} \cdot \Phi^{-1}\left(1 - \frac{\alpha}{2}\right)\right) \tag{4.7}$$

Bei gleicher Zählwertgröße genauer als »Capture-Recapture«. Vermeidung der systematischen Fehler durch Kommunikation zwischen den Inspekteuren und abweichende der Fehlererkennungswahrscheinlichkeiten.

Inspektionsdauer und Effizienz

Pareto-Verteilung der Nachweiszeit:

$$F_T(t_{\text{Insp}}) = \begin{cases} 0 & t_{\text{Insp}} \leq t_{\min} \\ 1 - \left(\frac{t_{\min}}{t_{\text{Insp}}}\right)^k & \text{sonst} \end{cases} \tag{4.8}$$

Effizienzabnahme bei pareto-verteilerter Nachweiszeit:

$$\frac{EFC(t_{\text{Insp}})}{\#DF(t_{\min})} = f_T(t_{\text{Insp}}) = \frac{\mathrm{d}F_T(t)}{\mathrm{d}t} = \frac{k \cdot t_{\min}^k}{t_{\text{Insp}}^{k+1}} \tag{4.9}$$

Ein Experiment und die allgemeine Erfahrung haben gezeigt, dass die Effizienz einer Inspektion statt dessen ineffiziente Einarbeitungs- und Ermüdungsphasen kennt. Eine gute Inspektionstechnologie vermeidet diese durch geschickte Arbeitsorganisation.

Inspektionstechniken: unterschiedliche Kompromisse zwischen Kreativitätsbeschränkung, Effizienz und Ergebnisvorhersagbarkeit:

- max. kreative Freiräume: Review in Kommentartechnik
- max. Ergebnisvorhersagbarkeit: formales Review in Sitzungstechnik.

Funktionstest

- IT-Systeme werden überwiegend digital realisiert, vor allem auch wegen Toleranz gegenüber Störungen und andere Probleme.
- Analoge Verarbeitung ist im wesentlichen auf die Ein- und Ausgabe begrenzt und wird beim Systemtest und isoliert vom übrigen System überprüft.
- Wenn Ausprobieren in der Systemumgebung problematisch (gefählich, schwierig, ...) auch Nachbildung der Systemumgebung durch Attrappen (HIL, Simulation).
- Prüfgerechten Entwurf, Sammlung von Voraussetzungen, gute Tests durchführen zu können. Kontaktiermöglichkeit für Messtechnik, Testmodi, ...
- Für den Test der analogen Verabeitungsketten bieten sich Loop-Tests mit Ein- und Ausgabe an der digitalen Peripherie an. Später in der Einsatzphase auch als Einschalt- und Wartungstests nutzbar.

Test digitaler Schaltungen

- Bereitstellung logischer Eingabewerte und Kontroll der Ausgabebits, vorzugsweise durch Vergleich mit Sollwerten.
- Kontrolle von Signallaufzeiten durch mehrfache Abtastung nach jeder Eingabeänderung.
- Zur Erhöhung der Fehlerüberdeckung und Vereinfachung der Testauswahl Einbau von Lese- und Schreibfunktionen für interne Speicher und Signale z.B. mit Scan-Registern.
- Datenaustausch mit Tester vorzugsweise seriell über Testbus.
- Boundary-Scan: Scan-Ring um den Schaltkreis als Ersatz für die Stecker oder Prüfspitzen zur mechanischen Kontaktierung.

JTAG-Testbus:

- Serieller Testbus für den Testerzugriff auf die Boundary-Scan-Ketten und andere Testhilfen .
- Außer für den isolierten Test der Leiterplattenverdrahtung, auch Steuerung interner Test- und Debug- und Programmierfunktionen.

Software-Test

Einbettung der Testobjekte in Programmrahmen, die Eingaben bereitstellen, Ausgaben kontrollieren und den Ablauf steuern.

Nützlich Hilfmittel für gründliche Kontrollen und die Fehlersuche:

- Debugger mit Schrittbetrieb, Haltpunkten, Lese- und Schreibzugriff auf interne Daten.
- Trace-Aufzeichung von Variablen und Ausgabesignalen, ...

Modularisierung, Testrahmen und Attrappen:

- Botton-Up: Entwurf und Test beginnend von den kleinsten zu immer größeren Bausteinen mit je einem zu entwerfenden Testrahmen je Baustein.
- Top-Down: Entwurf und Test beginnend mit dem Gesamttestrahmen, Testrahmen mit oberster Verarbeitungsschicht, ... Ersatz der fehlenden Bausteine durch Attrappen.

SW-Test: Schichten und ARM-Testbus

Schichten bieten vereinheitlichte Testschnittstellen für der Schicht zugeordnete Funktionen, teilweise sogar Script-Sprachen zur Programmierung von Tests.

Die beispielhaft besprochene ARM-Testbus-Architektur bietet sämtliche Test-, Programmier- und Debug-Schnittstellen für die Inbetriebnahme und die hardware-nahe Fehlersuche:

- Speicher- und Registerzugriff,
- Debugger-Funktionen,
- Trace-Aufzeichung, ...

Leiterplattentest

- Leiterplatten bestehen aus gründlich vorher getesteten Bauteilen. Test hauptsächlich auf Bestücks- und Verbindungsfehler.
- Bei größeren Stückzahlen Kontaktierung mit Nadelbettadapter. Bestückungs- und Verbindungstests spannungsfrei über elektrische Zweipunkmessungen.
- Der elektrische In-Circuit-Test verwendet zusätzlich Guard-Punkte zur Unterdrückung von Parallelströmen zum zu testenden Bauteil.
- Digitaler In-Circuit-Test ist im Gegensatz dazu ein Test unter Betriebsspannung, bei dem digitale Signalwerte auf den Verbindungen mit stromstarken Treibern auch kurzzeitig überschrieben werden.
- Wegen zunehmender mechanischer und elektrischer Probleme mit wachsender Schaltungsgröße und Packungsdichte Trend zum Ersatz der mechanischen Nadeln durch Boundary-Scan, d.h. in die HW eingebauten Lese- und Schreibfunktionen der logischen Werte an den Schaltkreisanschlüssen.

- Bestückungskontrolle durch Lesen der Hersteller- und Bauteilidentifikationsnummern.
- Optische Inspektion für typische elektrisch nicht nachweisbare aber die Lebenserwartung beeinträchtigende Verbindungsfehler.

Überwachung

Vergleich

Wiederholung Wertekontrollen

Die behandelten Verfahren für Wertekontrollen
- Soll-Ist-Vergleiche zur Ergebniskontrolle dynamischer Tests,
- Mehrfachberechnung und Vergleich und Loop-Test

haben zwei Quellen für Klassifizierungsfehler:
- falsche Sollwerte und
- Fehlfunktionen beim Vergleich

mit jeweils beiden Möglichkeiten der Fehlklassifizierung:
- Maskierung (Klassifizierung falscher Werte als richtig) und
- Phantom-MF (Klassifizierung korrekter Werte als falsch).

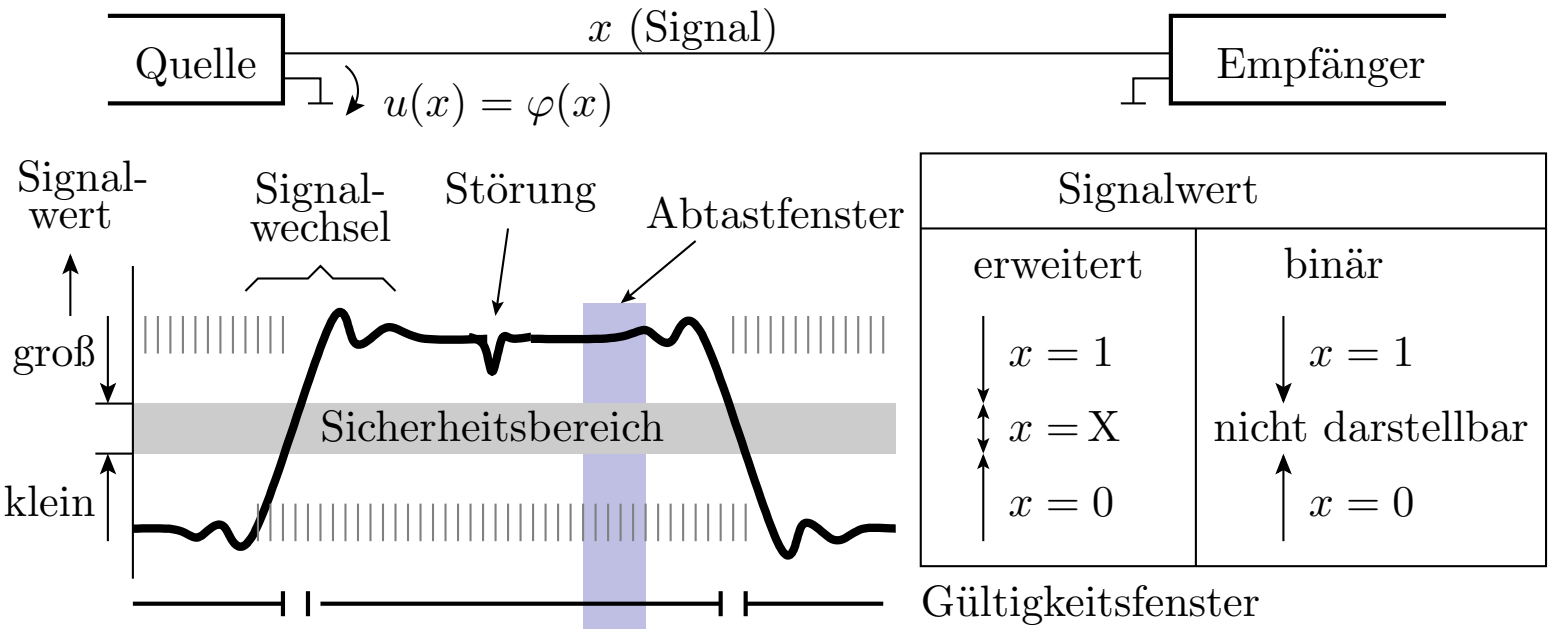
Fehlklassifizierung »Verdoppl. und Vergleich« durch falsche Sollwerte:

$$MC = \eta_{\mathrm{Div}} \tag{1.22}$$

$$\zeta_{\mathrm{Phan}} = \zeta_{\mathrm{Chk}} \cdot (1 - \eta_{\mathrm{Div}}) \tag{1.23}$$

MC	Fehlfunktionsüberdeckung (**m**alfunction **c**overage), Anteil nachweisbare Fehlfunktionen.
η_{Div}	**Div**ersitätsrate, Anteil der MFs ohne gemeinsame Ursache.
ζ_{Phan}	Phantom-MF-Rate.
ζ_{Chk}	MF-Rate **Ch**eck**e**r.

Vergleichsfehler



Für Bitwerte sind Vergleichsfehler unwahrscheinlich, weil

- die Toleranzbereiche viel größer als die Standardabweichungen unverfälscher Signale sind (praktische Ausschluss von Phantom-MF) und
- jede Bitinvertierung als Vergleichfehler zählt (Ausschluss von Fehlklassifikationen per Definition).

Analoge, Abtast- und numerische Werte

Die Verfälschungen analoger Werte durch
- Rauschen, Drift,
- induktives- und kapazitives Übersprechen

akkumulieren sich bei der Übertragung und Verarbeitung und sind danach kaum noch vom Nutzsignal zu trennen. Bei der Digitalisierung kommen hinzu:
- Quantisierungsfehler (Erwartungswert: $0{,}5 \cdot LSB$)
- Linearitäts- und andere Fehler.

Digitale numerische Berechnungen haben Rundungsfehler:
- Addition und Subtraktion: Akkumulation der absuluten Varianzen der Rundungsfehler
- Multiplikation und Division: Akkumulation der relativen Varianzen der Rundungsfehler.

(siehe Abschn. 3.1.2 *Lineare Transformation*).

LSB least significant bit (kleinster Quantisierungsschritt).

Fenstervergleich

Bereichszugehörigkeit eines zu kontrollierenden Wertes X:

$$XP = (x_{\min} \leq X \leq x_{\max})$$

überlagert von Verarbeitungsverfälschungen X_{F} durch Störungen, Quantisierungfehler und Rundungsfehler:

$$X_{\mathrm{M}} = X + X_{\mathrm{F}} \tag{3.22}$$

$$MP = (x_{\min} \leq X_{\mathrm{M}} \leq x_{\max})$$

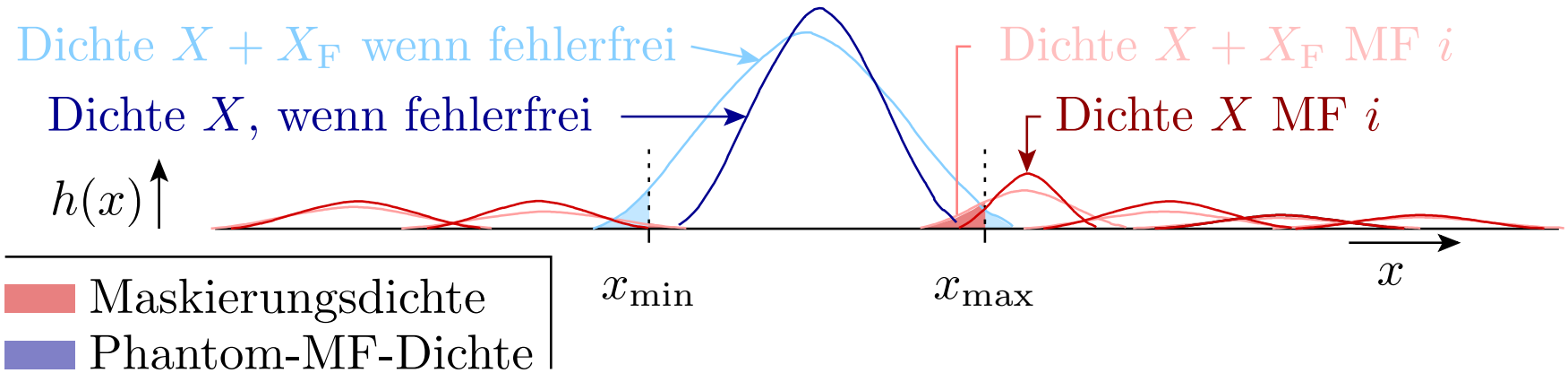
Der zu kontrollierende Wert X und die Verfälschung X_{F} sind in der Regel unabhängige Zufallsvariablen, für deren Erwartungswerte und Varianzen gilt:

$$\mathbb{E}\left[X_{\mathrm{M}}\right] = \mathbb{E}\left[X\right] + \mathbb{E}\left[X_{\mathrm{F}}\right] \tag{3.23}$$

$$\mathrm{Var}\left[X_{\mathrm{M}}\right] = \mathrm{Var}\left[X\right] + \mathrm{Var}\left[X_{\mathrm{F}}\right] \tag{3.24}$$

X	zu kontrollierender Wert (Zufallsvariable).
X_{F}	Verarbeitungsverfälschungen durch Störungen, Quantisierungs- und Rundungsfehler.
XP	Soll-Kontrollergebnis, ob tatsächlicher Wert zulässig.
MP	Ist-Kontrollergebnis, ob gemessene Wert zulässig.

Maskierung und Phantom-MF

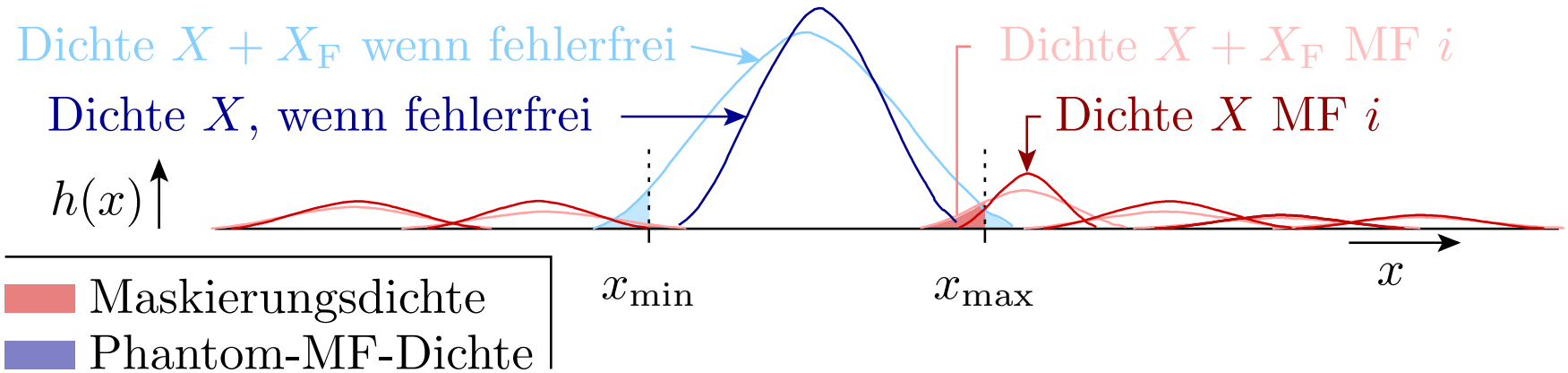


Die Verteilung zu kontrollierenden Werte X ist eine Mischverteilung aus korrekten Werten und Werten, die durch unterschiedliche fehlerhafte Verarbeitung entstehen.

Die gemischten Grundgesamtheiten für korrekte Verarbeitung und gleiche fehlerhafte Verarbeitungen tendieren zu Normalverteilungen.

Die Verarbeitungsverfälschungen durch Rauschen, Rundung und Quantisierung erhöhen die Standardabweichung und damit die Breite der einzelnen Dichtekurven der Modi.

Maskierungswahrscheinlichkeit

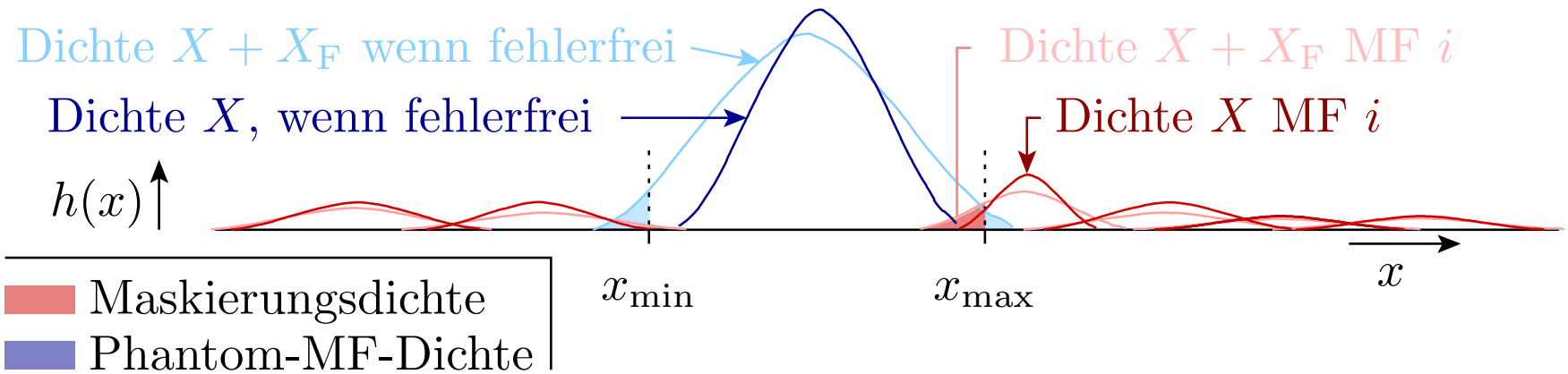


Die Erkennungswahrscheinlichkeit als bedingte Wahrscheinlichkeit, dass der gemessene Werte außerhalb des zulässigen Bereich liegen, wenn der tatsächliche Wert außerhalb liegt:

$$p_D = \mathbb{P}\left(MP|\overline{XP}\right) = \frac{\mathbb{P}((X_M<x_{min})\wedge(X<x_{min}))}{\mathbb{P}(X<x_{min})} + \frac{\mathbb{P}((X_M>x_{max})\wedge(X>x_{max}))}{\mathbb{P}(X>x_{max})}$$

Probleme für die praktische Nutzung der Beziehung bereitet die Abschätzung der Werteverteilung für fehlerhafte Werte, die ja offenbar nicht gegen eine Normalverteilung oder eine andere Verteilung, die sich mit wenigen Parametern beschreiben lässt, strebt.

Phantom-MF-Rate



Die Phantom-MF-Rate als bedingte Wahrscheinlichkeit, dass korrekte Messwert außerhalb des zulässigen Bereichs liegen:

$$\zeta_{\text{Phan}} = \mathbb{P}\left(\overline{MP}|XP\right) = \frac{\mathbb{P}(X_{\text{M}}<x_{\min})+\mathbb{P}(X_{\text{M}}>x_{\max})}{\mathbb{P}(x_{\min}\leq X\leq x_{\max})}$$

ist einfacher abzuschätzen. In eine benutzbaren System sind fast alle zu kontrollierenden Werte korrekt und der Bereich $x_{\min}, x_{\max}$ ist so gewählt, dass korrekte Werte ohne die normalen Verarbeitungsverfälschungen durch Rundung etc im zulässigen Bereich liegen:

$$\mathbb{P}\left(x_{\min} \leq X \leq x_{\max}\right) = 1$$

α	2,27%	0,13%	0	2%	1%	0,5%	0,2%	0,1%
$\Phi^{-1}(1-\alpha)$	2	3	4	2,05	2,33	2,57	2,88	3,10

Die Summe aus korrekten Werten und Verfälschungen ist typisch nomalverteilt mit Erwartungswert μ_{CM} und Standardabweichung σ_{CM}. In Anlehnung an Gl. 3.41 und 3.42 beträgt die Phantom-MF-Rate:

$$\zeta_{\mathrm{Phan}} = \Phi\left(\frac{x_{\min}-\mu_{\mathrm{CM}}}{\sigma_{\mathrm{CM}}}\right) + 1 - \Phi\left(\frac{x_{\max}-\mu_{\mathrm{CM}}}{\sigma_{\mathrm{CM}}}\right) \tag{10}$$

Nach Gl. 3.47 ergeben sich aus einer zulässigen Phantom-MF-Rate ζ_{Phan} die symmetrischen Bereichsgrenzen:

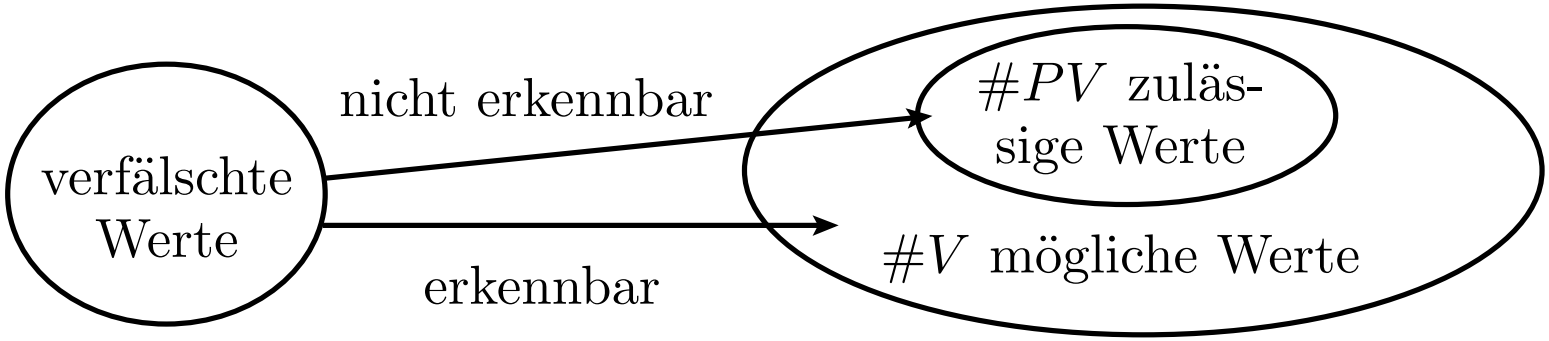
$$\mathrm{sr}\,(x) = \mu_{\mathrm{CM}} \mp \sigma_{\mathrm{CM}} \cdot \Phi^{-1}\left(1 - \frac{\zeta_{\mathrm{Phan}}}{2}\right) \tag{11}$$

Der Ausschluss von Phantom-MF verlangt mindestens einen $4\ldots6 \cdot \sigma_{\mathrm{CM}}$ Intervallradius.

ζ_{Phan}	Phantom-MF-Rate.
$\Phi(z)$	Verteilungsfunktion der standardisierten Normalverteilung.
σ_{CM}	kleinster zulässiger Wert des Fenstervergleichs.
σ_{CM}	größter zulässiger Wert des Fenstervergleichs.
μ_{CM}	Erwartungswert korrekter Werte mit normalen Verarbeitungsverfälschungen.
σ_{CM}	Standardabweichung korrekter Werte mit normalen Verarbeitungsverfälschungen.
$\Phi^{-1}(..)$	Inverse Funktion zur Verteilungsfunktion der standardisierten Normalverteilung.

Informationsredundanz

Wiederholung Informationsredundanz



Eine Fehlfunktion bildet eine zulässige Ausgabe entweder auf eine zulässige oder eine unzulässige Ausgabe ab. Wenn alle Verfälschungsmöglichkeiten gleichhäufig auftreten und alle unzulässigen Werte erkannt werden:

$$p_{\mathrm{D}} = 1 - \frac{\#PV}{\#V} \tag{1.19}$$

$$\zeta_{\mathrm{Phan}} = 0 \tag{1.20}$$

p_{D}	Nachweiswahrscheinlichkeit (detection probability).
$\#PV$	Anzahl der zulässigen Werte (number of **p**ermitted **v**alues).
$\#V$	Anzahl der möglichen Werte (number of possible **v**alues).
ζ_{Phan}	Phantom-MF-Rate.

Beispiel Rechtschreibtest

Wort im Wörterbuch enthalten?

- Erkennung, wenn nicht falsches Wort nicht im Wörterbuch wie »Maus« statt »Haus«. Viel schlechter als nach Gl. 1.19:

$$p_{\mathrm{D}} = 60\% \ldots 90\% \ll 1 - \frac{\#PV}{\#V}$$

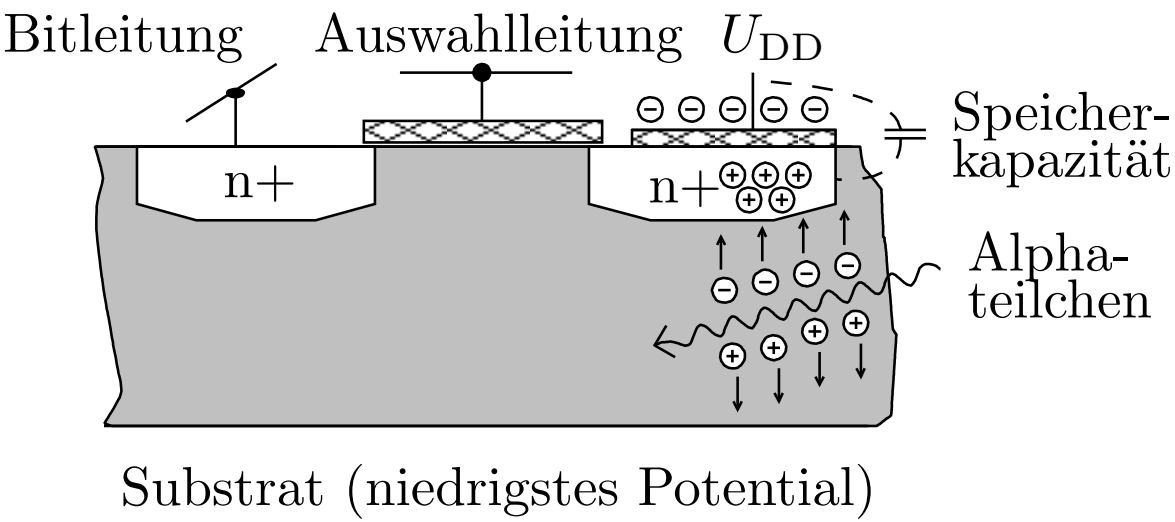
- Phantom-MF, zulässiges Wort nicht im Wörterbuch, typisch:

$$\zeta_{\mathrm{Phan}} = \frac{1 \ldots 10\,\mathrm{PM}}{1000\,\mathrm{DS}} \gg 0$$

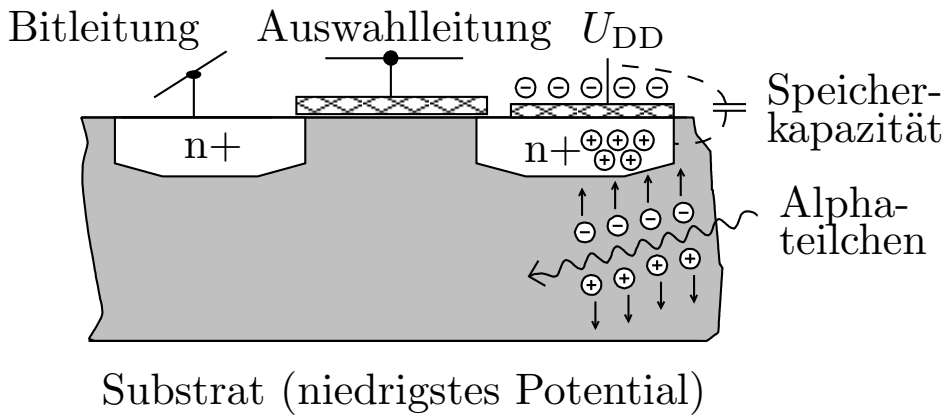
Beim Rechtschreibtest ist der Anteil der zulässigen Worte an den darstellbaren Zeichenfolgen fast null, aber bei Schreibfehlern entstehen überdurchschnittlich oft zulässige Worte und die Wörterbücher enthalten in er Regel auch bei weitem nicht alle zulässigen Zusammensetzungen und Abänderungen von Worten.

p_{D}	Nachweiswahrscheinlichkeit (detection probability).
ζ_{Phan}	Phantom-MF-Rate.
PM	Phantom-Fehlfunktion (**p**hantom **m**alfunction).
DS	gelieferter Service (delivered service).

Paritätstest für DRAMs und Speicherriegel



- Informationsspeicherung in winzigen Kapazitäten.
- Häufigste Ursache für Datenverfälschungen: Alphastrahlung.
- Deren Quellen radioaktiver Zerfall von Uran und Thorium, enthalten als Spurenelemente im Gehäuse und im Aluminium der Leiterbahnen oder Kernprozesse im Silizium durch Höhenstrahlung.
- Mittlerer Ereignisabstand: vieler Stunden oder Tage.
- Paritätstest erkennt alle ungeradzahligen und im Mitte 50% der Verfälschungen.



- Energie eines Alphateilchen: $5\,\mathrm{MeV}$. Energieverlust bei der Generierung eines Elektronen-Loch-Paares $\approx 3{,}6\,\mathrm{eV} \Rightarrow$ Generierung von $\approx 10^6$ Ladungsträgerpaaren. Reichweite $\approx 89\,\mu\mathrm{m}$. gespeicherte Ladung $\approx 10^5$ Ladungsträger. Datenverlust einer oder mehrerer benachbarter Zellen möglich.
- Mittlerer Zeitabstand zwischen zwei Datenverfälschungen Stunden. Gleichzeitige Verfälschung durch zwei Alphateilchen unwahrscheinlich.
- Geometrische Trennung der Zellen eines Datenworts (getrennte Schaltkreise oder Speichermatrizen) $\Rightarrow$ Je Zerfall Einzelbitverfälschung je gelesenes Datenwort.

Auch bei der Übertragung sind Verfälschungen selten. Datenobjekte so verschränken, das auch Fehler-Bursts auf Einzelbitfehler je Datenobjekt abgebildet werden.

Einzelbitfehler bei Übertragung und Speicherung

Wenn die Daten so auf Datenobjekte aufgeteilt werden, dass jedes Verfälschungsereignis max. ein Bit je Datenobjekt verfälscht, genügt ein fehlererkennender Code für Einzelbitfehler, also ein Paritätstest.

Für seltene unabhängige Bitverfälschungen ist die Anzahl der verfälschten Bits je Datenobjekt X poisson-verteilt:

$$\mathbb{P}\left(X=k\right)=e^{-\lambda}\cdot\frac{\lambda^k}{k!} \tag{3.32}$$

Paritätstest erkennt 50% und zwar alle ungeradzahligen Bitverf.:

$$p_{\mathrm{D}}=\frac{\mathbb{P}\left[X=1\right]+\mathbb{P}\left[X=3\right]+\ldots}{\mathbb{P}\left[X\geq 1\right]}=\frac{\mathrm{e}^{-\lambda}\cdot\left(\lambda+\frac{\lambda^3}{6}+\ldots\right)}{1-\mathrm{e}^{-\lambda}}$$

Für $\lambda\ll 1$ und Taylor-Reihe $\mathrm{e}^{-\lambda}=1-\lambda+\frac{\lambda^2}{2}-\ldots$:

$$p_{\mathrm{D}}=\frac{(1-\lambda)\cdot\lambda}{1-(1-\lambda)}=1-\lambda\gg 50\% \tag{12}$$

λ	zu erwartende Bitfehleranzahl je Datenwort.
p_{D}	Nachweiswahrscheinlichkeit (detection probability).

Wertebereichskontrolle

Die Menge der zulässigen Werte eines Datenobjekts ist meist viel kleiner als die Menge der darstellbaren Werte. Überwachung zusammenhängender Wertebereiche:

```
u32 a;                  // Alter Angestellter WB:18...67
                        // WB(u32):0...0xMFMFMFMF
if (a < 18 || a > 67) <Fehlerbehandlung>;
```

Wenn bei Verfälschung alle darstellbaren Werte gleichwahrscheinlich wären, Erkennungswahrscheinlichkeit:

$$p_{\mathrm{D}} = 1 - \frac{\overbrace{67 - 18 + 1}^{\text{Anz. zul. Werte}}}{\underbrace{2^{32}}_{\text{Anz. mögl. Werte}}} = 1 - 10^{-8}$$

Würde bedeuten, dass praktisch jede Verfälschung erkannt wird, aber:

- kleine Werte stehen viel öfter als große im Speicher,
- ...

- kleine Werte stehen viel öfter als große im Speicher„
- Verwechslung mit Alter andere Person, immer zulässig,
- Verwechslung mit der Hausnummer, oft zulässig, ...

Erkennungswahrscheinlichkeit viel schlechter als nach Gl. 1.19:

$$p_{\mathrm{D}} \ll 1 - \frac{\#PV}{\#V}$$

In einem Programm sind sehr viele WB-Kontrollen. Verlängerung der Codeworte um r Bit. Rückgewinnung mit LFSR als Schaltwerk, dss Schritt für Schritt dir inverse Operation zur Polynom-Multiplikation ausführt und Kontrolle des Endwertes des r-Bit LFSR: einprogrammierbar, auch für WB-Überläufe, auch vom Compiler automatisch einfügbar, auch zum Teil als statische Kontrollen beim Compilieren.

Die Leistungsfähigkeit von Wertebereichskontrollen liegt in der Vielzahl der Kontrollmöglichkeiten.

Erhöhung der Erkennungswahrscheinlichkeit von WB-Kontrollen:

- WB-Verschiebung durch Addition einer Konstanten,
- pseudozufällige Codierung der zulässigen Werte, ... (vergl. fehlererkennende Codes),
- zusätzliche Typüberwachung, ...

Fehlererkennende Codes

Prinzip der fehlererkennenden Codes

Umkehrbar eindeutige pseudo-zufällige Zuordnung der zulässigen Werte zu möglichen Werte.

Wenn die Verfälschungsursache »den Code-Schlüssel« nicht kennt, entstehen durch Fehlfunktionen weder bevorzugt zulässige noch unzulässige Werte. Damit gilt

$$p_{\mathrm{D}} = 1 - \frac{\#PV}{\#V} \tag{1.19}$$

Beispiel: Multiplikation mit einer ganzzahligen großen Konstanten K

$$s = K \cdot x$$

Erkennungswahrscheinlichkeit

$$p_{\mathrm{D}} = 1 - \frac{\#x}{\#x \cdot K} = 1 - \frac{1}{K} \tag{13}$$

Vor der Verarbeitung und Speicherung Multiplikation der zu überwachenden Werte mit K. Verfälschungen werden »an Divisionsrest s/K ungleich null« erkannt.

K große ganzzahlige Konstante, bevorzugt eine Primzahl.

Polynom-Multiplikation und -division

Codierung durch die Multiplikation mit einer Konstanten, allerdings nicht arithmetisch, sondern modulo-2. In Hard- oder Software einfacher als arithmetische Multiplikation:

Codierung

$$
\begin{array}{rr}
 & 10010101101 \\
\odot & 10011 \\
\hline
\oplus & 10010101101 \\
\oplus & 10010101101 \\
\oplus & 00000000000 \\
\oplus & 00000000000 \\
\oplus & 10010101101 \\
\hline
 & 1000111\,0\,0100111
\end{array}
$$

Decodierung

$$
\begin{array}{rl}
 & 1000111\,0\,0100111 : 10011 = 10010101101 \\
\oplus & 10011 \\
\hline
 & 10110 \\
\oplus & 10011 \\
\hline
 & 10101 \\
 & 10011 \\
\hline
 & 11000 \\
\oplus & 10011 \\
\hline
 & 10111 \\
\oplus & 10011 \\
\hline
 & 10011 \\
\oplus & 10011 \\
\hline
\text{Rest:} & 0000 \qquad \text{(unverfälscht)}
\end{array}
$$

In der Literatur finden Sie noch eine andere Beschreibung für die gerade vorgeführte abgewandelte Form der Multiplikation und Division von Bitvektoren. Mathematisch werden die zu multiplizierenden Faktoren als Polynome dargestellt:

- $10011 \Rightarrow 1 \cdot x^4 \oplus 0 \cdot x^3 \oplus 0 \cdot x^2 \oplus 1 \cdot x^1 \oplus 1 \cdot x^0 = x^4 \oplus x \oplus 1$
- $10010101101 \Rightarrow x^{10} \oplus x^7 \oplus x^5 \oplus x^3 \oplus x^2 \oplus 1$

Eine Multiplikation mit x beschreibt eine Verschiebung um eine Bitstelle. Die Multiplikation mit null oder eins ist eine UND-Verknüpfung und $\oplus$ die modulo-2-Addition (EXOR). Das Produkt beider Polynome

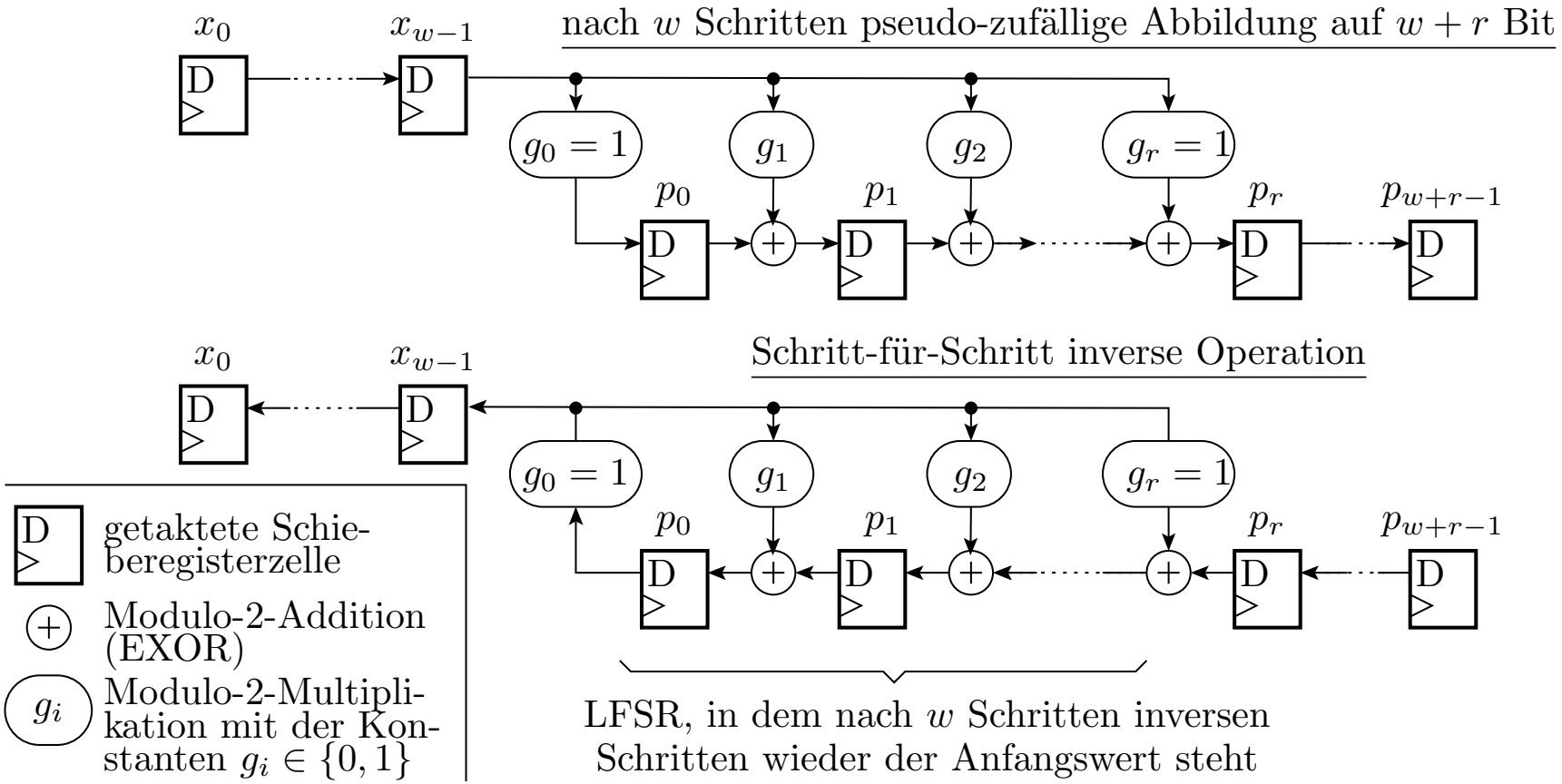
$$\left(x^{10} \oplus x^7 \oplus x^5 \oplus x^3 \oplus x^2 \oplus 1\right) \cdot \left(x^4 \oplus x \oplus 1\right) \quad = \\ x^{14} \oplus x^{10} \oplus x^9 \oplus x^8 \oplus x^5 \oplus x^2 \oplus x \oplus 1$$

repräsentiert denselben Bitvektor, der für die Multiplikation der Folgen auf der Folie zuvor berechnet wurde. Die Polynomdivision:

$$\left(x^{14} \oplus x^{10} \oplus x^9 \oplus x^8 \oplus x^5 \oplus x^2 \oplus x \oplus 1\right) : \left(x^4 \oplus x \oplus 1\right) \quad = \\ x^{10} \oplus x^7 \oplus x^5 \oplus x^3 \oplus x^2 \oplus 1$$

liefert ohne Rest das Polynom der Originalfolge.

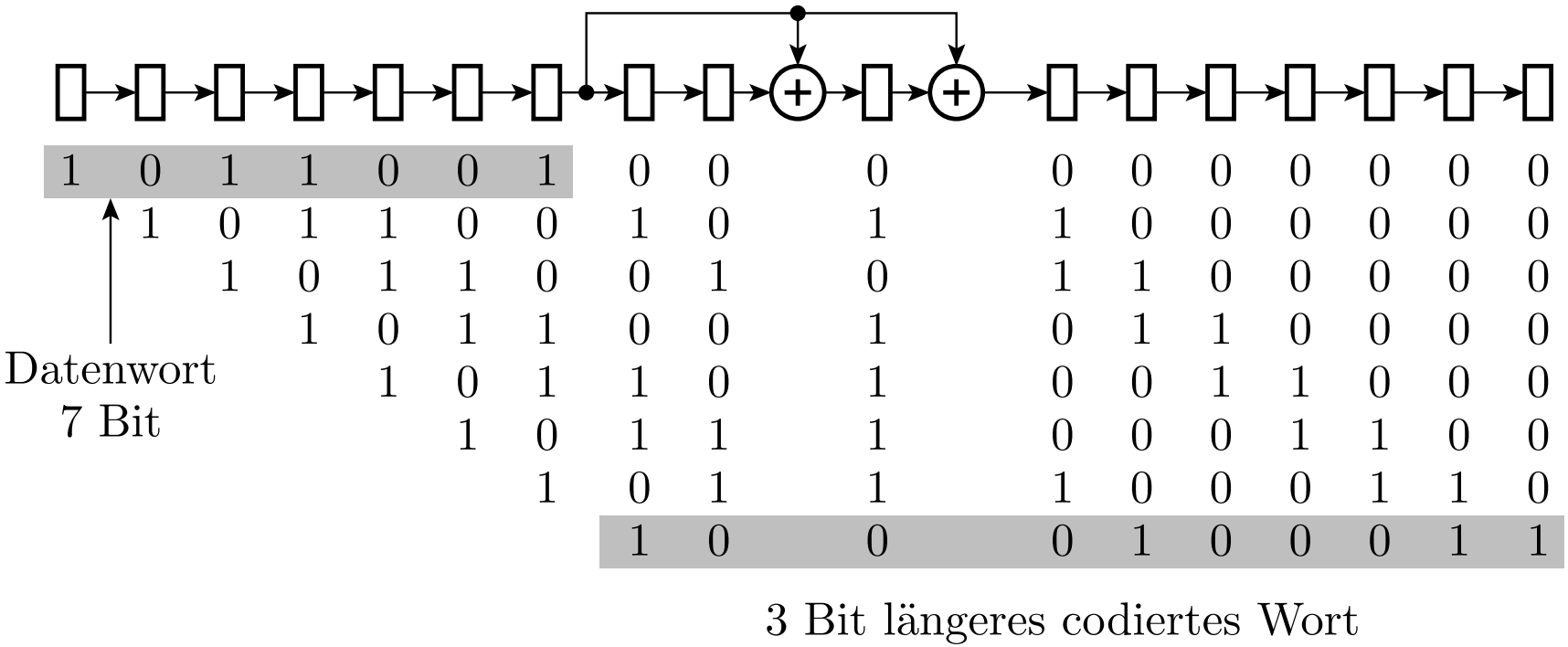
Polynommultiplikation und -Division mit SR



$$p_{\mathrm{D}} \geq 1 - 2^{-r} \tag{1.21}$$

SR	Schieberegister (**s**hift **r**egister).
LFSR	linear rückgekoppeltes Schieberegister (**l**inear **f**eeback **s**hift **r**egister).

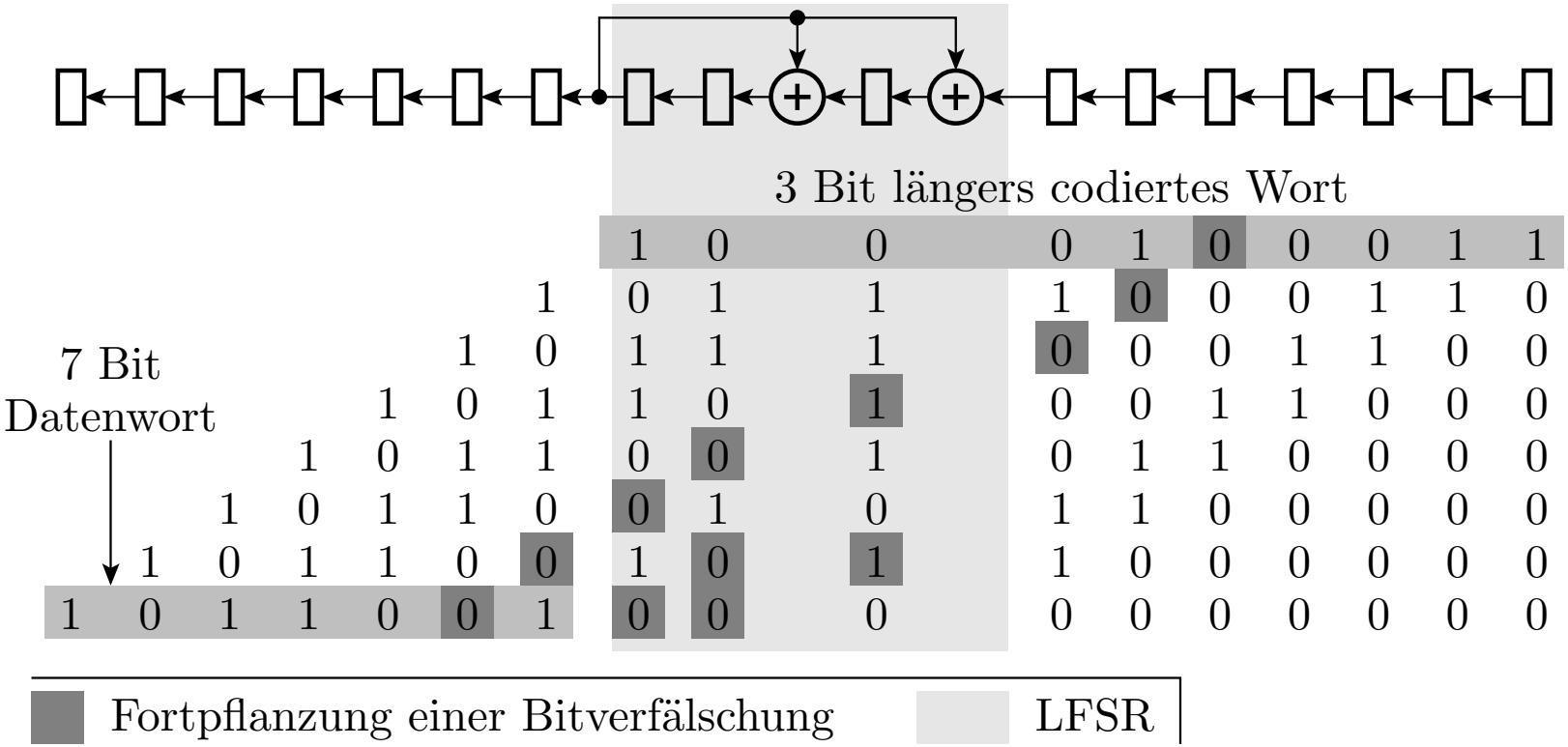
Codierung durch SR-Polynommultiplikation



- Das Ergebnis ist $r = 3$ Bit länger und pseudo-zufällig umcodiert.
- Anzahl der zulässigen Codeworte bleibt 2^7.
- Anzahl der möglichen Codeworte vergrößert sich auf 2^{10}.

p_{D}	Nachweiswahrscheinlichkeit (detection probability).
r	Anzahl der **r**edundanten Bits.

Rückgewinnung durch Polynomdivision



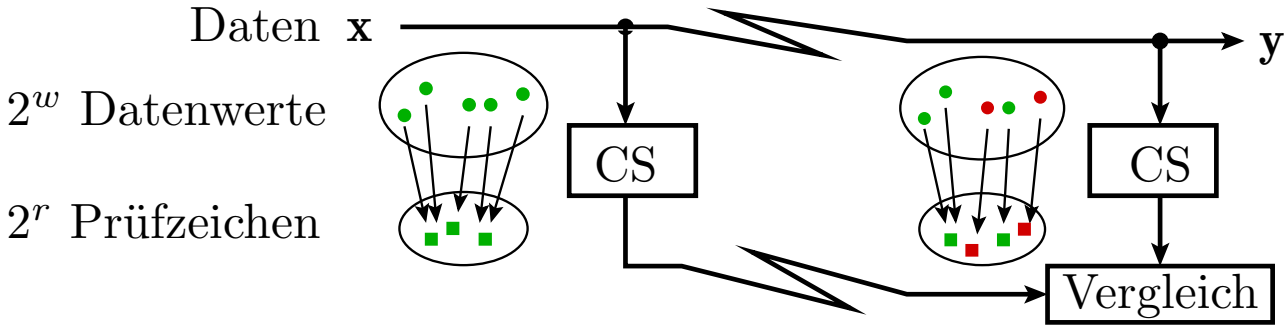
							1	0	0	0	1	0	0	0	1	1		
						1	0	1	1	1	0	0	0	1	1	0		
					1	0	1	1	1	0	0	0	1	1	0	0		
				1	0	1	1	0	1	0	0	1	1	0	0	0		
			1	0	1	1	0	0	1	0	1	1	0	0	0	0		
		1	0	1	1	0	0	1	0	1	1	0	0	0	0	0		
	1	0	1	1	0	0	1	0	1	1	0	0	0	0	0	0		
1	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0		

Eine Bitverfälschung verursacht mit der Wahrscheinlichkeit $p_{\mathrm{D}} = 1 - 2^{-3}$ einen von null abweichenden Endwert im LFSR.

LFSR	linear rückgekoppeltes Schieberegister (**l**inear **f**eeback **s**hift **r**egister).
p_{D}	Nachweiswahrscheinlichkeit (detection probability).

Prüfzeichen

Prüfzeichen



- Jedem w-Bit-Datenwort wird pseudo-zufällig genau eines der r-Bit-Prüfzeichen zugeordnet $(w \gg r)$.
- Nach der Übertragung oder Speicherung wird das Prüfzeichen ein zweites mal gebildet.
- Wenn weder die Daten noch das Prüfzeichen verfälscht sind, stimmen beide Prüfzeichen überein.

Für pseudo-zufällig gebildete Prüfzeichen gilt:

- Anzahl der zulässigen Prüfzeichen-Werte-Paare 2^w,
- Anzahl darstellbarer Paare 2^{w+r}:

$$p_{\mathrm{D}} \geq 1 - 2^{-r} \tag{1.21}$$

r	Anzahl der **r**edundanten Bits.
w	Datenbitanzahl (number of data bits).
CS	Prüfzeichen (**c**heck **s**ymbol).

Prüfsummen

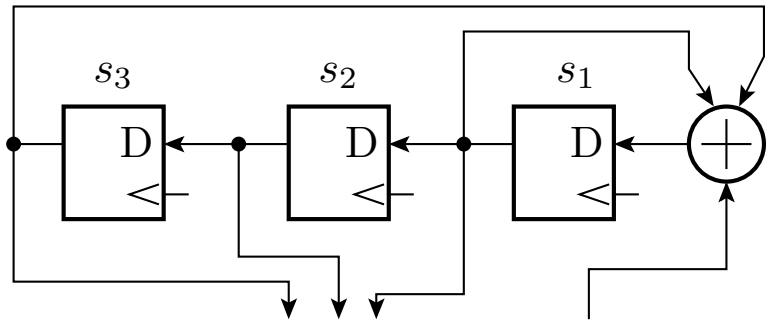
Prüfzeichbildung durch byteweise Addition oder EXOR-Verknüpfung.

einfache Genauigkeit	doppelte Genauigkeit	bitweises EXOR
1011 11	1011 11	1011
0010 2	0010 2	0110
1101 13	1101 13	1101
0100 4	0100 4	1100
(1) 1110 14 (+16)	0001 1110 30	1100

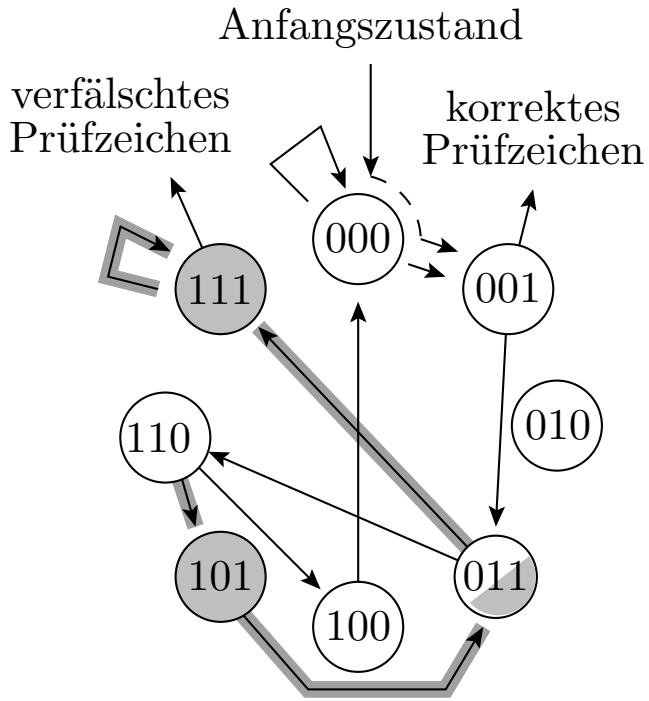
Bei »einfacher Genauigkeit« und »bitweisem EXOR« erscheint die Annahme »pseudo-zufällige Abbildung« gerechtfertigt*: $p_{\mathrm{D}} \approx 1 - 2^{-4}$.
Bei »doppelter Genauigkeit« bilden sich Verfälschungen vorzugsweise auf die niederwertigen Bits ab. Maskierungswahrscheinlichkeit: $2^{-4} > 1 - p_{\mathrm{D}} \gg 2^{-8}$.

p_{D} Nachweiswahrscheinlichkeit (detection probability).
$*$ Kein Nachweis für vertauschte Summationsreihenfolge.

Prüfzeichenbildung mit LFSR



Initialwert	0	0	0	1
Schritt 1:	0	0	1	0
Schritt 2:	0	1	1	1
Schritt 3:	1	1	0	1
Schritt 4:	1	0	0	1
Schritt 5:	0	0	0	0
Schritt 6:	0	0	0	1
Schritt 7:	0	0	1	

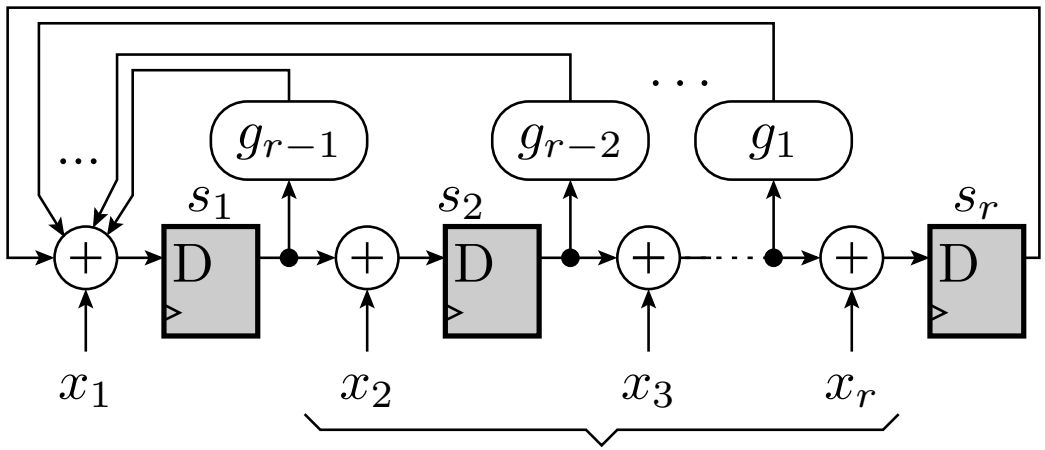


Das Prüfzeichen wird wie bei der CRC-Decodierung mit einem linear rückgekoppelten Schieberegister (LSFR) gebildet. Im Beispiel hat das LSFR im Gegensatz zur Polynomdivision (siehe Folie 4.94) eine zentrale Rückführung. Abbildung auch pseudo-zufällig.

LFSR linear rückgekoppeltes Schieberegister (**l**inear **f**eeback **s**hift **r**egister).

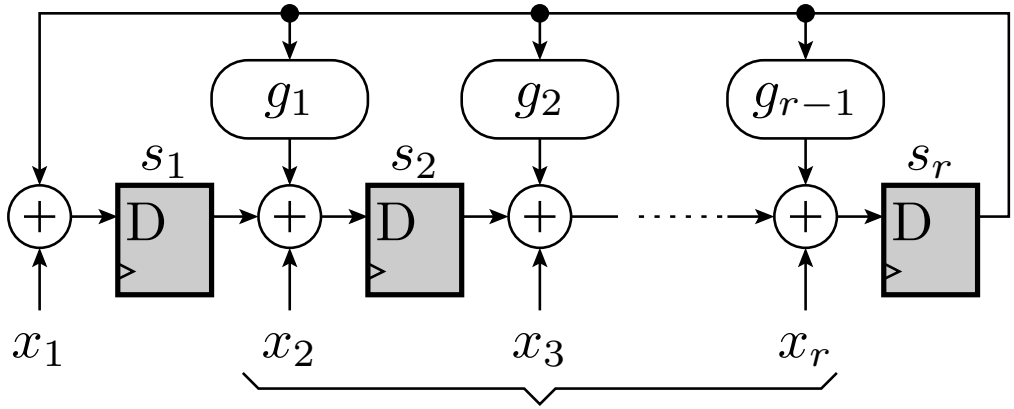
Allgemeine Struktur von LFSR

Für die Bildung auf Prüfzeichen ist es nur wichtig, dass die Abbildung pseudo-zufällig hinsichtlich der zu erwartenden Verfälschungen erfolgt. Diese Eigenschaft hat auch ein rückgekoppeltes Schieberegister, bei dem in jedem Zeitschritt mehrere Eingabebits modulo-2 zu den Registerzuständen addiert werden.



Erweiterungsmöglichkeit auf mehrere Eingänge

Die Rückführung darf dabei auch wie bei der Polynom-Division dezentral sein.



Erweiterungsmöglichkeit auf mehrere Eingänge

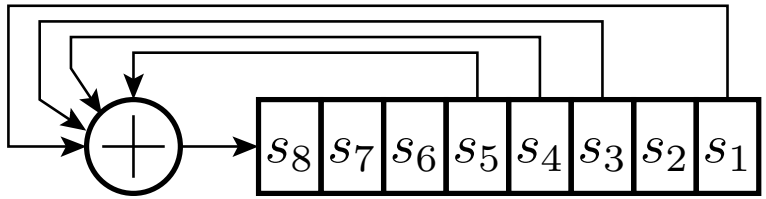
Die Koeffizienten g_i der Rückführung, bei der Polynom-Division das Divisor-Polynom, bestimmen die autonome Zyklusstruktur*. Die autonome Zyklusstruktur ist bei zentraler und dezentraler Rückführung mit denselben Rückführkoeffizienten gleich.

Bevorzugt werden lange Zyklen, insbesondere sog. primitive Polynome, bei denen alle Zustände außer »alles null« einen $2^r - 1$ langen Maximalzyklus bilden.

s_i	Speicherzelle (Bitwert) i.
g_i	Rückführkoeffizient i.
r	Bitanzahl des linear rückgekoppelten Schiegeregisters.
$*$	Zyklusstruktur ohne Eingaben.

Autonome Zykluslänge und -struktur von LFSR

Autonom bedeutet Zyklusstrukur für Eingabewerte »alle 0« oder für LFSR ohne Eingänge. Beispiel 8-Bit autonomes LFSR:



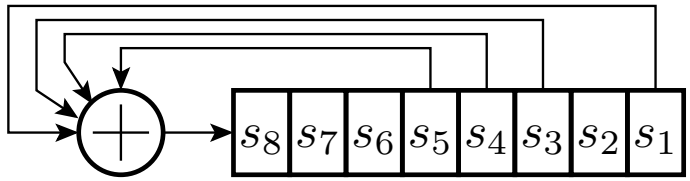
0	0	1	1	0	0	0	1
0	0	0	1	1	0	0	0
0	0	0	0	1	1	0	0
0	0	0	0	0	1	1	0
1	0	0	0	0	0	1	1
1	1	0	0	0	0	0	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Übergangsfunktion:

$$\begin{aligned} s_7 &= s_4 \oplus s_3 \oplus s_2 \oplus s_0 \\ s_i &= s_{i+1} \text{ für } i \in \{0, 1, 2, \ldots 6\} \end{aligned}$$

LFSR linear rückgekoppeltes Schieberegister (**l**inear **f**eeback **s**hift **r**egister).

Bestimmung der Zykluslänge durch Simulation

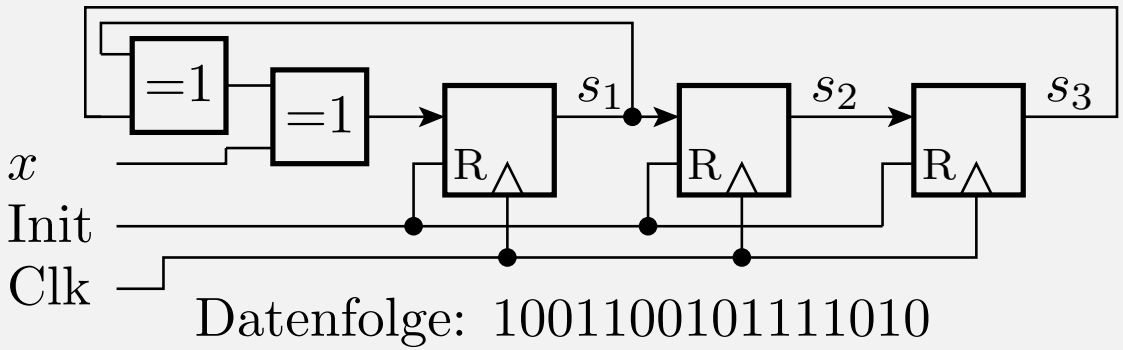



```
...
#define SA 0x31
uint8_t s = SA;          // Startwert setzen
while(1){
  s = (s>>1) ^ (s<<7) ^ ((s<<5)&0x80)
      ^ ((s<<4)&0x80) ^ ((s<<3)&0x80);
  <Ausgabe von s>
  if (s==SA) break;  // bis wieder Anfangswert
  ...                // weiter mit nicht enthal-
}                    // tenem Zustand als Startw.
```

- 0x00 geht in sich selbst über.
- Alle anderes 255 Zustände gehen zyklisch ineinander über.
- max. Zykluslänge $2^r - 1$: primitive Rückkopplung.

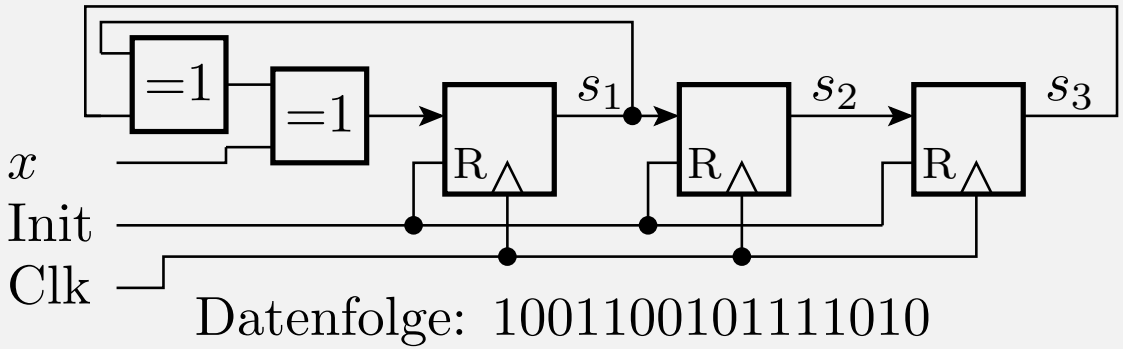
Beispiel 4.3: Prüfzeichen mit LFSR

Gegeben ist folgendes linear rückgekoppelte Schieberegister:



a) Welches Prüfzeichen $\mathrm{s} = s_1 s_2 s_3$ hat die Datenfolge bei Abbildung beginnend mit dem linken Bit und Startwert 000.

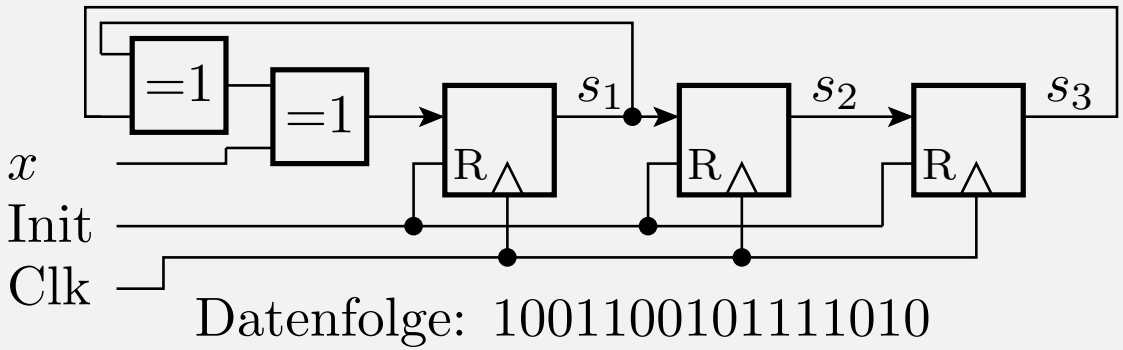
b) Wie hoch ist Fehlererkennungswahrscheinlichkeit?



a) Welches Prüfzeichen s $= s_1 s_2 s_3$ hat die Datenfolge bei Abbildung beginnend mit dem linken Bit und Startwert 000.

	x	s_1	s_2	s_3
0	1	0	0	0
1	0	1	0	0
2	0	1	1	0
3	1	1	1	1
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	1	1	0	1

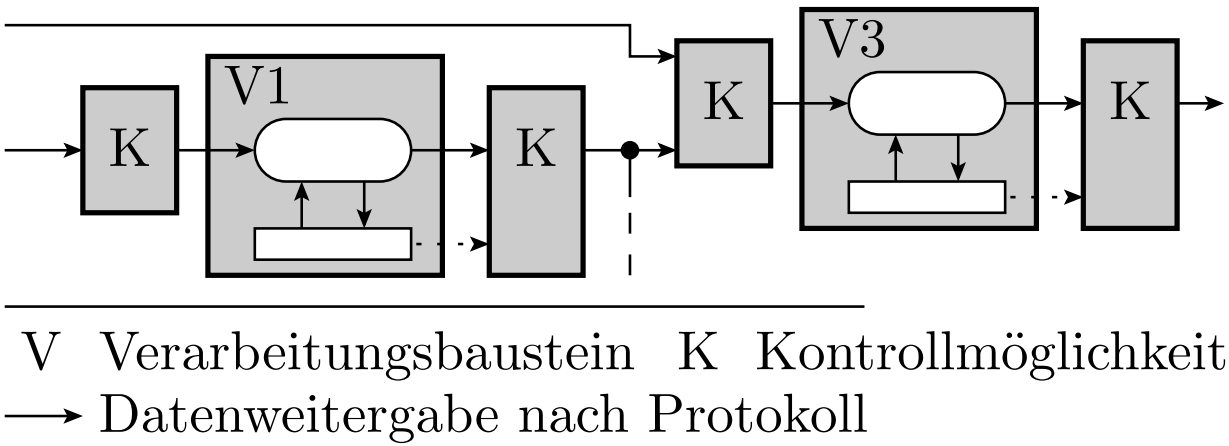
	x	s_1	s_2	s_3
8	0	1	1	0
9	1	1	1	1
10	1	1	1	1
11	1	1	1	1
12	1	1	1	1
13	0	1	1	1
14	1	0	1	1
15	0	0	0	1
PKZ:		1	0	0



b) Wie hoch ist Fehlererkennungswahrscheinlichkeit?

$$p_{\mathrm{D}} \approx 1 - 2^{-3} = 87{,}5\%$$

Protokolle



IT-Systeme bestehen in der Regel aus einer Vielzahl komplexer Bausteine, die über wohldefinierte Schnittstellen und Protokolle mit \AbbrCheckSymeinander kommunizieren:

- Software-Schnittstellen, Busprotokoll,
- Netzwerkprotokolle, Signaldefinitionen, Internet-Protokoll, ...

Für Prozessoren, APIs, ... hunderte Seiten Dokumentation

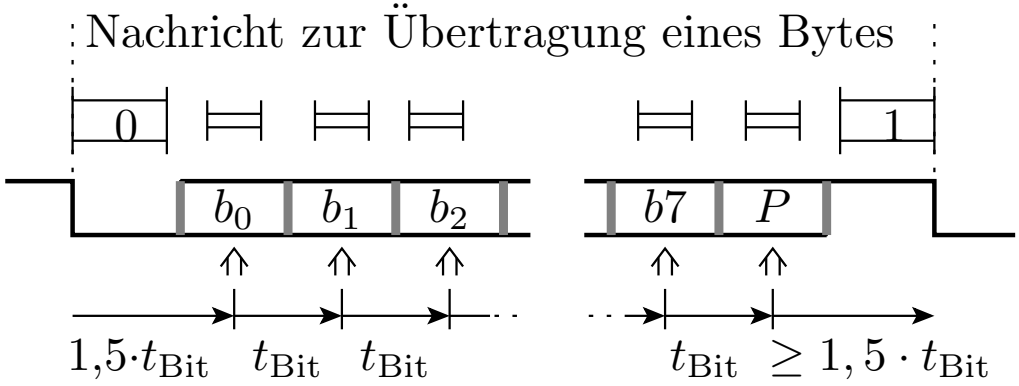
- welche Daten wie, mit welchen Befehlen in welcher Reihenfolge bei der Anforderung welcher SL zu übergeben sind,
- welche Vorbedingungen die Daten zu erfüllen haben,
- welche überprüfbaren Bedingungen korrekte Ergebnisse erfüllen,
- Antwortzeitschranken , ...

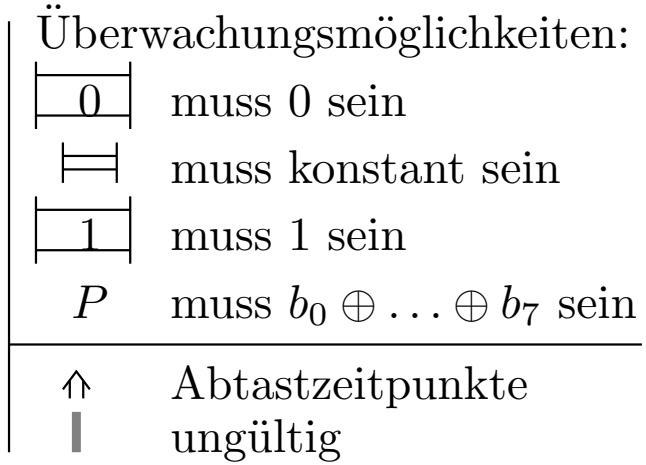
$\Rightarrow$ unzählige Möglichkeiten für Formatkontrollen.

Beispiel UART

Das UART-Protokoll ist der Klassiker zur bytweisen Datenübertragung zwischen Mikrorechner über eine Leitung je Übertragungsrichtung:

- Baud-Rate (Bitzeiten pro s): 4800, 9600, ...
- 1 Startbit mit Wert 0,
- 7 / 8 /9 Datenbits,
- kein /gerades / ungerades Paritätsbit und
- 1 / 2 Stoppbit(s) mit Wert 1.

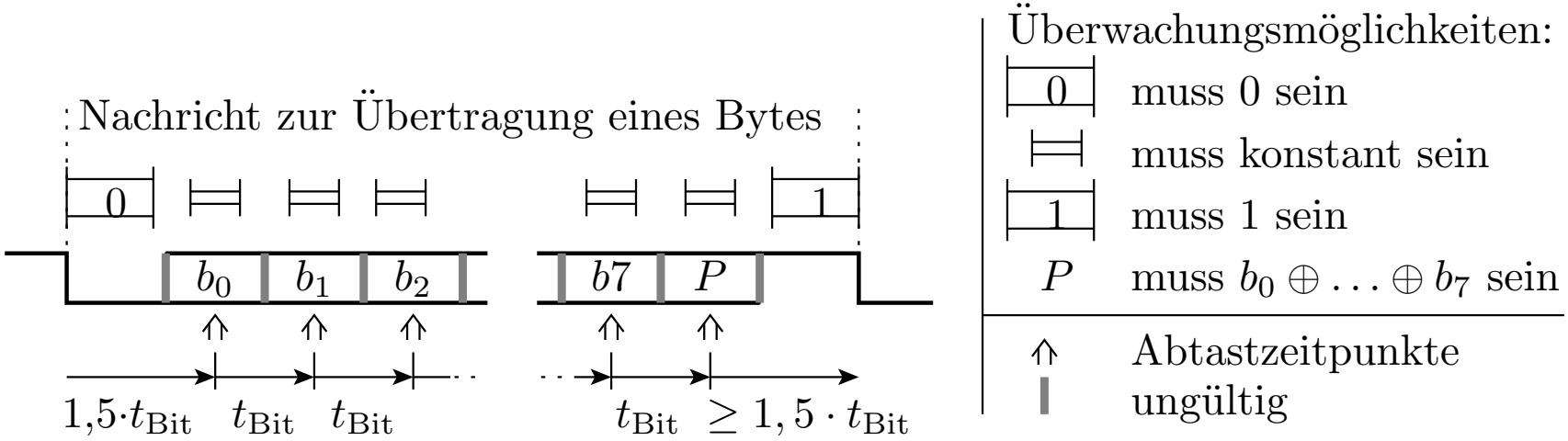




b_i	Datenbit i.
P	Partitätsbit.
t_{Bit}	Bitzeit.
UART	y.

mit universal asynchronous receiver transmitter Universal

Kontrollierbare Formateigenschaften

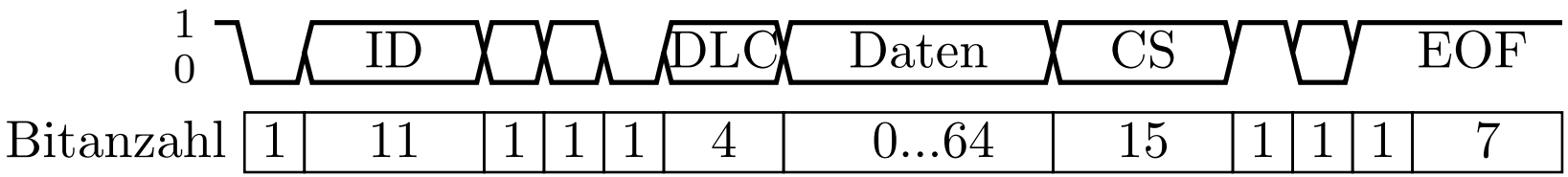


Erkennbare Formatfehler

- Paritätsfehler: Invertierung von 1, 3, 5, oder 7 Bit.
- Datenrahmenfehler: Datenwechsel in Zeitfenstern, in denen keine Änderung erlaubt. Zu kurze Start- oder Stoppbitzeit.
- Datenüberlauf: mehr ankommende Bytes als der Empfänger verarbeiten kann.

Überwachung durch die Prozessor-Hardware. Benachrichtigung der Software durch Setzen von durch die SW abfragbaren Bits in Spezialregistern der HW.

CAN-Bus



ID	Nachrichtennummer	CS	15-Bit Prüfzeichen
DLC	Datenlängencode	EOF	Ende des Datenrahmens

Eingesetzt z.B. zur Vernetzung von Fahrzeugsteuergeräten.

Erkennbare Formatfehler:

- 15-Bit Prüfzeichen, Erkennungssicherheit:

$$p_{\mathrm{D}} = 1 - 2^{-15} \approx 1 - 3 \cdot 10^{-5}$$

- Soll-/Ist-Abweichung konstante Bits, unzul. Datenänderungszeiten.

Fehlerfunktionsbehandlung:

- Wiederholung bei Nachrichtenkollision,
- Notfallreaktion bei Ausfall anderer Steuergeräte,
- Fehlerspeicher für aufgetretene Fehlfunktionen, ...

Ethernet-Protokoll

Sicherungsschicht			MAC-Empfänger	MAC-Absender	Protokolltyp	Nutzlast max. 1500 Bytes	Prüfzeichen 4 Byte	
Bitübertragungsschicht	Präambel	Startbyte						Lücke zum nächsten Paket
Byteanzahl	7	1	6	6	2	46 bis 1500	4	12

Erkennbare Formatfehler:

- 4-Byte-Prüfzeichen:

$$p_{\mathrm{D}} = 1 - 2^{-32} \approx 1 - 2 \cdot 10^{-10}$$

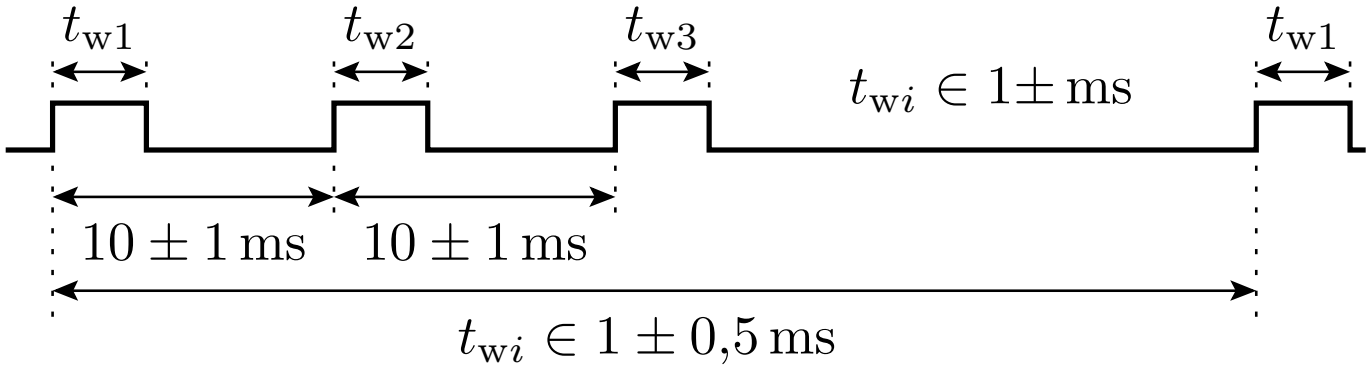
- korrekte Präambel, korrektes Startbyte, ...
- Wiederholanforderung bei fehlerhaft empfangenen oder nicht erhaltenen Datenpaketn, ...

Mittlere Zeit zwischen dem Empfang nicht erkannter verfälschter Datenpakete bei $10^5 \frac{\mathrm{DP}}{\mathrm{s}}$ (Datenpaketen pro s) und $10^{-4} \frac{\mathrm{MF}}{\mathrm{DP}}$ (Fehlfunktionen je Datenpaket):

$$\frac{1}{10^5 \frac{\mathrm{DP}}{\mathrm{s}} \cdot 10^{-4} \frac{\mathrm{MF}}{\mathrm{DP}} \cdot 2^{-32}} > 13{,}5\,\mathrm{Jahre}$$

Signalintegrität, Beispiel PWM

Pulsweitenmodulierte Signale (PWM) codieren die Information in den Pulsbreiten $t_{\mathrm{w}i}$ und werden u.a. zur Datenweitergabe von Sensoren an Rechner und von Rechnern an Aktoren genutzt.

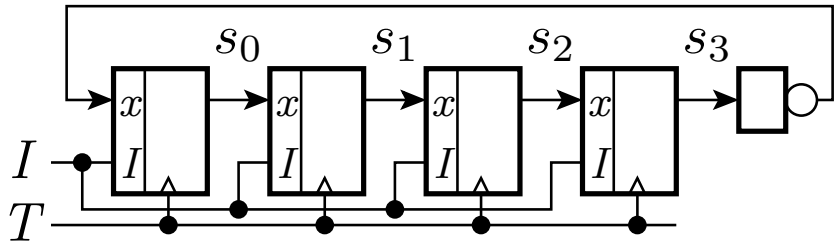


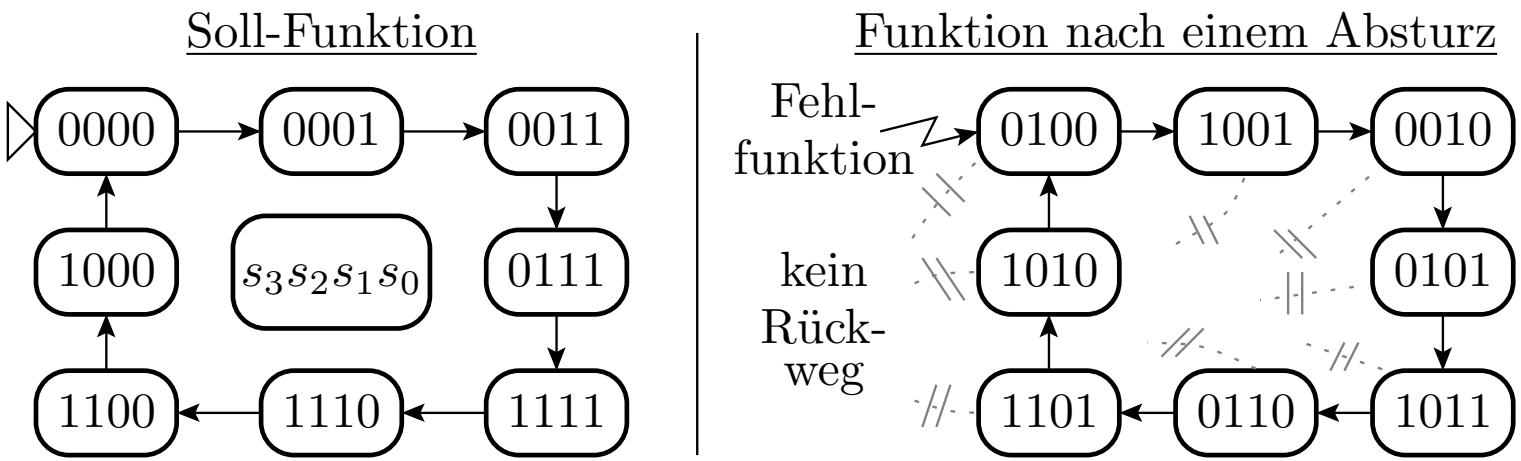
Überwachbare Formateigenschaften:

- Periodendauer,
- minimale und maximale Pulsbreite,
- Amplitude der Pulse.

Zeitüberwachung

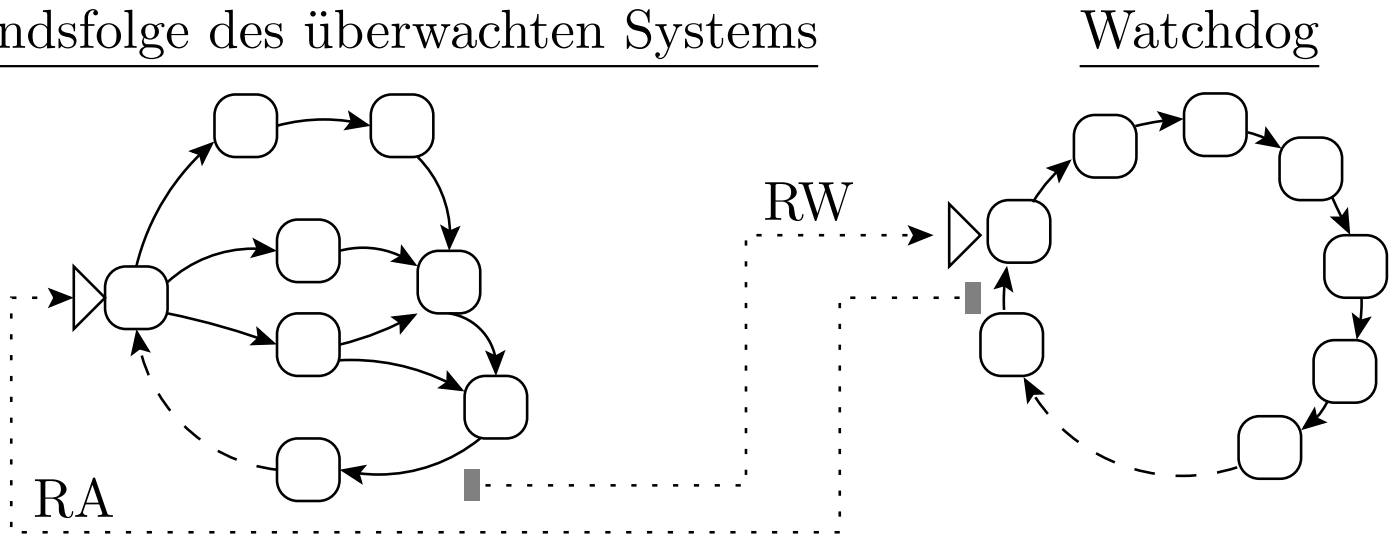
Redundante Zustände und Systemabsturz





- Automaten und Programme nutzen nur einen kleinen Teil der 2^n möglichen Zustände (n – Anzahl Zustandsbits).
- Wenn eine MF den Zustand so verfälscht, dass nie wieder ein zulässiger Zustand ereicht wird, Absturz.
- Bei komplexer HW und SW ist n um Zehnerpotenzen größer 4. Unüberschaubar viele Absturzmöglichkeiten.
- Abstürze sind lässtige, gut erkennbare, häufig beobachtete MF.

Watchdog



RA Ein Überlauf des Watchdogs initialisiert den Automaten neu.

RW Bei einem bestimmten, regelmäßig stattfindenden Zustandsübergang wird der Watchdog neu initialisiert.

Das überwachte System setzt in periodisch zu erreichenden Sollzuständen einen Zeitzähler zurück, der bei Überlauf das System neu startet und dabei auch wieder einen zulässigen Zustand herstellt.

Das Watchdog-Prinzip wird angewendet für

- einzelne Aufgaben,
- einzelne Programme in Multi-Task-Systemen,
- komplette Rechner, ...

Prozessoren haben in der Regel einen Hardware-Watchdog

- mit programmierbarer Zeit bis zum Überlauf,
- der, wenn eingeschaltet, nicht vom Programm, d.h. auch nicht durch Fehlfunktionen, ausschaltbar ist, sondern nur durch Neustart,
- bei Überlauf einen Betriebssystemaufruf zur Fehlerbehandlung und/oder einen Rechnerneustart auslöst.

[Debugger & Watchdog]

Invarianten

Invarianten

Invariante: die unabhängig von den zu verarbeitenden Daten immer erfüllt sein müssen. Überprüfbar

- durch Beweis, bei der Compilierung,
- durch Überwachung während der Laufzeit.

  [RUST Test Fail Fast, Release Fail Slow]

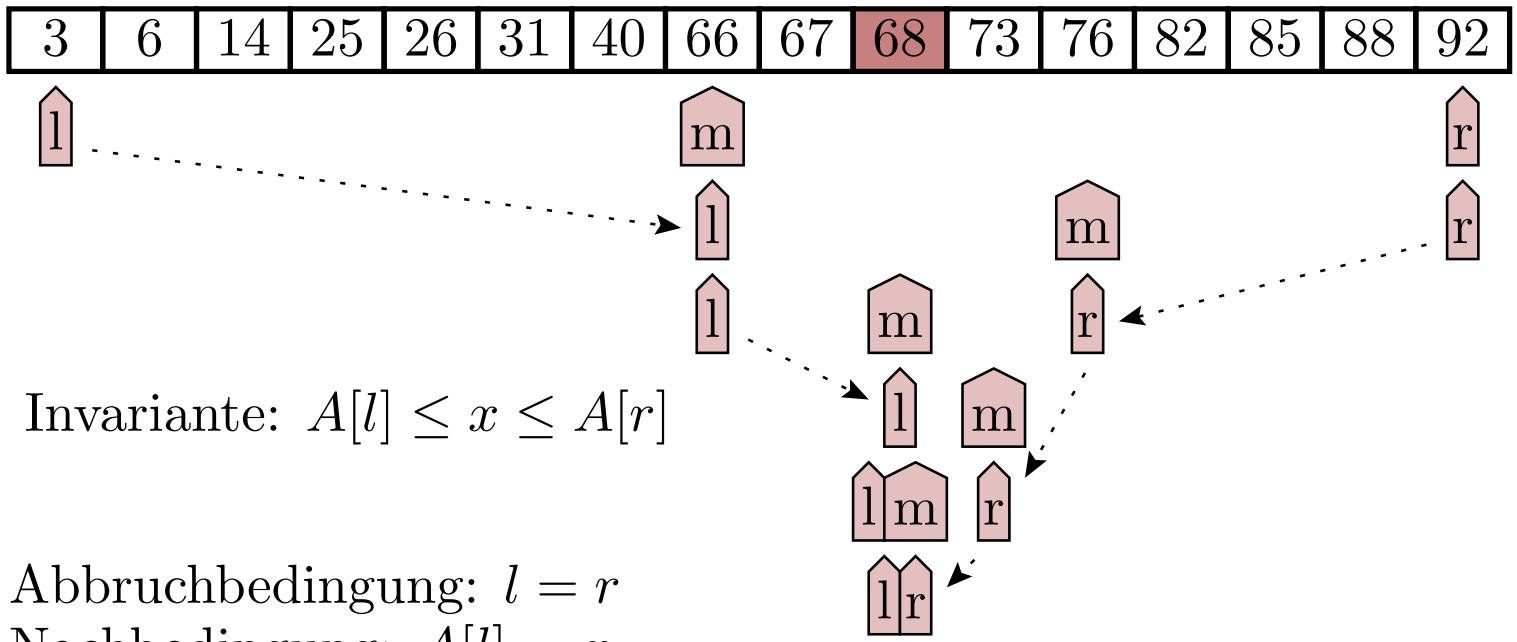
Beispiele für Invarianten:

- Sortieren einer Liste: Anzahl und Summe der Elemente.
- Task-Liste: die max. Anzahl der aktiven Tasks.
- Variable: Wertebereich,
- Zeiger: null oder Adresse eines Objekts mit zulässigem Typ.
- Geschlosses physikalisches System: Energie, Impuls, ...
- Iteration: Vorbedingungen, Schleifeninvarianten, Nachbedingungen.
- ...

Beispiel mit einer Schleifeninvarianten

gesucht: Position Schlüssel x

Vorbedingung: Feld $A[n]$ aufsteigend sortiert, n Elemente



Innerhalb der binären Suche nach dem Element mit Schlüsse x darf $A[l]$ nie größer und $A[r]$ nie kleiner als der gesuchte Schlüssel sein.

Syntax

Syntaxtest

Kontrolle, ob eine Zeichenfolge ein Wort einer formalen Sprache ist.

Formale Sprache: Definition zulässige Zeichenfolgen durch Syntaxregeln, deren Einhaltung sich durch einen spracherkennenden Automaten kontrollieren lässt.

Statischer Test für manuelle erstellte oder bearbeitete Programme und Programmeingaben.

Ein spracherkennenden Automat ist zwar selbst kein einfaches, schnell zu schreibendes Programm, aber das Programm für einen spracherkennenden Automaten lässt sich nach bekannten Algorithmen aus den Syntaxregeln der Sprache generieren.

Syntaxtests erkennen viele menschengemachter Fehler. Für maschinell erzeugte Daten, die nicht manuell zu bearbeiten sind, eignet sich die Weitergabe mit Prüfzeichen oder verschlüsselt mit fehlererkennenden oder korrigierenden Codes besser.

EBNF als Beispielregelwerk für formale Sprachen

Beschreibungsmittel der EBNF zur Definition von Sprachregeln:

Verwendung	Zeichen
Definition	NTS = Ersetzungsregel
Aufzählung	..., ...
Endezeichen	...;
Alternative	...|...
Option	[...]
Wiederholung	{...}
Gruppierung	(...)
Zeichenkette (Terminalsymbolfolge)	"..." oder '...'

EBNF Eweiterte Backus-Naur-Form.
NTS Nicht-Terminalsymbol, zu ersetzendes Symbol.

Beispiele für EBNF-Syntaxregeln

```
Zahl = ['-'],((ZiffernAusserNull,{Ziffer})|'0');
ZifferAusserNull = '1'|'2'|'3'| ... |'9';
Ziffer = ZifferAusserNull | '0';
```

Beispielzeichenfolgen:

```
'-1001': zulässig
'3,23' : nur bis Komma zulässig
'010'  : nur vor 1 zulässig
```

```
Bezeichner = Buchstabe,{Buchstabe|Ziffer},TZ;
TZ = ','|';'|Leerzeichen|...
```

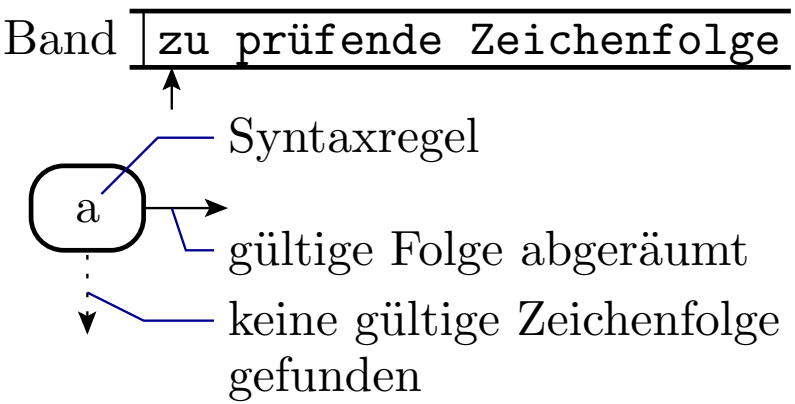
Beispielzeichenfolgen:

```
'a100;': zulässig,
'aa_3,': unzulässig wegen '_'
'1A'   : unzulässig wegen führender Ziffer
```

Spracherkennende Automaten

Die zu prüfende Zeichenfolge liegt auf einem Band mit einem Zeiger auf dem Anfang.

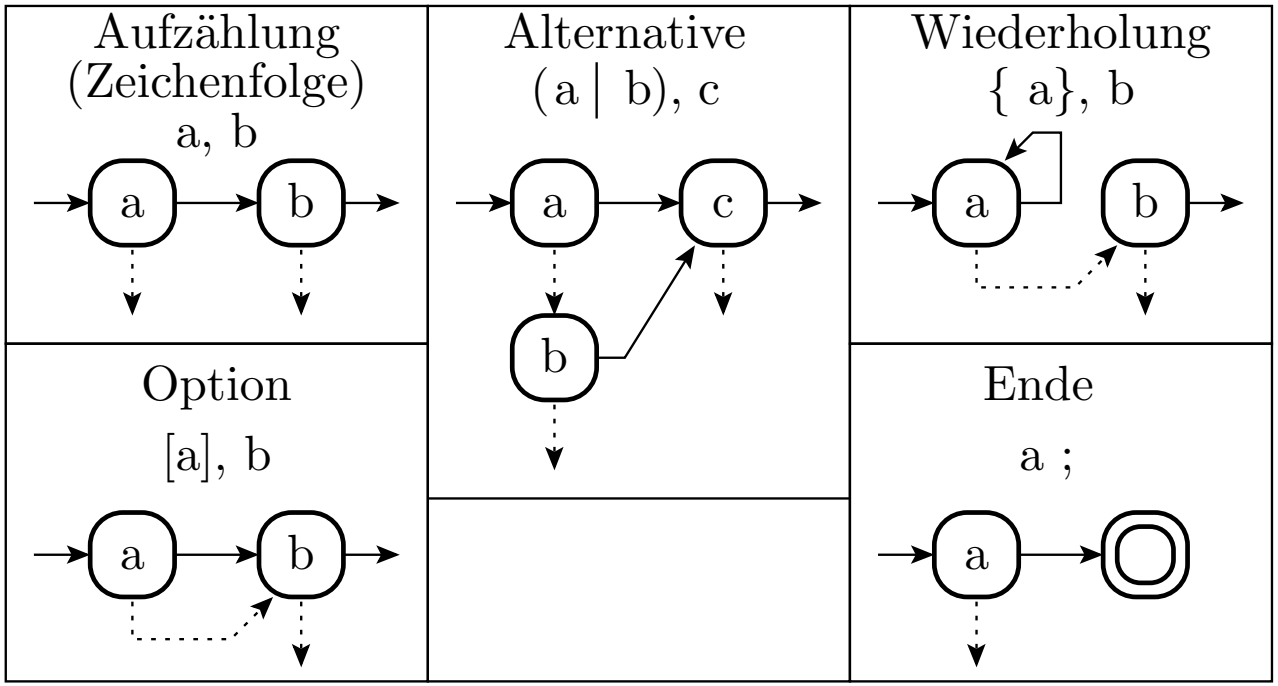
In jedem Automatenzustand wird versucht, eine Zeichenfolge nach der definierten Syntaxregel abzuräumen:



- Wenn möglich, wandert der Zeiger zum ersten Zeichen nach der abgeräumten Folge und der Knoten wird über $\longrightarrow$ verlassen.
- Sonst bleibt der Zeiger und der Knoten wird über $\downarrow$ verlassen.

$\downarrow$-Übergänge ohne dargestellten Zielknoten enden im Fehlerzustand.

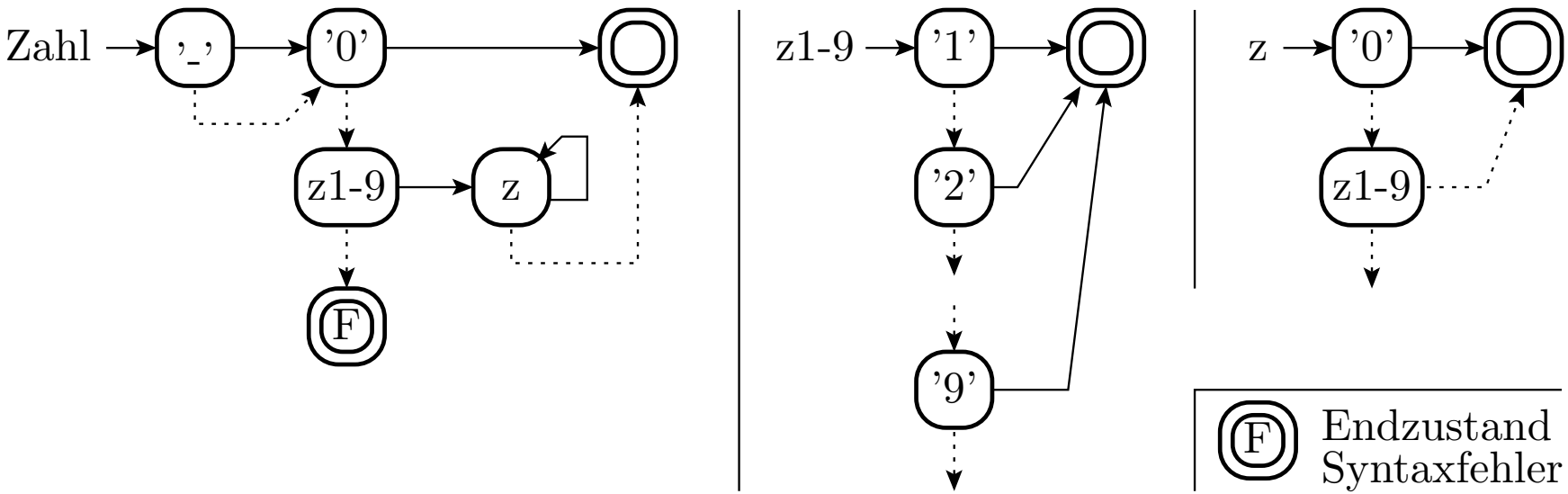
Von der EBNF zum Automaten



Beispiel:

```
Zahl = ['-'],((z1-9, {z}) | '0');
    z1-9 = '1' | '2' | ... | '9';
    z    = z1-9 | '0';
```

```
Zahl = ['-'],((z1-9, {z}) | '0');
z1-9 = '1' | '2' | ... | '9';
z    = z1-9 | '0';
```



Welche Zustände durchläuft der Automat. Was erkennt er?

- "125_": '-'↓, '0'↓, z1-9→, z→, z→, z↓, Endzustand: $125\underline{\surd}^{\downarrow}$
- "k89": '-'↓, '0'↓, z1-9↓, Endzustand: Syntaxfehler, $\overset{\downarrow}{-}$
- "-0701" : '-'→, '0' →, Endzustand: $0\overset{\downarrow}{\surd}7$

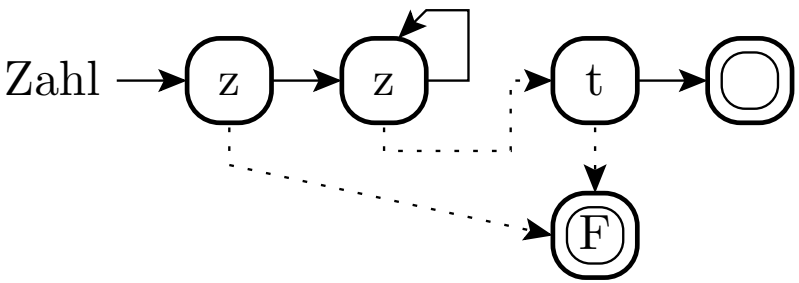
Abräumen an den Kanten

- Abräumen der Zeichen an den Kanten.
- Beschreibung derselben Syntaxregeln mit weniger Zuständen.
- Die »Sonst-Kanten« zum Fehlerzustand können weggelassen werden.

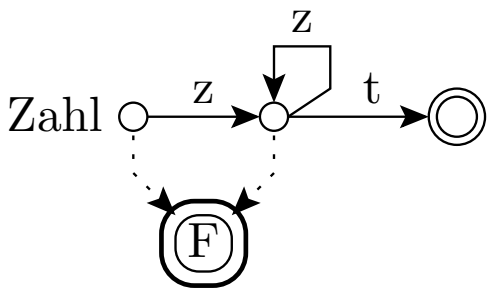
Beschreibung einer Zahl:

Zahl = z, {z}, t; z = '0' | '1' | ... | '9';
t = ' ' | ',' | ...; (Trennzeichen)

Abräumen in den Zuständen



Abräumen an den Kanten



Zusammenfassung

Kontrolle, Vergleich

Fehlkassifizierungen bei vergleichsbasierten Wertekontrollen (Mehrfachberechnung, Loop-Test, bekannte Sollwerte, ...):

- falsch bestimmte Sollwerte, z.B bei Zweifachberechnung und Vgl:

$$MC = \eta_{\mathrm{Div}} \tag{1.22}$$

$$\zeta_{\mathrm{Phan}} = \zeta_{\mathrm{Chk}} \cdot (1 - \eta_{\mathrm{Div}}) \tag{1.23}$$

- Vergleichsfehler.

Wenn der Vergleicher richtig funktioniert, ist für Einzelbits das Risiko für Vergleichsfehler praktisch immer vernachlässigbar.

Die Kontrolle analoger, Abtast- und numerischer Werte erfolgt über einen Fenstervergleich. Erkennungswahrscheinlichkeit hoch, aber Abschätzung schwierig. Phantom-MF-Rate:

$$\zeta_{\mathrm{Phan}} = \Phi\left(\frac{x_{\mathrm{min}} - \mu_{\mathrm{CM}}}{\sigma_{\mathrm{CM}}}\right) + 1 - \Phi\left(\frac{x_{\mathrm{max}} - \mu_{\mathrm{CM}}}{\sigma_{\mathrm{CM}}}\right) \tag{4.10}$$

$$\mathrm{sr}\,(x) = \mu_{\mathrm{CM}} \mp \sigma_{\mathrm{CM}} \cdot \Phi^{-1}\left(1 - \frac{\zeta_{\mathrm{Phan}}}{2}\right) \tag{4.11}$$

Vermeidbar durch ein Vergleichsfenster $\mu_{\mathrm{CM}} \mp 4 \ldots 6 \cdot \sigma_{\mathrm{CM}}$.

Informationsredundanz

Richtwerte für die Klassifizierungsfehler:

$$p_{\mathrm{D}} = 1 - \frac{\#PV}{\#V} \tag{1.19}$$

$$\zeta_{\mathrm{Phan}} = 0 \tag{1.20}$$

Voraussetzung: Gleichmäßiger pseudozufälliger Abbildung von Verfälschungen auf zulässige und unzulässige Werte. Nicht erfüllt z.B. bei:

- Rechtschreibtest durch Kontrolle ob Wörter im Wörterbuch:

$$p_{\mathrm{D}} \ll 1 - \frac{\#PV}{\#V}; \quad \zeta_{\mathrm{Phan}} > 0$$

- Paritätstest für Bitverfälschungen bei der Speicherung und Übertragung;

$$p_{\mathrm{D}} \approx 1 \gg 50\%$$

- Wertebereichstest:

$$p_{\mathrm{D}} \ll 1 - \frac{\#PV}{\#V}$$

aber es gibt viele Kontrollmöglichkeiten.

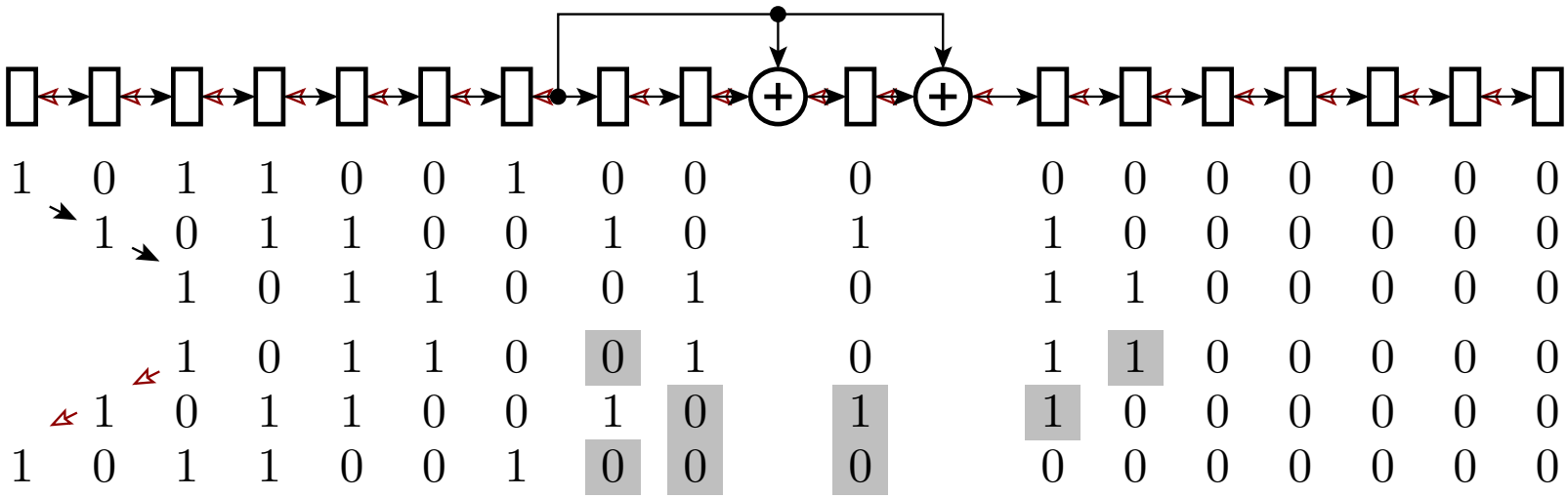
Fehlererkennende Codes

Pseudo-zufällige umkehrbar eindeutige Zuordnung von $\#PV$ zulässigen zu $\#V \gg \#PV$ möglichen Datenworten, z.B. durch Multiplikation einer großen Konstanten. Rückgewinnung durch Division und Kontrolle Divisionsrest:

$$p_{\mathrm{D}} = 1 - \frac{1}{\#K} \tag{4.13}$$

Bevorzugt für HW und SW: Codierung Polynommultiplikation, Rückgewinnung Polynomdivision mit r-Bit Schieberegistern:

$$p_{\mathrm{D}} \geq 1 - 2^{-r} \tag{1.21}$$



Prüfzeichen

Pseudo-zufällige Abbbildung großer Datenmassive auf ein r-Bit Prüfzeichen, das mit übertragen und gespeichert wird. Kontrolle durch nochmalige Bildung des Prüfkeinnzeichen für die gelesenen oder empfangenen Daten und Vergleich mit dem übertragenen Prüfzeichen. Erkennungswahrscheinlichkeit und Phantom-MF-Rate gleichfalls:

$$p_{\mathrm{D}} = 1 - \frac{\#PV}{\#V} \tag{1.19}$$

$$\zeta_{\mathrm{Phan}} = 0 \tag{1.20}$$

Bevorzugte Bildungsalgorithmen:

- arithmetische Aussummierung zu einer Prüfsumme,
- Exor-Summierung zu einer Prüfsumme,
- Bildung mit LFSR: seriell (1 Bit pro Schritt), parallel (Wort pro Schritt), mit dezentraler oder zentraler Rückführung, ...

Für LFSR werden solche mit primitiver Rückführung bevorzugt, d.h. r-Bit-LFSR, die im autonomen Betrieb alle $2^r - 1$ Zustände ungleich null in einem Maximalzyklus durchlaufen.

Protokolle, Zeitüberwachung

Protokolle beschreiben für Schnittstellen zwischen IT-Systemen und oder Funktionsbausteinen:

- welche Daten wie übergeben werden,
- einzuhaltende Vorbedingungen der Daten,
- Nutzungsbedingungen der Schnittstelle,
- zu übergebende Prüfzeichen,
- weitere kontrollierbare Kriterien wie Zeitschranken.

Als Beispiele wurden die Busprotokolle: USART, CAN und Ethernet und als Beispiel für einen einfachen Signalverlauf für die Informationsübergabe »PWM« erörtert.

Zeitüberwachung: Watchdog, vom Prinzip her ein freilaufener Zähler, der mit Lebenszeichen vom System rückgesetzt wird. Bei Ausbleiben der Lebenszeichen Fehlerbehandlung (Abbruch, Neuinitialisierung, ...)

Invarianten, Syntax

Beispielhaft wurde das Prinzip einer Schleifeninvarianten dargestellt.

Syntax-Kontrollen basieren auf der Datendarstellung durch formale Sprachen. Mit den einfachen Ersetzungsregeln »darf sein«, »darf n-mal vorkommen«, ... lassen sich viele Datenformate und auch Formate für Programme beschreiben.

Ein Programm für die Spracherkennung und Datenextraktion, dass auch noch verständliche Fehlermeldungen generiert, wird zwar schnell groß und kompliziert, lässt sich aber automatisch aus der Sprachbeschreibung generieren.

Syntaxtests mit spracherkennenden Automaten werden sowohl für statische Kontrollen von Programmbeschreibungen als auch zur Kontrolle manueller Eingaben genutzt. Respektable Erkennungswahrscheinlichkeit für manuelle Programmier- und Eingabefehler.

Für maschinell erzeugte Daten, die nicht manuell zu bearbeiten sind, eignet sich die Weitergabe mit Prüfzeichen oder verschlüsselt mit fehlererkennenden oder korrigierenden Codes besser.

Fehlertoleranz

Fehlertoleranz

(von lateinisch tolerare »erleiden, erdulden«)

Fehlertoleranz in der Technik: Aufrecterhalten der Funktion auch nach Auftreten eines Ausfalls oder einer internen Fehlfunktion durch Störungen, Fehler oder Fehlbedienung. Ziele von Fehlertoleranz

- Erhöhung der Verfügbarkeit und Lebensdauer durch Tolerierung von Ausfällen,
- Minderung der Rate der nach außen dringenden Fehlfunktionen und nicht erbringbaren Service-Leistungen durch Tolerierung von MF insbesondere durch Fehler,
- Erhöhung der Sicherheit gegenüber Datenverlusten durch redundante Datenspeicherung,
- Erhöhung der Betriebssicherheit durch einen Notbetrieb bei Ausfällen und internen MF.

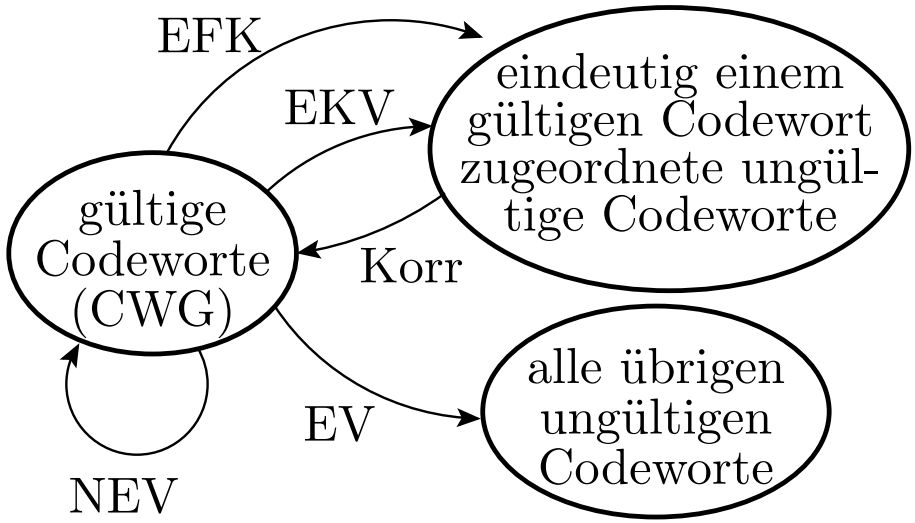
Massnahmen:

- RAID und Backup-Speicher,
- unterbrechungsfreie Stromversorgung (USV),
- Reserveeinheiten (kalt, warm, heiß, siehe Absch. 2.6.3), ...

Fehlerkorrigierende Codes

Fehlerkorrigierende Codes

EFK	erkennbar, aber falsch korrigierte Datenverfälschung
EKV	erkennbare und korrigierbare Datenverfälschung
EV	erkennbare, nicht korrigierbare Datenverfälschung
NEV	nicht erkennbare Datenverfälschung
Korr	Korrektur



Erweiterung der Menge der darstellbaren Codeworte um eine viel größere Menge korrigierbarer Codeworte und optional um unzulässige nicht korrigierbare Codeworte. Mindestbitanzahl:

$$2^{\#\mathrm{Bit}} \geq \#VCW + \#VCW \cdot \#CVC \tag{14}$$

Die Erkennungswahrscheinlichkeit als Anteil der übrigen ungültigen Codeworte verringert sich durch Korrekturmöglichkeiten.

$\#VCW$	Anzahl der gültigen Codeworte.
$\#CVC$	Anzahl korrigierbare Codeworte je gültiges Codewort.

Beispiel: Korrektur von Einzelbitfehler

Anzahl korrigierbare Codeworte je gültiges Codewort gleich Bitanzahl:

$$\#CVC = \#\text{Bit}$$

Mindestbitanzahl:

$$2^{\#\text{Bit}} \geq \#VCW + \#\text{Bit} \cdot \#VCW = (\#\text{Bit} + 1) \cdot \#VCW$$

Für $\#\text{VCW} = 256$ gültige Codeworte:

$$\begin{aligned} 2^{\#\text{Bit}} &\geq 256 \cdot (1 + \#\text{Bit}) \\ \#\text{Bit} &\geq 12 \end{aligned}$$

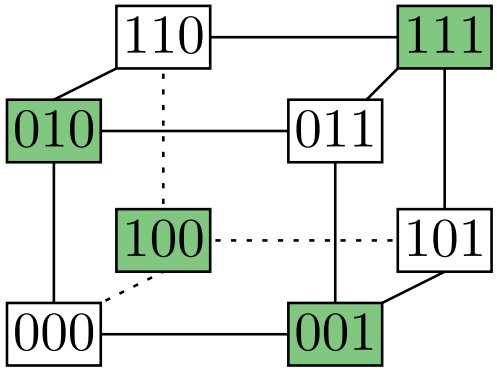
Probe:

$$2^{12} > 2^8 \cdot (1 + 12)$$

$\#Bit$	Bitanzahl des Codeworts.
$\#VCW$	Anzahl der gültigen Codeworte.
$\#CVC$	Anzahl korrigierbare Codeworte je gültiges Codewort.

Hamming-Codes

Hamming Codes werden durch die Hamming-Distanz Ham beschrieben. Das ist die Anzahl der Bitpositionen, in denen sich zwei Codeworte mindestens unterscheiden. Distanz von 2 oder mehr garantiert, dass eine 1-Bit Verfälschung nicht zu einem anderen gültigen Codewort führt.



- Erkennen alle Codeworte mit bis zu d verfälschten Bits:

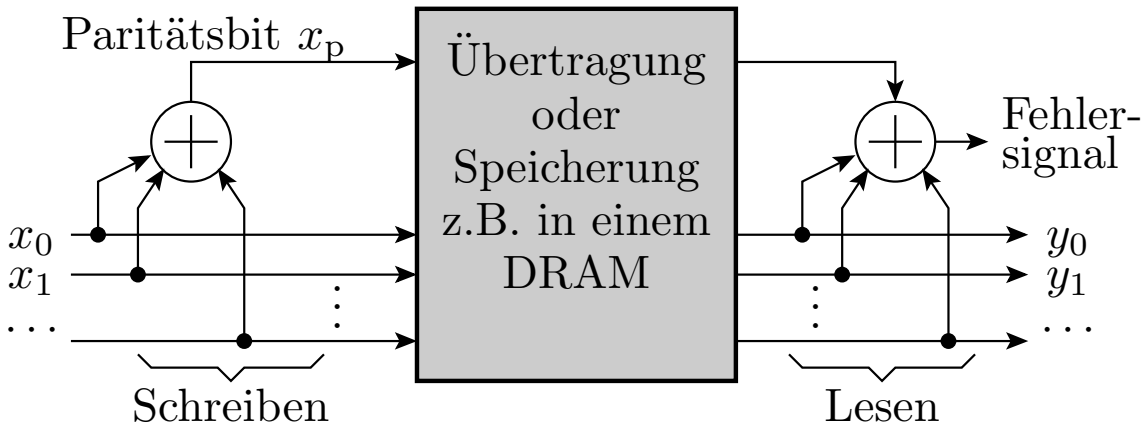
$$Ham \geq d + 1 \tag{15}$$

- Korrektur alle Codeworte mit bis zu c verfälschten Bits:

$$Ham \geq 2 \cdot c + 1 \tag{16}$$

Ham Hamming-Distanz.

Paritätsbit als Prüfzeichen ($Ham = 2$)



Einzelprüfbit, modulo-2 Summe (EXOR-Verknüpfung):

$$x_\mathrm{p} = x_{n-1} \oplus x_{n-2} \oplus \ldots \oplus x_1 \oplus x_0$$

bei gerader Anzahl von Einsen »0« sonst »1«.

Erkennt ungeradzahlige und insgesamt 50% der Bitverfälschungen.
Bei pseudozufälliger gleichmäßiger Abbildung nach

$$p_\mathrm{D} \geq 1 - 2^{-r} \tag{1.21}$$

Für $r = 1$ Prüfbit $p_\mathrm{D} = 50\%$.

p_D Nachweiswahrscheinlichkeit (detection probability).

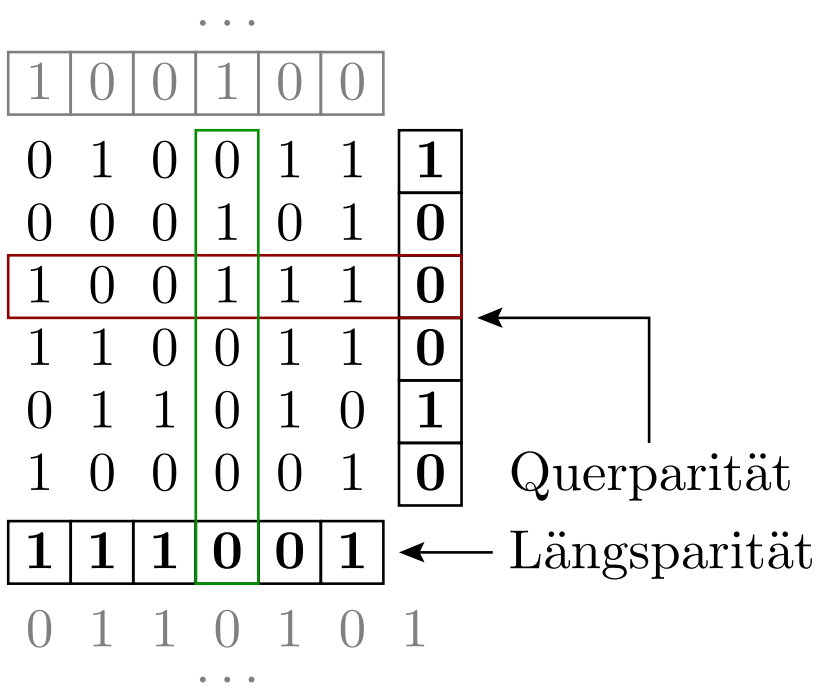
- Beim Lesen gespeicherter und Empfang übertragender Daten unter Ausschluss von Verfälschungen mehrerer Bits im selben Datenwort durch dieselbe Ursache

$$p_{\mathrm{D}} = 1 - \lambda \tag{4.12}$$

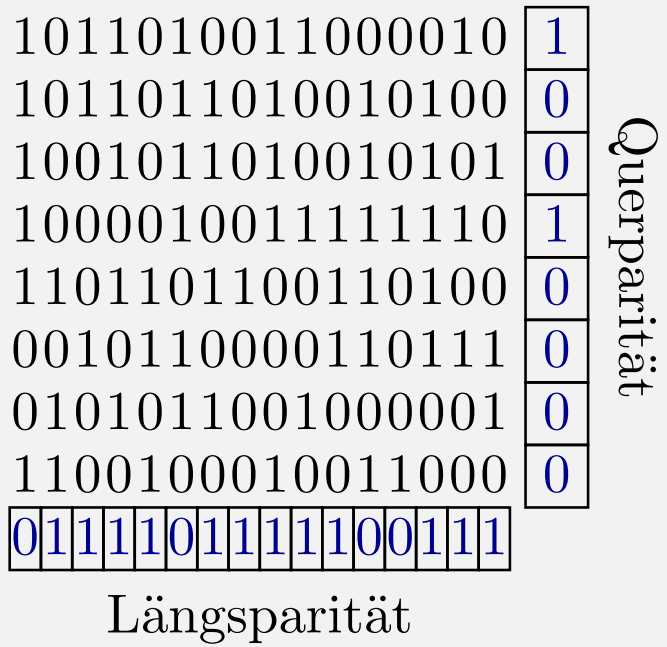
p_{D}	Nachweiswahrscheinlichkeit (detection probability).
λ	zu erwartende Bitfehleranzahl je Datenwort.

Kreuzparität (Fehlerkorrigierender Paritätscode)

Daten sind in einem 2-dimensionalen Array organisiert. Paritätsbildung für alle Zeilen und Spalten. Erlaubt Lokalisierung und Korrektur von 1-Bit Fehlern. Einsatz in redundanten Festplatten-Arrays (RAID 3 und RAID 5).



Beispiel 4.4: Kreuzparität

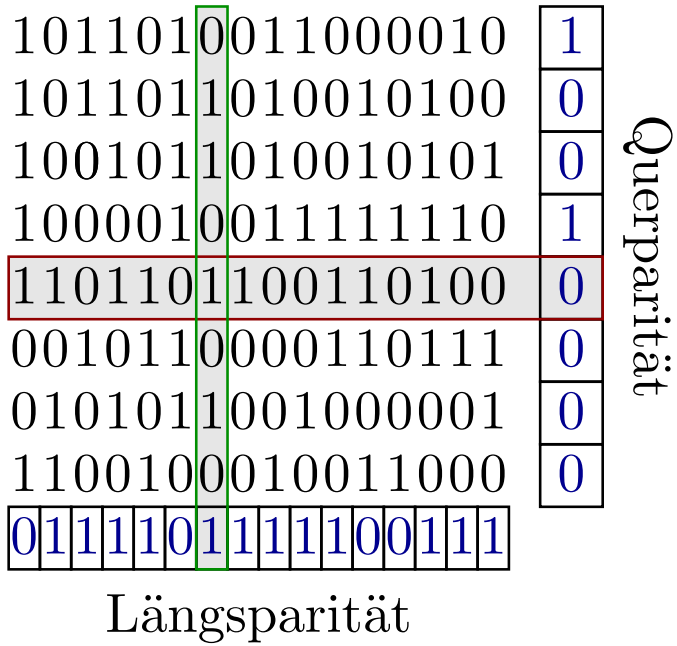


a) Kontrolle der Zeilen- und Spaltenparität.

b) Liegt keine, eine erkennbare nicht korrigierbare oder eine erkenn- und korrigierbare Verfälschung vor?

a) Kontrolle der Zeilen- und Spaltenparität.
b) Liegt keine, eine erkennbare nicht korrigierbare oder eine erkenn- und korrigierbare Verfälschung vor?

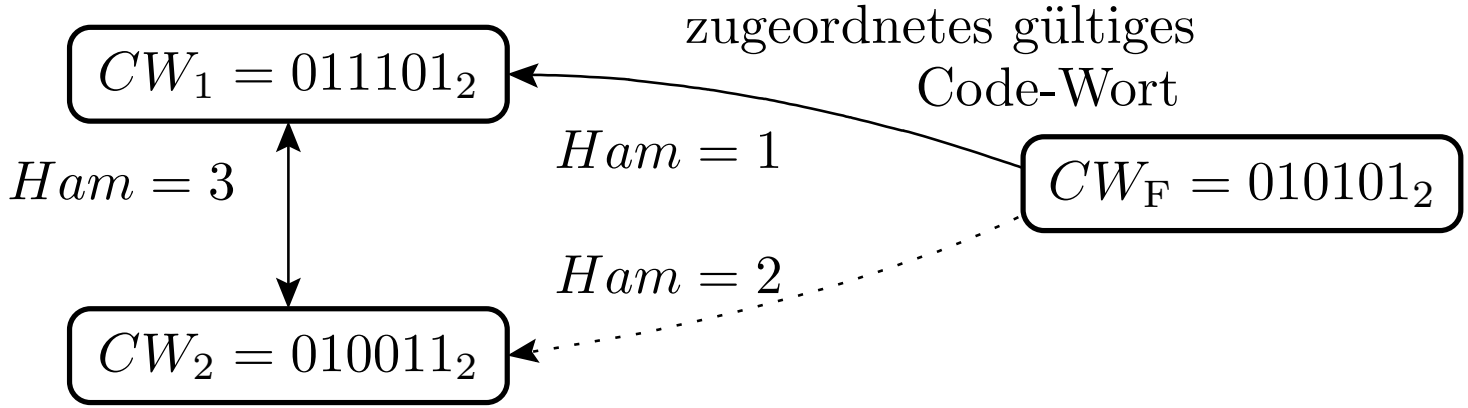
a) Kontrolle der Zeilen- und Spaltenparität:



b) Korrigierbar durch Invertierung des Bits in Zeile 5, Spalte 7 (null setzen).

1-Bit fehlerkorrigierende Hamming-Codes

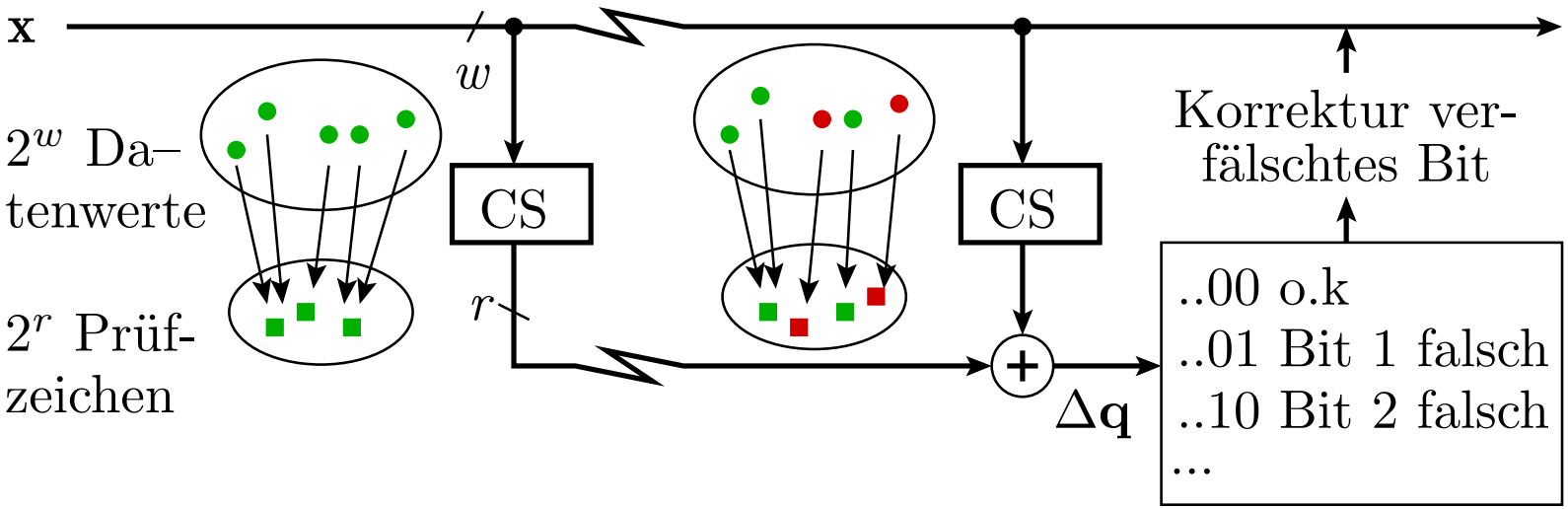
Ab einem Hamming-Abstand $Ham \geq 3$ ist jede 1-Bit-Verfälschung eindeutig einem gültigen Codewort zugeordnet.



Korrektur durch Ersatz des verfälschten Codeworts durch das mit Hamming-Distanz $Ham = 1$. Bei Hamming-Distanz $Ham = 3$ werden Codeworte mit zwei oder mehr verfälschten Bits falschen gültigen Codeworten zugeordnet.

Ham	Hamming-Distanz.
CW_i	Code-Wort i.
CW_F	verfälschtes Code-Wort.

Konstruktion 1-Bit fehlerkorrigierender Code



Ergänzung des w-Bit Datenworts **x** um ein r-Bit Prüfzeichen so, dass die mod-2 Summen des übertragenen und des nach der Übertragung gebildeten Prüfzeichens die Bitnummer des verfälschten Bits ist:

verfälschtes Bit	1	2	3	4	5	...
Prüfzeichendifferenz $\Delta\mathbf{q}$	..001	..010	..011	..100	..101	...

CS Bildung Prüfzeichen (**c**heck **s**ymbol generation).
w Datenbitanzahl (number of data bits).
r Anzahl der Prüfzeichenbits.

Prüfzeichenbildung für 1 Byte Einzelbitkorrektur

Anzahl der Prüfbits r nach Gl. 4.14 für $w = 8$:

$$2^{w+r} \geq \left(1 + \underbrace{w + r}_{\text{Anz. Einzelbitfehler}}\right) \cdot \underbrace{2^w}_{\#CV} = (1 + 12) \cdot 2^8;\ r = 4$$

verfälschtes Bit	1	2	3	4	5	...
Prüfzeichendifferenz $\Delta\mathbf{q}$	0001	0010	0011	0100	0101	...

$$\begin{aligned}
\Delta q_0 &= \underline{b_1} \oplus b_3 \oplus b_5 \oplus b_7 \oplus b_9 \oplus b_{11} \\
\Delta q_1 &= \underline{b_2} \oplus b_3 \oplus b_6 \oplus b_7 \oplus b_{10} \oplus b_{11} \\
\Delta q_2 &= \underline{b_4} \oplus b_5 \oplus b_6 \oplus b_7 \oplus b_{12} \\
\Delta q_3 &= \underline{b_8} \oplus b_9 \oplus b_{10} \oplus b_{11} \oplus b_{12}
\end{aligned}$$

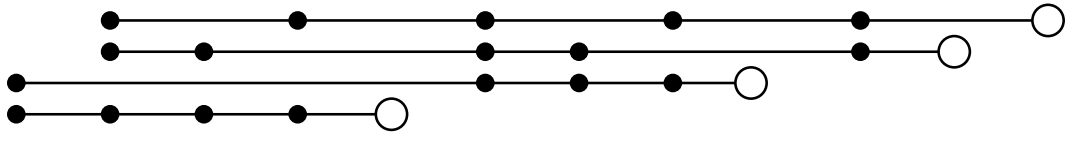
Die unterstrichenen Bits jeder Prüfsumme seinen die Prüfbits q_i.

q_i	Prüfbits.
x_i	Datenbits.
w	Datenbitanzahl (number of data bits).
r	Anzahl der Prüfzeichenbits.

Bitzuordnung, Prüfbit-Berechnung

Bitnummer	12	11	10	9	8	7	6	5	4	3	2	1
Zuordnung	x_7	x_6	x_5	x_4	q_3	x_3	x_2	x_1	q_2	x_0	q_1	q_0

○ Prüfbit
• Datenbit



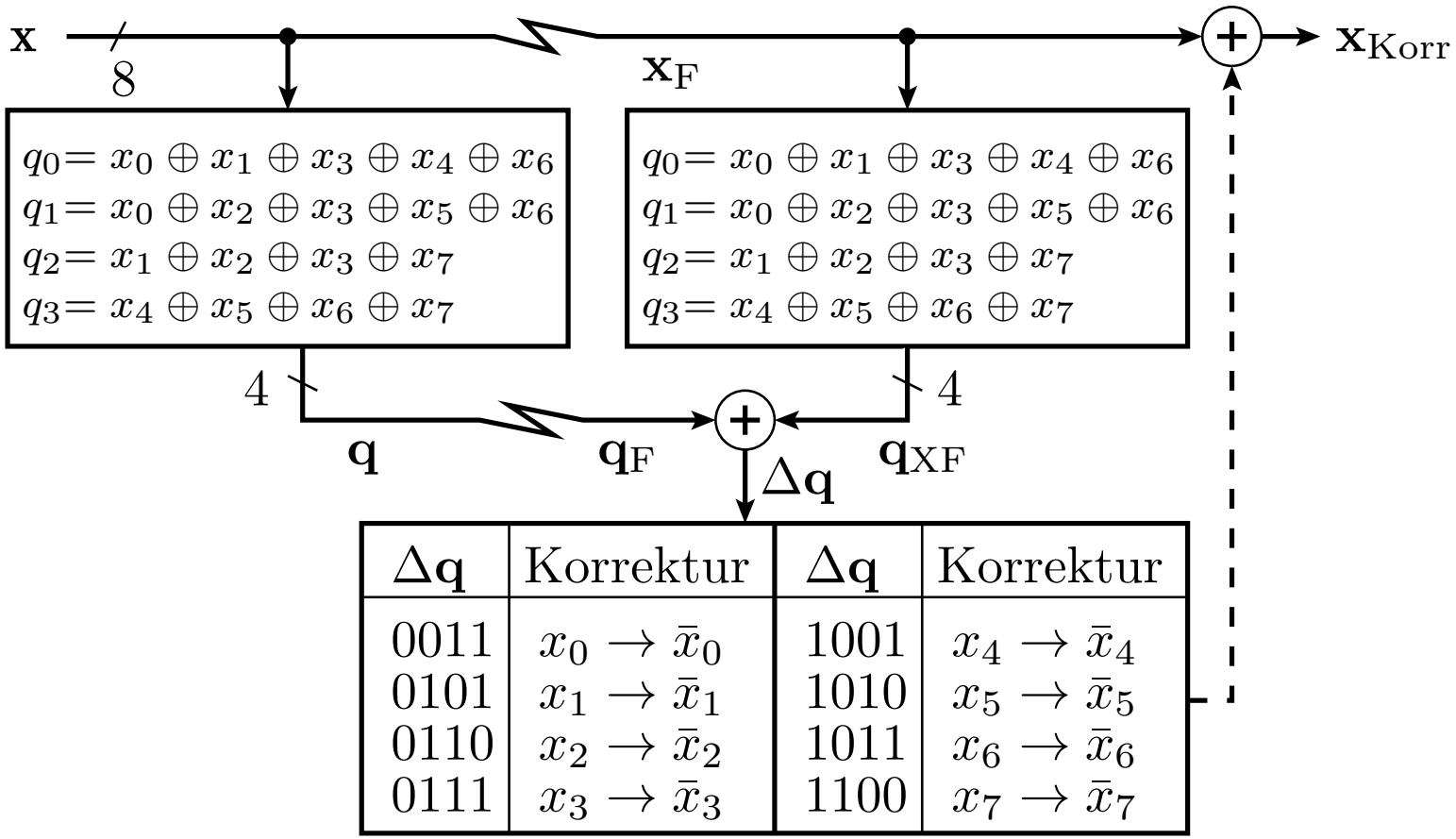
$$\begin{aligned}
\Delta q_0 &= b_1 \oplus b_3 \oplus b_5 \oplus b_7 \oplus b_9 \oplus b_{11} &&= q_0 \oplus x_0 \oplus x_1 \oplus x_3 \oplus x_4 \oplus x_6 \\
\Delta q_1 &= b_2 \oplus b_3 \oplus b_6 \oplus b_7 \oplus b_{10} \oplus b_{11} &&= q_1 \oplus x_0 \oplus x_2 \oplus x_3 \oplus x_5 \oplus x_6 \\
\Delta q_2 &= b_4 \oplus b_5 \oplus b_6 \oplus b_7 \oplus b_{12} &&= q_2 \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_7 \\
\Delta q_3 &= b_8 \oplus b_9 \oplus b_{10} \oplus b_{11} \oplus b_{12} &&= q_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7
\end{aligned}$$

Auflösung nach q_i für $\Delta\mathbf{q} = 0$:

$$\begin{aligned}
q_0 &= x_0 \oplus x_1 \oplus x_3 \oplus x_4 \oplus x_6 \\
q_1 &= x_0 \oplus x_2 \oplus x_3 \oplus x_5 \oplus x_6 \\
q_2 &= x_1 \oplus x_2 \oplus x_3 \oplus x_7 \\
q_3 &= x_4 \oplus x_5 \oplus x_6 \oplus x_7
\end{aligned}$$

q_i	Prüfbits.
x_i	Datenbits.

Codier-, Erkennungs- und Korrekturschaltung



$\Delta\mathbf{q}$	Korrektur	$\Delta\mathbf{q}$	Korrektur
0011	$x_0 \to \bar{x}_0$	1001	$x_4 \to \bar{x}_4$
0101	$x_1 \to \bar{x}_1$	1010	$x_5 \to \bar{x}_5$
0110	$x_2 \to \bar{x}_2$	1011	$x_6 \to \bar{x}_6$
0111	$x_3 \to \bar{x}_3$	1100	$x_7 \to \bar{x}_7$

- Gleiches Prüfbit-Bildung vor und nach Speichern/ Übertragung.
- Differenzbildung durch bitweises EXOR.
- Wenn $\Delta\mathbf{q} \neq 0$ Datenbit zugeordnet, Korrektur durch Invertierung.

Beispiel 4.5: Fehlerkorrigierender Code

b_{12}	b_{11}	b_{10}	b_9	b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1
x_7	x_6	x_5	x_4	q_3	x_3	x_2	x_1	q_2	x_0	q_1	q_0

$$q_0 = x_0 \oplus x_1 \oplus x_3 \oplus x_4 \oplus x_6 \qquad q_2 = x_1 \oplus x_2 \oplus x_3 \oplus x_7$$
$$q_1 = x_0 \oplus x_2 \oplus x_3 \oplus x_5 \oplus x_6 \qquad q_3 = x_4 \oplus x_5 \oplus x_6 \oplus x_7$$

a) Bilden Sie für den Datenwert $\mathbf{x}_\mathrm{a} = 0\text{x}8\text{B}$ das Codewort $\mathbf{b}_\mathrm{a}$.

b) Bestimmen Sie für $\mathbf{b}_\mathrm{b} = 0\text{xA}9\text{B}$ den codierten Wert $\mathbf{x}_\mathrm{b}$.

b_{12}	b_{11}	b_{10}	b_9	b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1
x_7	x_6	x_5	x_4	q_3	x_3	x_2	x_1	q_2	x_0	q_1	q_0

$$q_0 = x_0 \oplus x_1 \oplus x_3 \oplus x_4 \oplus x_6 \qquad q_2 = x_1 \oplus x_2 \oplus x_3 \oplus x_7$$
$$q_1 = x_0 \oplus x_2 \oplus x_3 \oplus x_5 \oplus x_6 \qquad q_3 = x_4 \oplus x_5 \oplus x_6 \oplus x_7$$

a) Bilden Sie für den Datenwert $\mathbf{x}_\mathrm{a} = 0\mathrm{x}8\mathrm{B}$ das Codewort $\mathbf{b}_\mathrm{a}$.

b) Bestimmen Sie für $\mathbf{b}_\mathrm{b} = 0\mathrm{xA9B}$ den codierten Wert $\mathbf{x}_\mathrm{b}$.

Bitnummer	12	11	10	9	8	7	6	5	4	3	2	1	
Zuordnung	x_7	x_6	x_5	x_4	q_3	x_3	x_2	x_1	q_2	x_0	q_1	q_0	
Kontrollbits													
$\mathbf{x}_\mathrm{a} = 0\mathrm{x}8\mathrm{B}$													$\mathbf{b}_\mathrm{a} =$
$\mathbf{b}_\mathrm{b} = 0\mathrm{xA9B}$													$\Delta\mathbf{q}_\mathrm{b} =$

a) Bilden Sie für den Datenwert $\mathbf{x}_\mathrm{a} = 0\mathrm{x}8\mathrm{B}$ das Codewort $\mathbf{b}_\mathrm{a}$.

b) Bestimmen Sie für $\mathbf{b}_\mathrm{b} = 0\mathrm{xA9B}$ den codierten Wert $\mathbf{x}_\mathrm{b}$.

Bitnummer	12	11	10	9	8	7	6	5	4	3	2	1	
Zuordnung	x_7	x_6	x_5	x_4	q_3	x_3	x_2	x_1	q_2	x_0	q_1	q_0	
Kontrollbits													
$\mathbf{x}_\mathrm{a} = 0\mathrm{x}8\mathrm{B}$	1	0	0	0	1	1	0	1	1	1	0	1	$\mathbf{b}_\mathrm{a} = 0\mathrm{x}8\mathrm{DD}$
$\mathbf{b}_\mathrm{b} = 0\mathrm{xA9B}$	1	0	1	0	1	0	0	1	1	0	1	1	$\Delta\mathbf{q}_\mathrm{b} = 12_{10}$

a) Der Wert 0x8B hat das Code-Wort 0x8DD

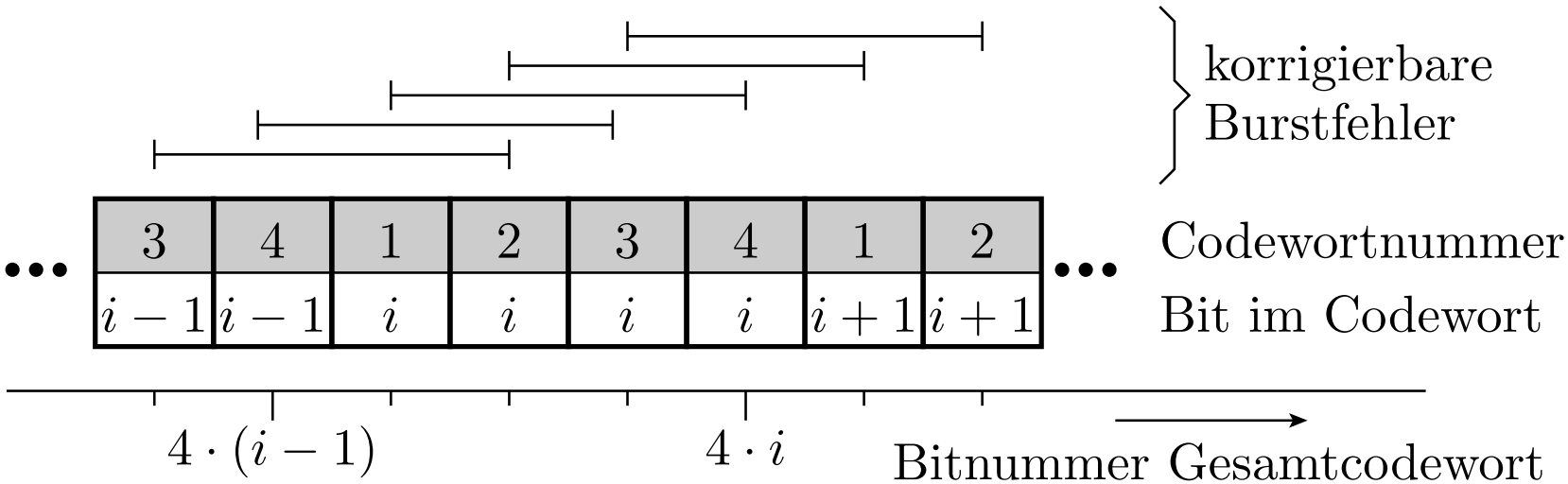
b) Im Code-Wort 0x49B steckt der Wert 0xA2 mit $\Delta\mathbf{q} = 0\mathrm{b}1100 = 12$. Das verfälschte 12. Bit im Code-Wort ist Datenbit x_7. Invertierung von x_7 ergibt den Datenwert $\mathbf{x}_\mathrm{b} = 0\mathrm{x}22$.

Korrigierbare Burstfehler durch Verschränkung

Burstfehler

Verfälschung von bis zu m aufeinanderfolgende Bits. Typisch für Lesefehler von CDs und Übertragungsfehler.

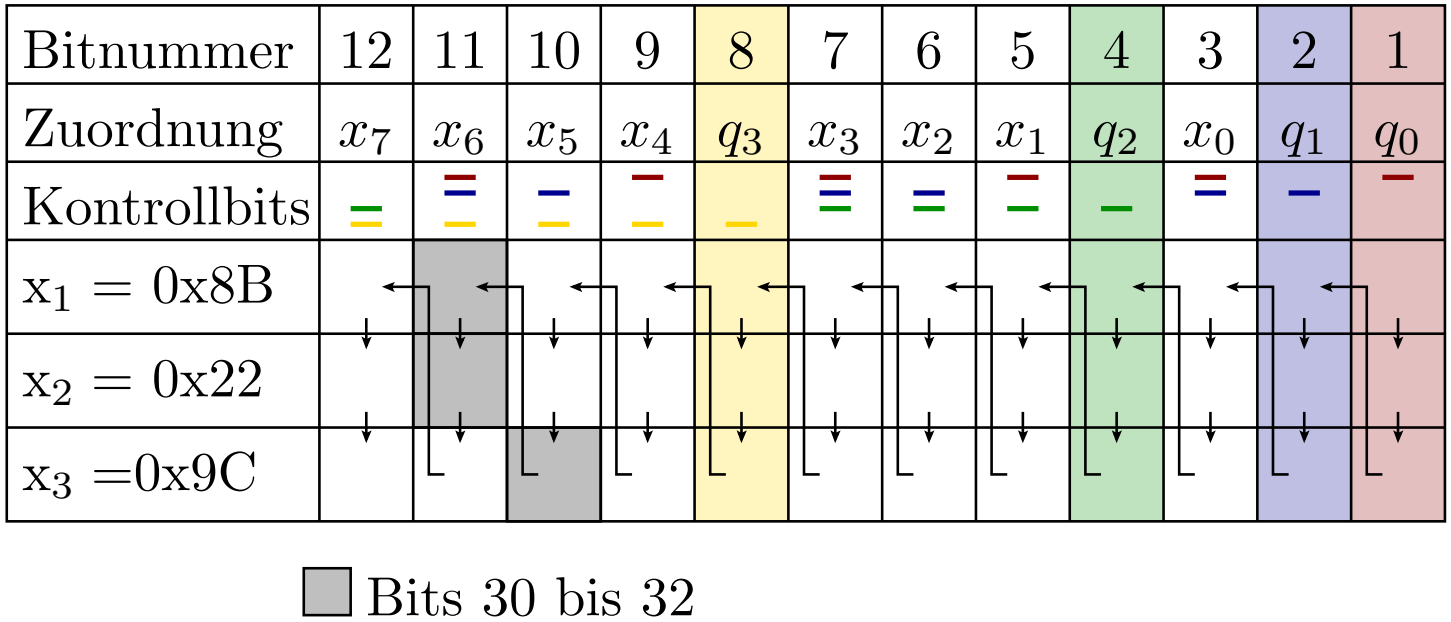
Zusammensetzen eines fehlerkorrigierenden Codes für m-Bit Burstfehler für eine $m \cdot n$ Bit lange Folgen aus m fehlerkorrigierenden Codeworten für 1-Bit-Fehler für n Bit lange Folgen durch Verschränkung.



Beispiel 4.6: Burstfehler

a) Codierung Datenfolge 0x8B, 0x22, 0x9C so, dass bis zu 3-Bit lange Burstfehler korrigierbar sind, durch Verschränkung von je drei aufeinanderfolgenden H8-12-Codeworten.

b) Zeigen Sie, dass eine Invertierung der Bits 30 bis 32 korrigiert wird.



a) Codierung Datenfolge 0x8B, 0x22, 0x9C so, dass bis zu 3-Bit lange Burstfehler korrigierbar sind, durch Verschränkung von je drei aufeinanderfolgenden H8-12-Codeworten.

Bitnummer	12	11	10	9	8	7	6	5	4	3	2	1
Zuordnung	x_7	x_6	x_5	x_4	q_3	x_3	x_2	x_1	q_2	x_0	q_1	q_0
Kontrollbits	=	=	=	=	=	=	=	=	=	=	=	=
x_1 = 0x8B	1	0	0	0	1	1	0	1	1	1	0	1
x_2 = 0x22	0	0	1	0	1	0	0	1	1	0	1	1
x_3 =0x9C	1	0	0	1	0	1	1	0	1	0	0	0

b) Zeigen Sie, dass eine Invertierung der Bits 30 bis 32 korrigiert wird.

Bitnummer	12	11	10	9	8	7	6	5	4	3	2	1	
Zuordnung	x_7	x_6	x_5	x_4	q_3	x_3	x_2	x_1	q_2	x_0	q_1	q_0	
Kontrollbits	=	=	=	=	=	=	=	=	=	=	=	=	
$x_1 = 0x8B$	1	0	0	0	1	1	0	1	1	1	0	1	$\Delta\mathbf{q} = 1011_2$
$x_2 = 0x22$	0	0	1	0	1	0	0	1	1	0	1	1	$\Delta\mathbf{q} = 1011_2$
$x_3 = 0x9C$	1	0	0	1	0	1	1	0	1	0	0	0	$\Delta\mathbf{q} = 1010_2$

☐ Durch Burstfehler invertierte Bits 30 bis 32,

RAID und Backup

RAID 0 und 1

Organisation mehrerer physischer Massenspeicher (Festplatten oder SSD) zu einem logischen Laufwerk. Mit RAID können Systeme Ausfälle eines oder mehrerer Festplatten oder SSD ohne Datenverlust und in vielen Fällen sogar ohne Ausfallzeiten überstehen.

RAID 0: Verteilung der Daten blockweise auf mehrere Festplatte so, dass sich der Datendurchsatz erhöht. Die Speicherkapazitäten summieren sich, aber keine Tolerierung von Plattenausfällen.

RAID 1: Zwei gespiegelte Festplatten. Die Daten werden versetzt geschrieben, so dass das Schreiben etwas länger dauert, aber mit nahe doppelter Geschwindigkeit gelesen werden kann. Bei Ausfall einer Platte existieren alle Daten noch auf der zweiten Festplatte. Die Lesegeschwindigkeit reduziert sich, aber das System bleibt funktionsfähig.

RAID	Redundantes Array unabhÃ€ngiger Festplatten.
SSD	**S**olid **S**tate **D**rive, Festplattennachbildung mit Halbleiterspeichern.

RAID 0+1: Verschaltung der Hälfte der Festplatten zu einem RAID 0 und Spiegelung auf die andere Hälfte. Tolerierung von Plattenschäden in einem der beiden RAID 0, solange das andere RAID 0 intakt ist.

RAID 10: Verschaltung paarweise gespiegelter Platten zu einem RAID 0. Tolerierung von einem Plattenschaden je gespiegeltes Plattenpaar.

RAID mit Paritätsblöcken (RAID 2 bis 7)

	Paritätsbildung	Ausfall Disc 2	Ausfall Disc 4
Disc 1: Daten	0001 0011 0010	0001 0011 0010	0001 0011 0010
Disc 2: Daten	1101 0101 0000	xxxx xxxx xxxx	1101 0101 0000
Disc 3: Daten	1101 1111 1100	1101 1111 1100	1101 1111 1100
Disc 4: Parität	**0001 1001 1110**	0001 1001 1110	xxxx xxxx xxxx
	Wiederherstellung:	**1101 0101 0000**	**0001 1001 1110**

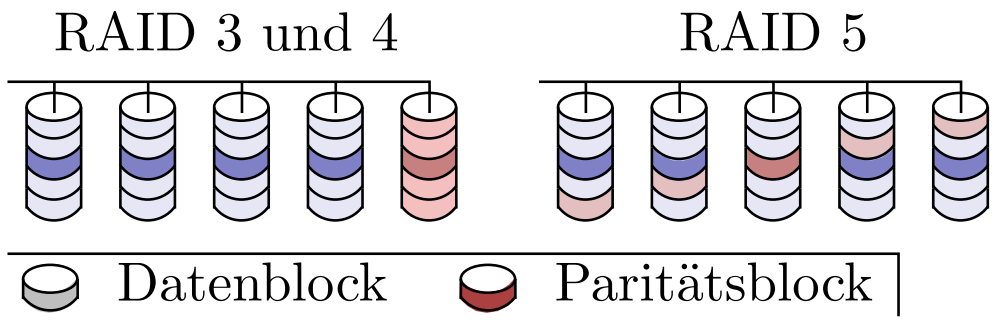
Datenverteilung wie bei RAID 0 blockweise über mehrere Platten.
Bildung und Speicherung von Paritätsblöcken so, dass

- nur Blöcke unterscheidlicher Platten zusammengefasst werden,
- der Paritätsblock auf noch einer anderen Platte gespeichert wird.

Nach Ausfall einer Platte lassen sich alle Blöcke auf der ausgefallenen Platte aus denen auf anderen Platten gespeicherten Blöcken durch bitweise EXOR rekonstruieren, vorausgesetzt es ist bekannt

- welche Platte ausgefallen ist und
- und wo die zugehörigen ver-EXOR-ten Blöcke stehen.

Verteilung der Paritätsblöcke auf Platten



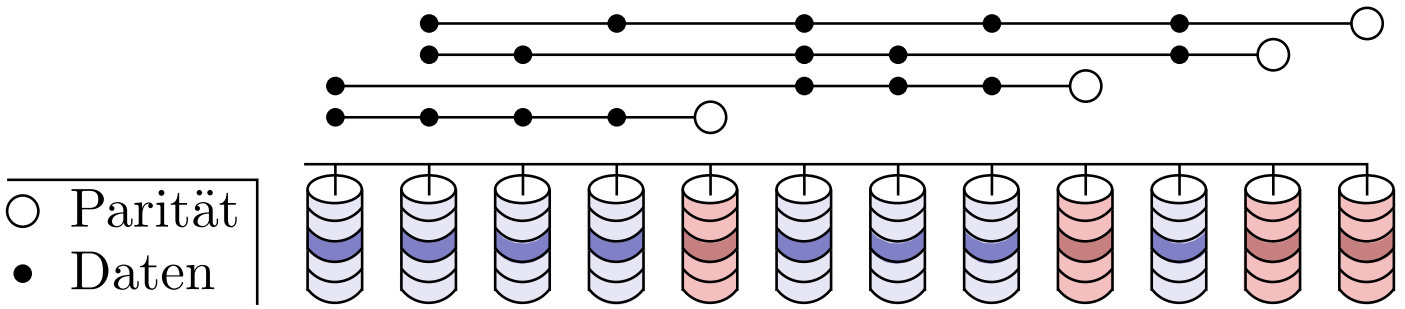
Schreiben eines Blocks:

- Block lesen, Paritätsänderung bestimmen, schreiben,
- Paritätsblock lesen, ändern, schreiben.

Wenn die Paritätsblöcke alle auf derselben Festplatte stehen (RAID 3 und 4), viel mehr Schreib-Lese-Zugriffe auf diese. Verkürzte Lebensdauer. Alternative ist gleichmäßige Verteilung der Paritätsblock auf alle Platten (RAID 5).

Erweiterte Fehlertoleranz

Festplatte	12	11	10	9	8	7	6	5	4	3	2	1
	x_7	x_6	x_5	x_4	q_3	x_3	x_2	x_1	q_2	x_0	q_1	q_0



Statt einem Paritätsblock, mehrere nach dem Prinzip der 1-Bit-Fehlerkorrektur. Toleriert nicht nur komplette Plattenausfälle, sondern auch einzelnen Bitverfälschungen (RAID 2).

Weitere Ansätze:

- Zusätzliche Längsparitätsblöcke zur Wiederherstellung einzelner verfälschter Blöcke nicht komplett ausgefallener Platten nach dem Prinzip der Kreuzparität.
- Fehlertoleranz für mehr als einen zeitgleichen Plattenausfall, ...

Wiederherstellung, weitere Schutzmaßnahmen

Ersatz der ausgefallenen Platte vor dem nächsten Ausfall:

- Laufwerkwechsel in der Regel im ausgeschalteten Zustand. Rebuilt (Rekonstruktion der verlorenen Daten) bei Systemstart.
- Hot-Fix: Reserveplatte, die bei Ausfall für die ausgefallene Platte in das RAID eingebunden wird.
- Hot-Swap. Häufigkeit Plattenaustausch und Rebuilt im Betrieb.

Neben HW-Ausfällen gibt es weitere Ursachen für den Verlust schwer wiederzubeschaffende Daten. Auftrittshäufigkeiten:

- 59% Hardwareprobleme
- 26% Anwenderfehler
- 9% Softwarefehler
- 2% Schadware (Vieren, ...)
- 2% Naturkatstrophen
- 2% sonstiges.

Selbst ausgefeilte RAIDs tolerieren nur HW-Probleme. Den einzig wirklich zuverlässigen Schutz gegen Datenverluste bieteten konsequent geplante und durchgeführte Backups.

Redundanz

Redundanz

(von lateinisch redundare, überlaufen, sich reichlich ergießen).

In der Technik sind Redundanzen zusätzliche Ressourcen, die im fehler- und störungsfreien Betrieb nicht benötigt werden:

- Motoren, Baugruppen, komplette Geräte, Steuerleitungen,
- Stromversorgung, Leistungsreserve, ...

zur Erhöhung der Verfügbarkeit, Zuverlässigkeit und/oder Sicherheit.

Arten von Redundanz:

- Standby-Redundanz: Service-Übernahme nach Ausfällen z.B. durch eine USV nach Stromausfall.
- räumlich verteilte Redundanz: Service-Übernahme nach örtlich begrenzten Zerstörungen z.B. eines zerstörten Rechenzentrums.
- diversitäte Redundanz: Service-Übernahme bei Fehlfunktionen durch Fehler (siehe Abschn. 1.2 *MF-Behandlung*).

Bei Standby-Redundanz wird zwischen heißer und kalter Redundanz unterschieden (siehe Abschn. 3.6.3 *Reserveeinheiten*)

KooN (k out of n) Systeme

System aus n Komponenten (Rechnerknoten, Server, Verbindungen, ...), das insgesamt verfügbar ist, solange k Komponenten verfügbar sind. Die Anzahl der verfügbaren Komponenten ist binomial-verteilt:

$$\mathbb{P}(X = k) = \binom{n}{k} \cdot p^k \cdot (1-p)^{n-k} \tag{3.27}$$

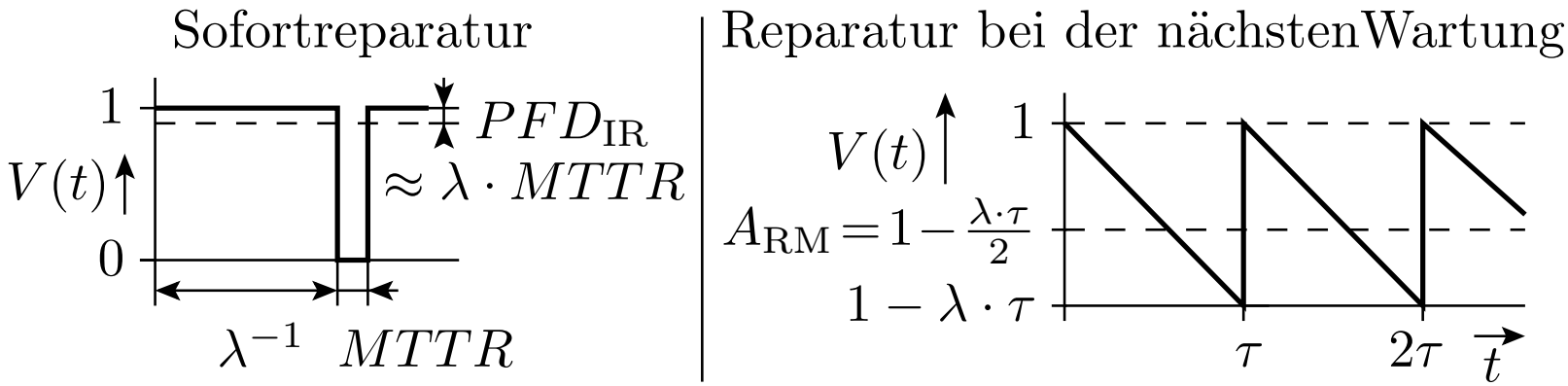
Der Ausfall auch von weniger als $n - k$ redundanten Komponenten kann einen Leistungsabfall:

- Aufgabenumverteilung auf andere Systembestandteile.
- Bei Notstromversorgung Abschalten von Systemteilen.
- Transmission Control Protocol (TCP): auch dann noch eine sichere Punkt-zu-Punkt-Verbindung, wenn einzelne Knoten im Netzwerk überlastet, falsch eingestellt sind oder Daten verfälschen.
- Bis zum Ausfall der letzten Einheit ist auch auch bei $k > 1$ oft noch ein Notbetrieb möglich.

n	Anzahl der existierenden Teilsysteme.
k	Anzahl der Teilsysteme, die von den n vorhandenen verfügbar sein müssen.
p	Wahrscheinlichkeit der Verfügbarkeit $V = 1 - PFD$ eines einzelnen Teilsystems.

Verfügbarkeit 1oo1 mit Wartung

Gewartetet Systeme ohne Redundanz (siehe Abschn. 3.6.4 *Wartung*).



Ohne NS durch nicht korrigierbare erkannte Fehlfunktionen:

$$PFD = \eta_{\mathrm{IR}} \cdot \lambda \cdot MTTR + (1 - \eta_{\mathrm{IR}}) \cdot \frac{\lambda \cdot \tau}{2} \tag{3.95}$$

V	Überlebenswahrscheinlichkeit.
η_{IR}	Anteil der Ausfälle, die durch Sofortreparatur beseitigt werden.
PFD	Wahrscheinlichkeit der Nichtverfügbarkeit (**p**robability of **f**ailure on **d**emand).
$MTTR$	mittlere Reparaturzeit (**m**ean **t**ime **t**o **r**epair).
τ	Wartungsintervall.
λ	Ausfallrate.
NS	keine Service-Leistung (no service).

Beispiel 4.7: PFD und Verfügbarkeit eines Autos

Fahrdauer innerhalb eines Wartungsintervalls $\frac{10.000\text{km}}{50\,\text{km/h}} = 200\text{h}$, Ausfallrate $\lambda = 10^{-3}\,\text{h}^{-1}$, Anteil der sofort bemerkten Ausfälle $\eta_\text{D} = 80\%$, mittlere Reparaturzeit $MTTR = 2\,\text{h}$.

PFD Wahrscheinlichkeit der Nichtverfügbarkeit (**p**robability of **f**ailure on **d**emand).

Wartungsintervall $\tau = 200\text{h}$, $\lambda = 10^{-3}\,\text{h}^{-1}$, $\eta_\text{D} = 80\%$, $MTTR = 2\,\text{h}$

a) Wahrscheinlichkeit der Nichtverfügbarkeit?

b) Verfügbarkeit?

Wartungsintervall $\tau = 200\text{h}$, $\lambda = 10^{-3}\,\text{h}^{-1}$, $\eta_\text{D} = 80\%$, $MTTR = 2\,\text{h}$

a) Wahrscheinlichkeit der Nichtverfügbarkeit?

b) Verfügbarkeit?

a)

$$PFD = \eta_\text{IR} \cdot \lambda \cdot MTTR + (1 - \eta_\text{IR}) \cdot \tfrac{\lambda \cdot \tau}{2} \tag{3.95}$$

$$PFD_\text{KFZ} = 0{,}8 \cdot 10^{-3}\,\text{h}^{-1} \cdot 2\,\text{h} + \frac{0{,}2 \cdot 10^{-3}\,\text{h}^{-1} \cdot 200\text{h}}{2} = 2{,}16\%$$

b)

$$A = 1 - PFD \tag{1.2}$$

$$V_\text{KFZ} = 97{,}84\%$$

Verfügbarkeit von 1oo2 Systemen

Wahrscheinlichkeit, dass mindestens eine von zwei unabhängig ausfallenden Komponenten mit Überlebenswahrscheinlichkeit

$$V\left(t\right) = \mathrm{e}^{-\lambda \cdot t} \tag{3.89}$$

überlebt:

$$V_{\mathrm{1oo2.IF}}\left(t\right) = 1 - \left(1 - e^{\lambda t}\right)^2 = 2 \cdot \mathrm{e}^{\lambda t} - \mathrm{e}^{2\lambda t}$$

Wenn Reparatur erst bei der Wartung, ist die Verfügbarkeit die mittlere Überlebenswahrscheinlichkeit in einem Wartungsintervall τ:A

$$\begin{aligned} A_{\mathrm{1oo2.IFRM}} &= \frac{1}{\tau} \int_0^\tau \left(2 \cdot \mathrm{e}^{\lambda t} - \mathrm{e}^{2\lambda t}\right) \cdot \mathrm{d}\tau \\ &= \frac{2}{\lambda\tau} \cdot \left(1 - \mathrm{e}^{\lambda\tau}\right) - \frac{1}{2\lambda\tau} \cdot \left(1 - \mathrm{e}^{2\lambda\tau}\right) \end{aligned}$$

1oo2	1 out of 2, Gesamtsystem verfügbar, wenn 1 von 2 Systemen verfügbar ist.
$V_{*.\mathrm{IF}}$	Überlebenswahrscheinlichkeit bei unabhängigen Ausfallursache.
$A_{*.\mathrm{IFRM}}$	Verfügbarkeit unabhängigen Ausfallursache und Reparatur erst bei nächster Wartung.

$$A_{1oo2.IFRM} = \frac{2}{\lambda\tau} \cdot \left(1 - e^{\lambda\tau}\right) - \frac{1}{2\lambda\tau} \cdot \left(1 - e^{2\lambda\tau}\right)$$

Mit der Näherung $e^{-x} = 1 - x + \frac{x^2}{2} - \frac{x^3}{6}$

$$\frac{2}{\lambda\tau} \cdot \left(1 - e^{\lambda\tau}\right) = \frac{2}{\lambda\tau} \cdot \left(1 - \left(1 - \lambda\tau + \frac{(\lambda\tau)^2}{2} - \frac{(\lambda\tau)^3}{6}\right)\right) = 2 - \lambda\tau + \frac{(\lambda\tau)^2}{3}$$

$$\frac{1}{2\lambda\tau} \cdot \left(1 - e^{2\lambda\tau}\right) = \frac{1}{2\lambda\tau} \cdot \left(1 - \left(1 - 2\lambda\tau + \frac{(2\lambda\tau)^2}{2} - \frac{(2\lambda\tau)^3}{6}\right)\right) = 1 - \lambda\tau + \frac{2 \cdot (\lambda\tau)^2}{3}$$

$$A_{1oo2.IFRM} = 2 - \lambda\tau + \frac{(\lambda\tau)^2}{3} - 1 - \lambda\tau + \frac{2 \cdot (\lambda\tau)^2}{3} = 1 - \frac{(\lambda\tau)^2}{3}$$

Wenn beide Komponenten unabhängig voneinander ausfallen, nimmt die PFD mit dem Quadrat von $\lambda\tau$ ab:

$$PFD_{1oo2.IFRN} = 1 - A_{1oo2.IFRM} = \frac{(\lambda\tau)^2}{3}$$

1oo2	1 out of 2, Gesamtsystem verfügbar, wenn 1 von 2 Systemen verfügbar ist.
A	Verfügbarkeit (**a**vailability).
PFD	Wahrscheinlichkeit der Nichtverfügbarkeit (**p**robability of **f**ailure on **d**emand).
λ	Ausfallrate.
τ	Wartungsintervall.
$*_{*.IFRM}$	unabhängigen Ausfallursache und Reparatur erst bei nächster Wartung.

Gleiche Ausfallursache und Sofortreparatur

Für zwei unabhängig ausfallende Systeme und Sofortreparatur ist die Wahrscheinlichkeit, dass nicht beide zufällig gleichzeitg ausgefallen sind, das Quadrat der Wahrscheinlichkeiten, dass nach Gl. (Gl. 3.94) eines ausgefallen ist:

$$PFD_{1oo2.\mathrm{IFIR}} = (\lambda \cdot MTTR)^2$$

Wenn beide Komponenten zeitgleich durch dieselbe Ursache ausfallen (CC-Ausfälle), ist die PFD die eines Einzelsystems.

- Reparatur erst bei der nächten Wartung (Gl. 3.93):

$$PFD_{1oo2.\mathrm{CCRM}} = \frac{\lambda\tau}{2}$$

- Sofortreparatur (Gl. 3.94):

$$PFD_{1oo2.\mathrm{CCIR}} = \lambda \cdot MTTR$$

PFD	Wahrscheinlichkeit der Nichtverfügbarkeit (**p**robability of **f**ailure on **d**emand).
$*_{*.\mathrm{CCRM}}$	gleiche Ausfallursache und Reparatur erst bei nächster Wartung.
$*_{*.\mathrm{IFIR}}$	unabhängigen Ausfallursache und Sofortreparatur.
$*_{*.\mathrm{CCIR}}$	gleiche Ausfallursache und Sofortreparatur.

Gesamte PFD als gewichteter Mittelwert:

$$\begin{aligned} PFD_{1oo2} &= \eta_{CC} \cdot (\eta_{IR} \cdot PFD_{1oo2.CCIR} + (1-\eta_{IR}) \cdot PFD_{1oo2.CCRM}) \\ &+ (1-\eta_{CC}) \cdot (\eta_{IR} \cdot PFD_{1oo2.IFIR} + (1-\eta_{IR}) \cdot PFD_{1oo2.IFRM}) \\ &= \eta_{CC} \cdot \left(\eta_{IR} \cdot \lambda \cdot MTTR + (1-\eta_{IR}) \cdot \frac{\lambda\tau}{2}\right) \\ &+ (1-\eta_{CC}) \cdot \left(\eta_{IR} \cdot (\lambda \cdot MTTR)^2 + (1-\eta_{IR}) \cdot \frac{(\lambda\tau)^2}{3}\right) \end{aligned}$$

Im Normalfall $MTTR \ll \tau$ (Reparaturzeit viel kürzer als Wartungsintervall):

$$\begin{aligned} PFD_{1oo2} &= (1-\eta_{IR}) \cdot \frac{\lambda\tau}{2} + (1-\eta_{CC}) \cdot (1-\eta_{IR}) \cdot \frac{(\lambda\tau)^2}{3} \\ &= (1-\eta_{IR}) \cdot \left(\frac{\eta_{CC} \cdot \lambda\tau}{2} + \frac{(1-\eta_{CC}) \cdot (\lambda\tau)^2}{3}\right) \end{aligned} \tag{17}$$

1oo2-Redundanz verringert bei einem hohen Anteil von CC-Ausfällen $\eta_{CC} \gg \lambda\tau$ die PFD auf den Anteil der CC-Ausfälle und für seltene CC-Ausfälle $\eta_{CC} \ll \lambda\tau$ nimmt die PFD mit dem Quadrat der PFD eines Einzelsystems ab.

η_{CC}	Anteil der gleichzeitigen Ausfälle durch dieselbe Ursache.
η_{IR}	Anteil der Ausfälle, die durch Sofortreparatur beseitigt werden.
λ	Ausfallrate.
τ	Wartungsintervall

Weitere k aus n Systeme

Systemen mit größerem n oder k verhalten sich bei CC-Ausfällen auch wie ein Einzelsystem, d.h. die n Einzelsysteme fallen gemeinsam aus:

$$PFD_{koon.\mathrm{CCRM}} = PFD_{1oo2.\mathrm{CCRM}} = \frac{\lambda\tau}{2}$$

Für unabhängige Ausfälle und Reparatur erst bei der nächsten Wartung beträgt die PFD nach [6]:

k	1	2	3	4
$PFD_{koo2.\mathrm{IFRM}}$	$\frac{(\lambda\cdot\tau)^2}{3}$	$\lambda\cdot\tau$	-	-
\EqPFDeooz$PFD_{koo3.\mathrm{IFRM}}$	$\frac{(\lambda\cdot\tau)^3}{4}$	$(\lambda\cdot\tau)^2$	$\frac{3\cdot\lambda\cdot\tau}{2}$	-
$PFD_{koo4.\mathrm{IFRM}}$	$\frac{(\lambda\cdot\tau)^4}{5}$	$(\lambda\cdot\tau)^3$	$2\cdot(\lambda\cdot\tau)^2$	$2\cdot\lambda\cdot\tau$

In Anlehnung an Gl. 4.17 ergibt sich als Gesamt-PFD für den Normalfall $MTTR \ll \tau$ (Reparaturzeit viel kürzer als Wartungsintervall):

$$PFD_{koon} = (1-\eta_{\mathrm{IR}})\cdot\left(\frac{\eta_{\mathrm{CC}}\cdot\lambda\tau}{2} + (1-\eta_{\mathrm{CC}})\cdot PFD_{koon.\mathrm{IFRM}}\right) \qquad (18)$$

k	1	2	3	4
$PFD_{koo2.\mathrm{IFRM}}$	$\frac{(\lambda\cdot\tau)^2}{3}$	$\lambda\cdot\tau$	-	-
$PFD_{koo3.\mathrm{IFRM}}$	$\frac{(\lambda\cdot\tau)^3}{4}$	$(\lambda\cdot\tau)^2$	$\frac{3\cdot\lambda\cdot\tau}{2}$	-
$PFD_{koo4.\mathrm{IFRM}}$	$\frac{(\lambda\cdot\tau)^4}{5}$	$(\lambda\cdot\tau)^3$	$2\cdot(\lambda\cdot\tau)^2$	$2\cdot\lambda\cdot\tau$

$$PFD_{koon} = (1-\eta_{\mathrm{IR}})\cdot\left(\frac{\eta_{\mathrm{CC}}\cdot\lambda\tau}{2} + (1-\eta_{\mathrm{CC}})\cdot PFD_{koon.\mathrm{IFRM}}\right) \quad (4.18)$$

Anmerkungen:

- Hohe Verfügbarkeit mit und ohne Redundanzen erfordert geringe Ausfallraten und angepasste Wartungsintervalle.
- Einen wirksame Verfügbarkeitserhöhung durch redundante Einheiten verlangt, dass CC-Ausfälle sehr unwahrscheinlich sind: kalte Reserve, räumliche Trennung, ...

PFD	Wahrscheinlichkeit der Nichtverfügbarkeit (**p**robability of **f**ailure on **d**emand).
$koon$	System aus n Komponenten, verfÃŒgbar, solange k Komponenten verfÃŒgbar sind.
η_{CC}	Anteil der gleichzeitigen Ausfälle durch dieselbe Ursache.
η_{IR}	Anteil der Ausfälle, die durch Sofortreparatur beseitigt werden.
λ	Ausfallrate.
τ	Wartungsintervall.
CC	gemeinsame Ursache (common cause).

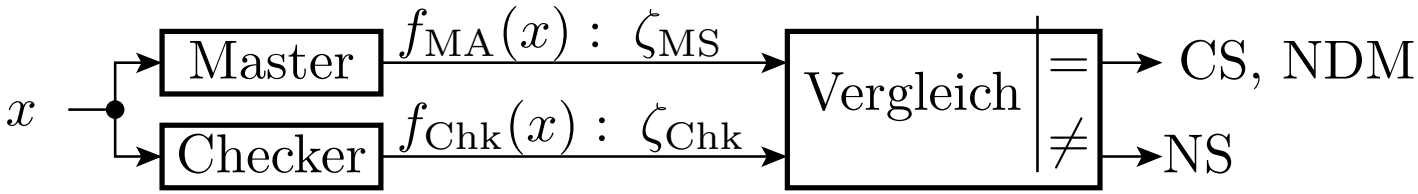
Nutzung von Redundanzen in der Praxis

- Regelmäßig wartbare Systeme ohne extreme Anforderungen an die Verfügbarkeit und Sicherheit sind in der Regel 1oo1 (keine Redundanz).
- Im Maschinenbau je nach Sicherheitsstufe: 1oo1 oder 1oo2.
- Nur Systeme ohne einen in kurzer Zeit erreichbaren sicheren Zustand, z. B.: Flugzeugsteuerungen, Atomkraftwerke, Chemiereaktoren, auch 2oo2, 2oo3 oder 2oo4.
- Große KooN-Redundanzen: fehlertolerante Server-Cluster, Verbindungsnetzwerke, ..., also Systeme, die ohnehin aus vielen gleichen Teilsystemen bestehen mit hohen Anforderungen an die Verfügbarkeit.

Systemlösungen

Master-Checker-System mit Notbetrieb

Zweifachberechnung und nur Nutzung bei Übereinstimmung:

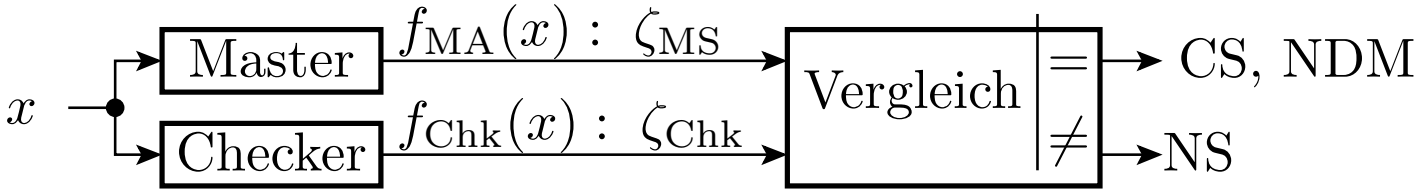


Verfügbarkeit und Zuverlässigkeit ohne Berücksichtigung von Ausfällen:
Der Anteil der DS verringert sich um $(1-\eta_{\mathrm{DS}})\cdot\zeta$ erkannte Master- $\eta_{\mathrm{DS}}\cdot\zeta$
Checker-MF und der Anteil der MF um $(1-\eta_{\mathrm{DS}})$:

$$A_{\mathrm{MCS}} = 1 - \zeta \cdot \frac{MTTR}{MTS} \tag{1.27}$$

$$R_{\mathrm{MCS}} = \frac{R-1}{(1-\eta_{\mathrm{Div}})} \tag{1.28}$$

A_{MCS}	Verfügbarkeit von **M**aster-**C**hecker-**S**ystemen ohne Korrektur und Ausfälle.
R_{MCS}	Zuverlässigkeit von **M**aster-**C**hecker-**S**ystemen ohne Korrektur.
ζ_{MA}	Fehlfunktionsrate **Ma**ster.
ζ_{Chk}	MF-Rate **Ch**ecke**r**.
η_{Div}	**Div**ersitätsrate, Anteil der MFs ohne gemeinsame Ursache.
$MTTR$	mittlere Reparaturzeit (**m**ean **t**ime **t**o **r**epair).
MTS	mittlere Service-Dauer (**m**ean **t**ime to **s**ervice).



Zweifachberechnung und Vergleich ist ein 2oo2-System mit eine ausfallbezogenen Verfügbarkeit nach Gl. 4.18 von:

$$A_{2\mathrm{oo}2} = 1 - \eta_{\mathrm{CC}} \cdot (1 - \eta_{\mathrm{IR}}) \cdot \frac{\lambda\tau}{2} + (1 - \eta_{\mathrm{CC}}) \cdot (1 - \eta_{\mathrm{IR}}) \cdot \lambda \cdot \tau$$
$$= 1 - (1 - \eta_{\mathrm{IR}}) \cdot \left(1 - \tfrac{\eta_{\mathrm{CC}}}{2}\right) \cdot \lambda \cdot \tau$$

Insgesamt verfügbar als 2oo2 **UND** als MCS:

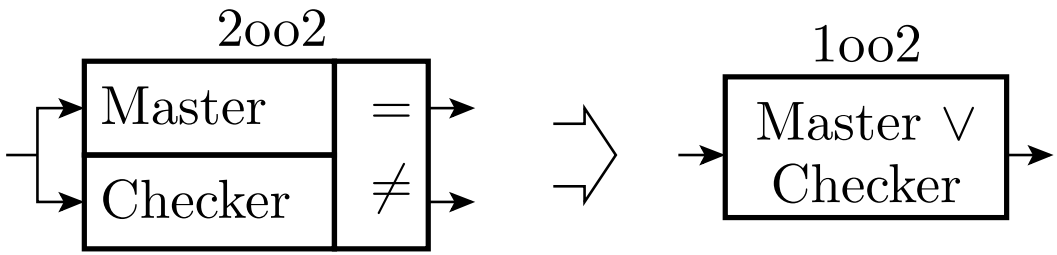
$$A_{2\mathrm{oo}2.\mathrm{MCS}} = A_{2\mathrm{oo}2} \cdot A_{\mathrm{MCS}}$$

Mit beiden Verfügbarkeiten nahe 100%:

$$A_{2\mathrm{oo}2.\mathrm{MCS}} = 1 - (1 - \eta_{\mathrm{IR}}) \cdot \left(1 - \tfrac{\eta_{\mathrm{CC}}}{2}\right) \cdot \lambda \cdot \tau - \zeta \cdot \frac{MTTR}{MTS} \qquad (19)$$

$A_{2\mathrm{oo}2}$	Verfügbarkeit von 2 aus 2 Komponenten.
η_{CC}	Anteil der gleichzeitigen Ausfälle durch dieselbe Ursache.
η_{IR}	Anteil der Ausfälle, die durch Sofortreparatur beseitigt werden.
λ	Ausfallrate.
τ	Wartungsintervall.
A_{MCS}	Verfügbarkeit von **M**aster-**C**hecker-**S**ystemen ohne Korrektur und Ausfälle.

Verfügbarkeitserhöhung durch Notbetrieb



Ohne Vergleich und Aussortieren abweichender Ergebnisse ist ein Master-Checker ein 1oo2-System mit der ausfallbezogenen deutlich höheren Verfügbarkeit:

$$PFD_{1oo2} == (1 - \eta_{IR}) \cdot \left(\frac{\eta_{CC} \cdot \lambda\tau}{2} + \frac{(1-\eta_{CC}) \cdot (\lambda\tau)^2}{3} \right) \tag{4.17}$$

Die Zuverlässigkeit im Notbetrieb ist nur R statt $\approx R((1 - \eta_{Div})$ und im Mittel etwas kleiner, wenn das System in seltenen Fällen Master-Checker Überwachung verfügbar ist:

$$R_{MCN} = \frac{R-1}{(1-\eta_{Div})} \cdot \frac{A_{2oo2}}{A_{1oo2}} + R \cdot \frac{A_{1oo2}-A_{2oo2}}{A_{1oo2}} \tag{20}$$

Mehrfachberechung mit Mehrheitsentscheid

Mindests 3 identische System führen dieselbe Berechnung aus und geben das Mehrheitsergebnis weiter. Verfügbarkeit und Zuverlässigkeit ohne Berücksichtigung von Ausfällen:

$$A_{\mathrm{MV3}} = 1 - \zeta \cdot (1 - \eta_{\mathrm{Div}}) \cdot \frac{MTTR}{MTS} \tag{1.31}$$

$$R_{\mathrm{MV3}} = \frac{R}{(1-\eta_{\mathrm{Div}})} - 1 \tag{1.32}$$

Nach einem Ausfall reduziert sich die Zuverlässigkeit bis zum Abschluss der Reparatur auf die eines Master-Checker-Systems

$$R_{\mathrm{MCS}} = \frac{R-1}{(1-\eta_{\mathrm{Div}})} \tag{1.28}$$

und wenn vor Abschluss der Reparatur ein zweites System ausfällt, auf die Grundzuverlässigkeit R.

R_{MV3} Zuverlässigkeit der Ergebnisse eines 3-Versionen-Mehrheitsentscheids.

Ausfallbezogene Verfügbarkeit

Wenn nur das Ausfallrisiko betrachtet wird, ist das System verfügbar:

- mit 3-Versionen Mehheitsentscheid:

$$A_{3oo3} = 1 - \eta_{CC} \cdot (1 - \eta_{IR}) \cdot \frac{\lambda\tau}{2} + (1 - \eta_{CC}) \cdot (1 - \eta_{IR}) \cdot \frac{3 \cdot \lambda \cdot \tau}{2}$$

- mindestens mit Master-Checker-Überwachung:

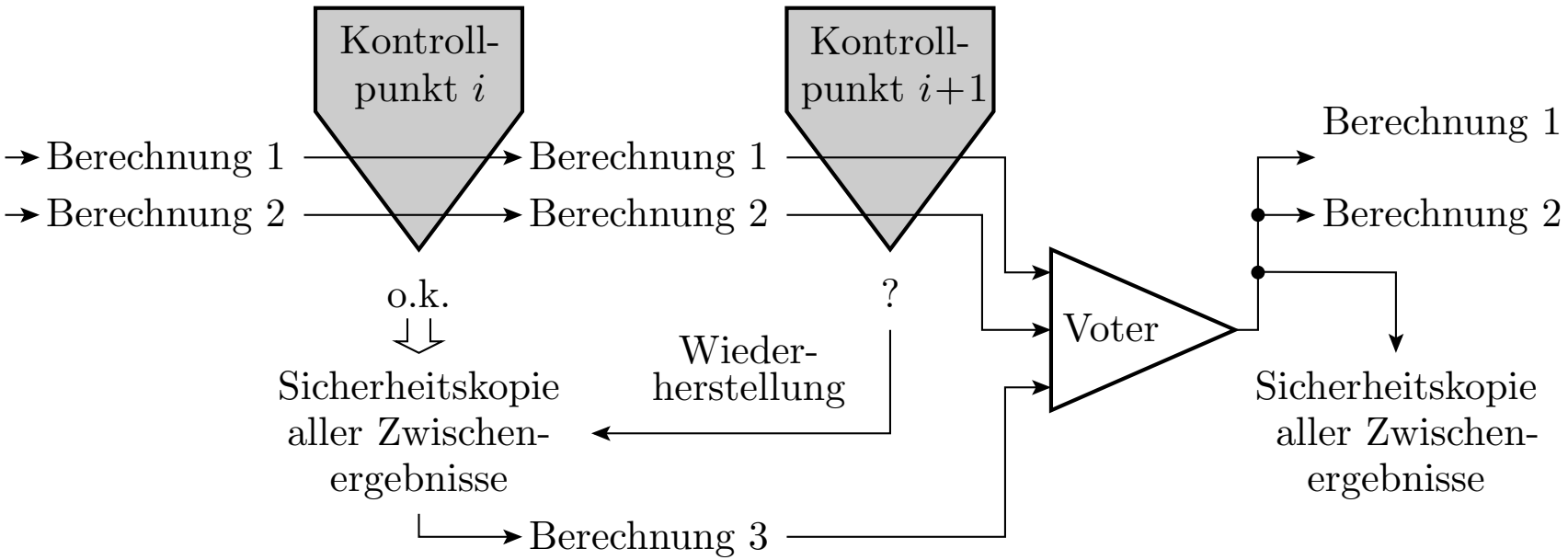
$$A_{2oo3} = \left(1 - \eta_{CC} \cdot (1 - \eta_{IR}) \cdot \frac{\lambda\tau}{2} + (1 - \eta_{CC}) \cdot (1 - \eta_{IR}) \cdot (\lambda \cdot \tau)^2\right) \cdot \eta_{DS}$$

- mindestens als Einzelsystem:

$$A_{1oo3} = 1 - \eta_{CC} \cdot (1 - \eta_{IR}) \cdot \frac{\lambda\tau}{2} + (1 - \eta_{CC}) \cdot (1 - \eta_{IR}) \cdot \frac{(\lambda \cdot \tau)^3}{4}$$

Ein 2oo3-System mit Mehrfachberechung und Vergleich wurde bereits 1956 von John von Neumann für die damaligen Röhrenrechner vorgeschlagen, in denen alle paar Stunden eine Röhre ausgefallen ist.

Check-Point-Roll-Back-Recovery [5]



- Zwei parallele möglichst unabhängig verfügbare Berechnungen.
- An einprogrammierten Kontrollpunkten im Programm werden die Bearbeitungszustände (Variablen, Register, ...) vergleichen.
- Bei Übereinstimmung Speicherung des Bearbeitungszustands in einem geschützten Speicher.
- Bei Abweichung, Laden der letzten Sicherheitskopie und Berechnungswiederholung (Roll-Back Recovery).

- Nach Roll-Back Recovery am nächsten Kontrollpunkt wieder Vergleich.
- Wenn Übereinstimmung, diesen als gesicherten Zustand speichern, sonst Abbruch.

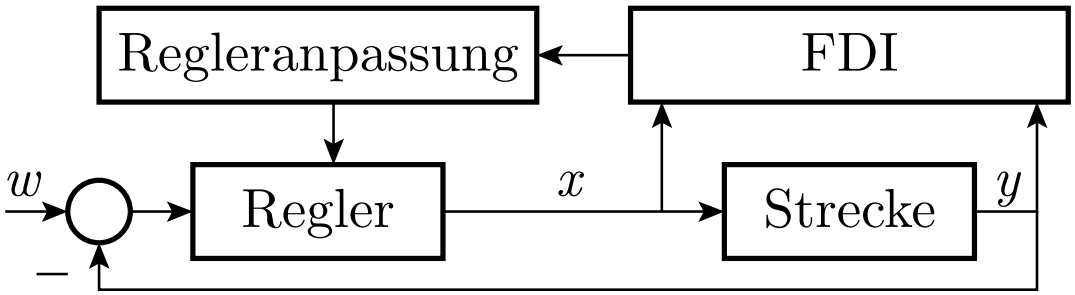
Vergleich mit 3-Versionen Mehrheitsentscheid:

- nur 2 Systeme und für die meisten Service-Leistungen nur zwei Berechnungen erforderlich.
- ausfallbezogenen Teilverfügbarkeit: Mehheitsentscheid 2oo2, mindestens ein Einzelsyystem 1oo2.

Beispiel: Sequoia-System [1]:

- Berechnung auf zwei Prozessoren mit eigenem Write-Back-Cache.
- Vergleich in jedem Takt.
- Zustands-Backup bei Ereignissen wie Stack-Überlauf und Prozesswechsel.
- Hauptspeicher hat die Funktion des stabilen Speichers.

Fehlertolerantes Regelungssystem



In einem Reglersystem wird vom Sollwert w der zu regelnde Ist-Wert y abgezogen. Aus der Differenz bildet der Regler den Stellwert x für die Regelstrecke (z.B. eine Heizung, wenn y eine Temperatur ist).

Hinzufügen einer zusätzlichen Überwachungs- und Fehlerbehandlungsschicht (FDI) mit den Aufgaben:

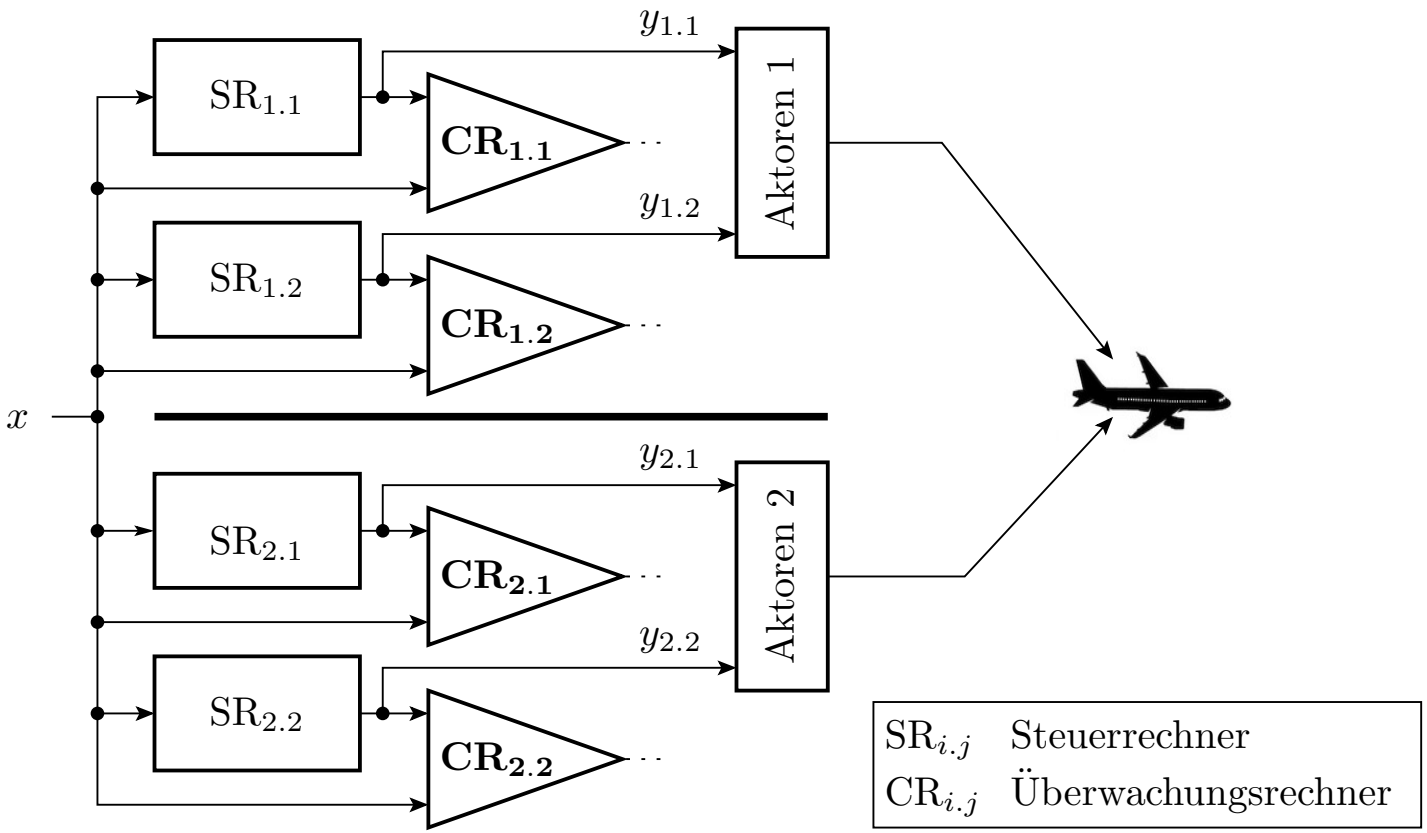
- Fehlerdiagnose (Abschätzung von Fehlerursache und -ort) und
- Anpassung der Regelung an den aktuellen Fehlerzustand so, dass eine Mindestfunktionalität gewährleistet bleibt.

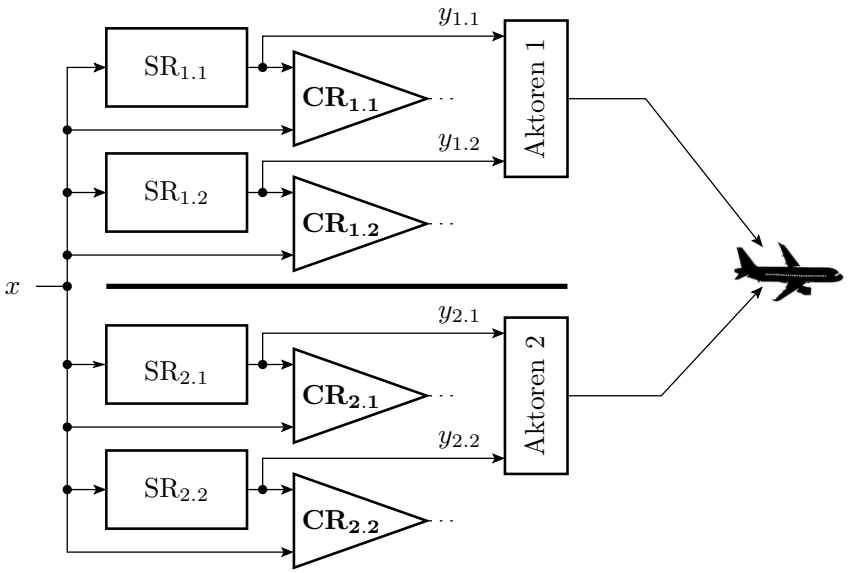
Zur Erzielung von Fehlertoleranz gegenüber Fehlfunktionen von Regler und Strecke durch Störungen, Fehler und Ausfälle.

FDI **F**ehler **D**etektion, Isolation und **I**dentifikation.

Flugsteuersystem Airbus A3XX [8]

Hochsicherheitskritische Anwendungen müssen möglichst alle Fehlfunktionen, auch solche durch nicht erkannte Entwurfsfehler, nicht erkannte Fertigungsfehler und Ausfälle tolerieren.





- Zwei identische Systeme mit allen Sensoren, Aktoren und zwei Rechnerpaaren.
- Jedes Rechnerpaar besteht aus einem Steuerrechner $SR_{i.j}$, der die Aktoren ansteuert, und einem Überwachungsrechner $CR_{i.j}$.
- Normalzustand Rechner $SR_{1.1}$ steuert und $CR_{1.1}$ überwacht. Zweites Rechnerpaar Stand-By. System 2 abgeschaltet.
- Bei Ausfall übernimmt Rechnerpaar 1 von Rechnerpaar 2. Bei Komplett-, Sensor- oder Aktorausfällen übernimmt System 2 von System 1.

Diversität: Rechner unterschiedlicher Hersteller, getrennte Software-Entwicklung nach Spezifikationen, die unabhängig von einer gemeinsamen Basisspezifikation abgeleitet wurden.

Zusammenfassung

Fehlerkorrigierende Codes

Erweiterung der Menge der darstellbaren Codeworte um eine viel größere Menge korrigierbarer Codeworte und optional um unzulässige nicht korrigierbare Codeworte. Mindestbitanzahl:

$$2^{\#\mathrm{Bit}} \geq \#VCW + \#VCW \cdot \#CVC \tag{4.14}$$

Hamming-Distanz:

- zum Erkennen aller d-Bit Verfälschungen:

$$Ham \geq d + 1 \tag{4.15}$$

- zur Korrektor aller c-Bit Verfälschungen:

$$Ham \geq 2 \cdot c + 1 \tag{4.16}$$

Hamming-Codes

- Paritätsbit: modulo-2 Summe (EXOR-Verknüpfung) aller Datenbits. Nachweis aller ungeradzahligen Verfälschungen.
- Kreuzparität: Zeilen- und Spaltenparität für ein 2D-Datenfeld. Korrektur bis zu einem verfälschten Bit.
- 1-Bit fehlerkorrigierender Hamming-Code. Rechenweg zur Festlegung der EXOR-Summen für die Kontrollbits so, dass die Kontrollbitdifferenz die binärcodirte Nummer des verfälschten Bits ist.
- Konstruktion von Codes für die Burst-Fehler Korrektur durch Verschränkung von Code-Worten für die Korrektur von Einzelbitverfälschungen.

RAID und Backup

Ein RAID ist ein logischen Laufwerk aus mehreren physischen Festplatten oder SSD, optional mit redundanter Datenspeicherung:

- RAID0: redundanzfreie Speicherung auf mehrere Festplatten.
- RAID1: paarweise gespiegelte Platten oder Plattengruppen. Tolerierung von mindestens einem Plattenausfall.
- RAID mit Paritätsblöcken: bei n Platten speichern auf $n-1$ Platten Datenblöcke und auf eine ein Paritätsblock. Erlaubt nach einem Plattenausfall die Rekonstruktion aller Daten aus denen der noch verfügbaren Platten.
- Erweiteret Fehlertoleranz: Techniken für Bitfehlerkorrektur , ...
- Ho-Fix: Reserveplatte, die sofort nach Plattenausfall deren Funktion übernimmt.
- Rebuilt: Rekonstruktion der verlorenen Daten für die Ersatzplatte.

Selbst ausgefeilte RAIDs tolerieren nur HW-Probleme. Wirklich zuverlässigen Schutz gegen Datenverluste bieteten Backups.

Redundanzen, KooN-Systeme

Wahrscheinlichkeit der Nicht-Verfügbarkeit:

- 1oo2-System:

$$PFD_{1oo2} == (1 - \eta_{IR}) \cdot \left(\frac{\eta_{CC} \cdot \lambda\tau}{2} + \frac{(1-\eta_{CC}) \cdot (\lambda\tau)^2}{3} \right) \quad (4.17)$$

- Weitere KooN-Systeme:

k	1	2	3	4
$PFD_{koo2.IFRM}$	$\frac{(\lambda \cdot \tau)^2}{3}$	$\lambda \cdot \tau$	-	-
$PFD_{koo3.IFRM}$	$\frac{(\lambda \cdot \tau)^3}{4}$	$(\lambda \cdot \tau)^2$	$\frac{3 \cdot \lambda \cdot \tau}{2}$	-
$PFD_{koo4.IFRM}$	$\frac{(\lambda \cdot \tau)^4}{5}$	$(\lambda \cdot \tau)^3$	$2 \cdot (\lambda \cdot \tau)^2$	$2 \cdot \lambda \cdot \tau$

$$PFD_{koon} = (1 - \eta_{IR}) \cdot \left(\frac{\eta_{CC} \cdot \lambda\tau}{2} + (1 - \eta_{CC}) \cdot PFD_{koon.IFRM} \right) \quad (4.18)$$

Praktische Nutzung:

- Maschinen bis 1oo2
- Flugzeugsteuerungen, Atomkraftwerke, Chemiereaktoren, .. auch 2oo2 bis 2oo4.

Systemlösungen

Master-Checker-System mit Notbetrieb:

- Verfügbarkeit als Master-Checker-System:

$$A_{2oo2.\mathrm{MCS}} = 1 - (1 - \eta_{\mathrm{IR}}) \cdot \left(1 - \frac{\eta_{\mathrm{CC}}}{2}\right) \cdot \lambda \cdot \tau - \zeta \cdot \frac{MTTR}{MTS} \quad (4.19)$$

- Verfügbarkeit einschließlich Notbetrieb ohne Checker und Vergleich:

$$PFD_{1oo2} == (1 - \eta_{\mathrm{IR}}) \cdot \left(\frac{\eta_{\mathrm{CC}} \cdot \lambda\tau}{2} + \frac{(1-\eta_{\mathrm{CC}}) \cdot (\lambda\tau)^2}{3}\right) \quad (4.17)$$

- Zuverlässigkeit mit Umschaltung in den Notbetrieb:

$$R_{\mathrm{MCN}} = \frac{R-1}{(1-\eta_{\mathrm{Div}})} \cdot \frac{A_{2oo2}}{A_{1oo2}} + R \cdot \frac{A_{1oo2} - A_{2oo2}}{A_{1oo2}} \quad (4.20)$$

2-Versionen Mehrheitsentscheid mit den Notbetriebsarten:

- 2oo2 Master-Checker, wenn ein System ausgefallen und noch nicht repariert ist, und
- 1oo1 Einzelsystem, solange zwie Systeme ausgefallen und noch nicht repariert sind.

fast 1oo3 Verfügbarkeit und Zuverlässigkeiterhöhung auf 2 aus 3 Mehrheitsentscheid.

Check-Point-Roll-Back-Recovery: Master-Checker-Paar mit Drittberechnung und Mehrheitsentscheid nach Vergleichsfehlern. Spart das dritte System, braucht aber ein Backup für die Zwischenergebnisse an den Kontrollpunkten für die möglicherweise erforderliche dritte Berechnung.

Fehlertolerantes Regelungssystem: zusätzlichen Überwachungs- und Fehlerbehandlungsschicht (FDI) zur Anpassung der Regelung an den aktuellen Fehlerzustand. Die den Fehlerzuständen zugeordneten Regler bilden eine funktionale Redundanz.

Flugsteuersystem Airbus: Beispiel für ein komplexes System mit

- Paaren aus Master-Rechnen und Kontrollrechnern zur Überwachung und MF-Behandlung,
- diversitären Redundanzen für Fehlertoleranz gegenüber SW-Fehlern und
- Komplett-Redundanz incl. aller Sensoren und Aktoren zur Minimierung der Risiken gegenüber allen denkbaren CC-Ausfällen.

Literatur

4. Literatur

[1] P.A. Bernstein.
Sequoia: a fault-tolerant tightly coupled multiprocessor for transaction processing.
Computer, 21(2):37–45, 1988.

[2] Nader B. Ebrahimi.
On the statistical analysis of the number of errors remaining in a software design document after inspection.
IEEE Transactions on Software Engineering, 23(8):529–532, 1997.

[3] Peter Liggesmeyer.
Software-Qualität: Testen, Analysieren und Verifizieren von Software.
Spectrum, 2002.

[4] Frank Padberg, Thomas Ragg, and Ralf Schoknecht.
Using machine learning for estimating the defect content after an inspection.
IEEE Transactions on Software Engineering, 30(1):17–28, 2004.

[5] D. K. Pradhan, D. D. Sharma, and N. H Vaidya.
Roll-forward checkpointing schemes.
In *Lecture Notes in Computer Science 744*, pages 93–116. Springer Verlag, 1994.

[6] Marvin Rausand and Arnljot Hsyland.
Systems Reliability Theory, Models, Statistical Methods, and Applications.
Wiley-Interscience, 2004.

[7] Qinbao Song, Martin Shepperd, Michelle Cartwright, and Carolyn Mair.
Software defect association mining and defect correction effort prediction.
IEEE Transactions on Software Engineering, 32(2):69–82, 2006.

[8] Pascal Traverse.
Dependability of digital computers on board airplanes.
Dependable Computing for critical applications, 4:134–152, 1991.

(siehe Folie 4.14 *Boundary-Scan*) 41
(siehe Folie 4.90 *Polynom-Multiplikation und -division*) 94
(siehe Folie 4.94 *Rückgewinnung durch Polynomdivision*) 98