Formelsammlung

Wahrscheinlichkeit und Information

Ein Ereignis „x“ trete mit der Wahrscheinlichkeit p(x) auf, dann ist das Auftreten dieses Ereignisses verbunden mit der Information I(x):

$$I(x) = \log_2\left(\frac{1}{p(x)}\right)$$

mit $\log_2(z) = \mathrm{ld}(z)$ „logarithmus dualis“

Entscheidungsgehalt

Liefert eine Informationsquelle n unterschiedliche Ereignisse x_i i = 1,n (die nicht mit gleicher Wahrscheinlichkeit auftreten müssen) so wird mit

$$H_0(n) = \mathrm{ld}(n) \ \ [\mathrm{bit}]$$

der Entscheidungsgehalt H_0 der Informationsquelle berechnet.

Entropie und Redundanz

Da die einzelnen Ereignisse x_i i = 1,n u.U. nicht mit gleicher Wahrscheinlichkeit auftreten ist auch ein Maß für den mittleren Informationsgehalt einer Quelle notwendig:

$$H = \sum_{i=1}^{n} p_i \log_2\left(\frac{1}{p_i}\right)$$

Der mittlere Informationsgehalt der Quelle wird als **Entropie** der Quelle bezeichnet.

Der Unterschied zwischen dem Entscheidungsgehalt und der Entropie einer Quelle die n Ereignisse x_i liefert wird mit der **Redundanz R** charakterisiert:

$$R = |\, H(n) - H_0(n) \,| \quad [\mathrm{bit}]$$

Codes

Ein **Code** ist eine **Abbildungsvorschrift**, die jedem Zeichen eines Zeichenvorrats (Urbildmenge) eindeutig ein Zeichen oder eine Zeichenfolge aus einem möglicherweise anderen Zeichenvorrat (Bildmenge) zuordnet.

Ein **Binärcode** ist ein Code, in dem Informationen durch Sequenzen von **zwei** verschiedenen Symbolen (zum Beispiel 1/0 oder wahr/falsch) dargestellt werden.

Man unterscheidet **numerische** Codes (Binärcode, 8421-(BCD-)Code, Excess-3-Code, Stibitz-Code, Aiken-Code, m-aus-n-Code etc.) und **alfanumerische** Codes (ASCII, EBCDIC, Unicode etc.). Spezielle numerische Codes sind **einschrittig** (z. B. Gray-Code).

Es existieren Codes mit **fester Wortlänge** (Block-Codes) und Codes mit **variabler Wortlänge**, z. B. der Morsecode.

Coderedundanz

Für Blockcodes gilt:

absolute Redundanz: $R = ld\ M - ld\ N = m - ld\ N$

mit: M Anzahl der möglichen Code-Wörter
N Anzahl der verwendeten Code-Wörter
m Wortbreite beim Binärcode

relative Redundanz: $r = (ld\ M - ld\ N) / ld\ M = (m - ld\ N) / m$

Für Codes mit variabler Codewortlänge gilt:

absolute Redundanz: $R = m - H$

mit: m mittlere Codewortlänge
H Entropie

relative Redundanz: $r = (m - H) / m$

Code-Sicherung

Oft wählt man absichtlich eine **redundante** Codierung, so dass sich die Codewörter zweier Zeichen (Nutzwörter) durch möglichst viele binäre Stellen von allen anderen Nutzwörtern unterscheiden. Zwischen den Nutzwörtern sind also eine Anzahl von Wörtern eingeschoben, die kein Zeichen repräsentieren und demnach nur infolge einer Störung entstehen können.

Stellendistanz: Anzahl der Stellen in denen sich zwei Codewörter voneinander unterscheiden.
Hamming-Distanz: Minimum aller Stellendistanzen zwischen Wörtern des Codes.

Für unterschiedlich lange Code-Wörter ist die Stellendistanz nicht definiert.

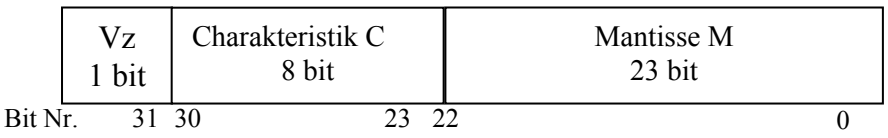
Bedingungen für die Erkennbarkeit und Korrigierbarkeit von Fehlern:

- Es können maximal **h-1** Fehler in einem Wort erkannt werden.
- Es können maximal **(h-1)/2** Fehler in einem Wort korrigiert werden.

Eine häufig verwendete Möglichkeit der Fehlererkennung ist das Hinzufügen eines **Paritätsbits** zu jedem Code-Wort, welches die Anzahl der Einsen (oder Nullen) auf eine **gerade** Anzahl (gerade Parität, even Parity) oder **ungerade** Anzahl (ungerade Parität, odd Parity) ergänzen. Ein-Bit-Fehler in einem Code-Wort können damit erkannt werden.

Gleitpunktformat

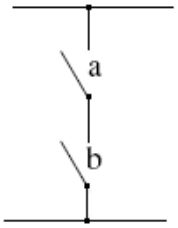
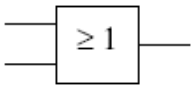
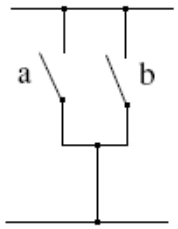
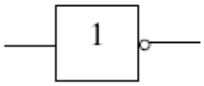
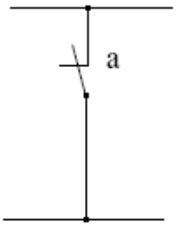
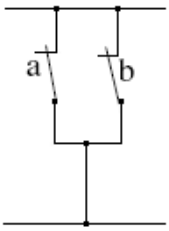
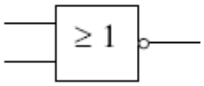
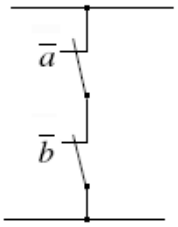
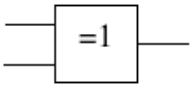
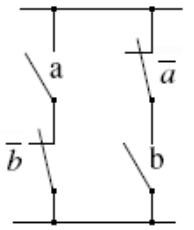
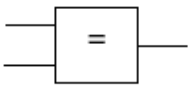
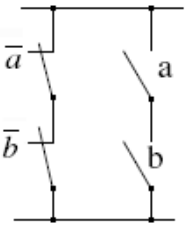
Charakteristik C	Mantisse M	Zahlenwert
000...000	beliebig	$(-1)^{Vz} \cdot 0{,}M \cdot 2^{-126}$
$1 \leq C \leq 2^n - 1$	beliebig	$(-1)^{Vz} \cdot 1{,}M \cdot 2^{C-127}$
111...111	= 0	$(-1)^{Vz} \cdot \infty$
111...111	≠ 0	not a number



Logiktheoreme

1.	$x \vee 0 = x$	$x \wedge 1 = x$
2.	$x \vee 1 = 1$	$x \wedge 0 = 0$
3. Idempotenz-Gesetz	$x \vee x = x$	$x \wedge x = x$
4.	$x \vee \overline{x} = 1$	$x \wedge \overline{x} = 0$
5. Doppelte Negation	$\overline{\overline{x}} = x$	
6. Kommutativgesetze	$a \vee b = b \vee a$	$a \wedge b = b \wedge a$
7. Assoziativgesetze	$a \vee b \vee c = a \vee (b \vee c) = (a \vee b) \vee c$	$a \wedge b \wedge c = a \wedge (b \wedge c) = (a \wedge b) \wedge c$
8. Distributivgesetze	$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$	$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$
9. Absorptionsgesetze	$a \vee (a \wedge b) = a$	$a \wedge (a \vee b) = a$
10.	$a \vee (\overline{a} \wedge b) = a \vee b$	$a \wedge (\overline{a} \vee b) = a \wedge b$
11. Expansionsgesetze	$a = (a \wedge b) \vee (a \wedge \overline{b})$	$a = (a \vee b) \wedge (a \vee \overline{b})$
12. Theoreme von de Morgan	$\overline{x_0 \vee x_1 \vee x_2 \vee \ldots \vee x_n} = \overline{x}_0 \wedge \overline{x}_1 \wedge \overline{x}_2 \wedge \ldots \wedge \overline{x}_n$ $\overline{x_0 \wedge x_1 \wedge x_2 \wedge \ldots \wedge x_n} = \overline{x}_0 \vee \overline{x}_1 \vee \overline{x}_2 \vee \ldots \vee \overline{x}_n$	
13. Satz von Shannon	$\overline{F}(x_0, x_1, x_2, \ldots, x_n, \wedge, \vee) = F(\overline{x}_0, \overline{x}_1, \overline{x}_2, \ldots, \overline{x}_n, \vee, \wedge)$	

Logiksymbole

Funktion	Schaltsymbol	Gleichung	Wertetabelle	Kontaktplan
UND		$F = a \bullet b$	a b F 0 0 0 0 1 0 1 0 0 1 1 1	
ODER		$F = a + b$	a b F 0 0 0 0 1 1 1 0 1 1 1 1	
NICHT		$F = \overline{a}$	a F 0 1 1 0	
NAND		$F = \overline{a \bullet b}$	a b F 0 0 1 0 1 1 1 0 1 1 1 0	
NOR		$F = \overline{a + b}$	a b F 0 0 1 0 1 0 1 0 0 1 1 0	
EXOR		$F = \overline{a}b + a\overline{b}$ $F = a \oplus b$	a b F 0 0 0 0 1 1 1 0 1 1 1 0	
EXNOR		$F = ab + \overline{a}\overline{b}$ $F = \overline{a \oplus b}$	a b F 0 0 1 0 1 0 1 0 0 1 1 1	

Normalformen

Für den systematischen Entwurf digitaler Schaltungen wird die Darstellung von Funktionen in der so genannten **Normalform** benötigt. In den Normalformen werden die logischen Funktionen in einer einheitlichen Form beschrieben, in der nur Negationen, UND sowie ODER vorkommen. In der Booleschen Algebra sind zwei Normalformen gebräuchlich, die **disjunktive Normalform DNF** und die **konjunktive Normalform KNF**.

- **Minterm:** Ein Minterm ist die **konjunktive** Verknüpfung **(UND)** aller Eingangsvariablen, wobei jede Eingangsvariable (negiert oder nichtnegiert) vorkommen muss. Ein **Minterm** ist somit eine Boolesches Funktion, die für genau eine Eingangskombination den Ausgangswert 1 und für alle anderen Kombinationen den Wert 0 annimmt. Für n Eingangsvariable existieren genau 2^n Minterme.
- **Maxterm**: Ein Maxterm ist die **disjunktive** Verknüpfung **(ODER)** aller Eingangsvariablen, wobei jede Eingangvariable (negiert oder nichtnegiert) vorkommen muss. Die zugehörigen Maxterm-Funktionen besitzen bei genau einer Eingangskombination den Ausgangswert 0, bei allen übrigen $2^n - 1$ Eingangskombinationen hat die Funktion den Wert 1.

Die Normalform einer logischen Funktion erhält man entweder durch

- **disjunktive** Verknüpfungen von **Mintermen (disjunktive Normalform DNF)** oder durch
- **konjunktive** Verknüpfungen von **Maxtermen (konjunktive Normalform KNF)**.

Die **disjunktive Normalform DNF** einer Schaltfunktion wird aus der Wahrheitstabelle gewonnen, indem

- für jede Zeile der Wertetabelle, in der eine **1** als Funktionswert steht, der zugehörige **Minterm** gebildet wird.
- steht eine **0** in der Eingangsvariablenspalte, wird diese Variable **negiert**, steht dort eine **1**, wird sie **nicht negiert** in die UND-Verknüpfung (Konjunktion) aller Eingangsvariablen aufgenommen.
- die so entstandenen **Minterme** werden untereinander ODER-verknüpft (disjunkt =>DNF)

Die **konjunktive Normalform KNF** einer Schaltfunktion wird aus der Wahrheitstabelle gewonnen, indem

- für jede Zeile der Wertetabelle, in der eine **0 als Funktionswert** steht, der zugehörige Maxterm gebildet wird.
- steht eine **1** in der Eingangsvariablenspalte, wird diese Variable **negiert**, steht dort eine **0**, wird sie **nicht negiert** in die ODER-Verknüpfung (Disjunktion) aller Eingangsvariablen aufgenommen
- die so entstandenen **Maxterme** werden untereinander UND-verknüpft (konjunktiv =>KNF)

A	B	C	Q
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Minterm

DNF: $Q = (\bar{A} \wedge \bar{B} \wedge C) \vee (A \wedge B \wedge C) \vee (A \wedge \bar{B} \wedge \bar{C}) \vee (A \wedge \bar{B} \wedge C) \vee (A \wedge B \wedge \bar{C})$

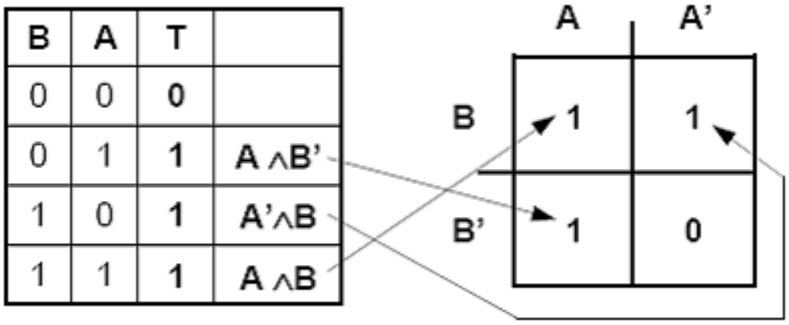
KNF: $Q = (A \vee B \vee C) \wedge (A \vee \bar{B} \vee \bar{C}) \wedge (\bar{A} \vee \bar{B} \vee \bar{C})$

Maxterm

Je nachdem, ob für eine gegebene Schaltfunktion die Zahl der Minterme (**1** in der Ausgangsspalte) oder der Maxterme (**0** in der Ausgangsspalte) kleiner ist, ergibt entweder die DNF oder die KNF einen einfacheren Logik-Ausdruck.

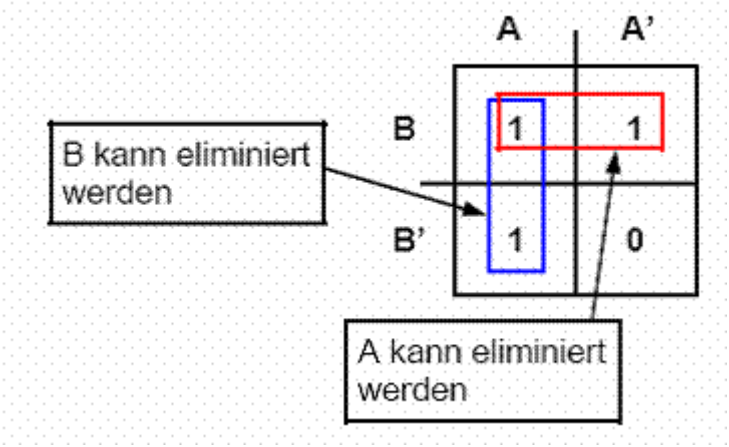
KV-Diagramme

1. Schritt: Wahrheitstabelle in das KV-Diagramm überführen:



B	A	T	
0	0	0	
0	1	1	A ∧B'
1	0	1	A'∧B
1	1	1	A ∧B

2. Schritt: Rechtecke bilden:



3. Schritt: Vereinfachte Funktionsgleichung erstellen:

Für jedes Rechteck kann anhand der Koordinatenbezeichnungen eine Formel angegeben werden. Dabei werden Variablen welche **invers** und **nicht invers** (in Eingenform) auftreten, **weggelassen.**

Für das Beispiel im Bild oben gilt:

- Im senkrechten Rechteck kommt die Variablen negiert und nichtnegiert vor und kann entfallen daher:

  $(A \wedge B) \vee (A \wedge \overline{B}) = A \vee (B \wedge \overline{B}) = A \wedge (1) = A$

- Im waagrechten Rechteck tritt A negiert und nichtnegiert auf und entfällt daher.

  $(A \wedge B) \vee (\overline{A} \wedge B) = B \wedge (A \vee \overline{A}) = B \wedge (1) = B$

- Die vereinfachte Formel lautet: $\mathbf{Y = A \vee B}$

Regeln für die Gruppenbildung

- Benachbarte Felder in die eine Eins eingetragen ist, werden zu Gruppen zusammengefasst.
- Eine Gruppe darf keine Felder mit Nullen enthalten. (Oft werden die Nullen nicht mitgeschrieben und die Felder leer gelassen. In diesem Fall darf eine Gruppe keine leeren Felder enthalten.)
- Alle Einsen müssen in Gruppen zusammengefasst werden.
- Benachbarte Felder mit Einsen werden zu einer Gruppe zusammengefasst. (Felder, die sich diagonal an den Ecken berühren, zählen nicht als benachbart.

- Die Gruppen müssen so groß wie möglich sein.
- Es müssen so wenig Gruppen wie möglich gebildet werden.
- Die Gruppen dürfen nur eine Größe haben, die einer Zweierpotenz entspricht (1, 2, 4, 8, 16 …).
- Die Gruppen müssen rechteckige Blöcke sein.
- Die Gruppen dürfen sich überlappen.
- Die Gruppen dürfen über die Ränder hinweggehen. (Bei KV-Diagrammen mit 3 Variablen sind rechter und linker Rand benachbart, bei KV-Tafeln mit 4 Variablen zusätzlich auch der obere und untere Rand.)
- Zwei Gruppen dürfen nicht exakt die gleichen Einsen umfassen.
- Es darf keine Gruppe vollständig von einer anderen Gruppe umschlossen werden.
- Die reduzierte Formel ergibt sich als ODER-Verknüpfung der einzelnen „Rechteck-Formeln“.

Zusammenfassung:

Man sucht eine vollständige Überdeckung der Einsen mit möglichst großen rechteckigen Blöcken.

Bezeichnungen beim Karnaugh-Diagramm

Primimplikanten (PI): Bezeichnung für die so groß wie möglich gewählten Blöcke von 'Einsen' im Karnaugh-Diagramm

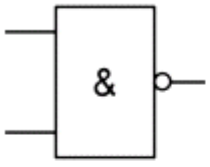
Essentielle PI: Primimplikanten, die mindestens eine '1' enthalten, die sonst von keinem anderen Block abgedeckt sind → die minimale Lösung enthält zumindest die essentiellen Primimplikanten.

Nicht-essentielle PI: Primimplikanten, die obige Bedingung nicht erfüllen.

Redundante PI: Nicht essentielle PI, die nur bereits von essentiellen PI abgedeckte 'Einsen' markieren → überflüssig.

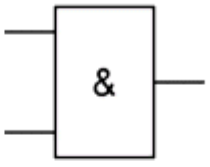
Die Gatter als KV-Tafeln

NAND-Gatter



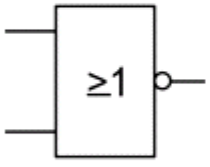
	A	$\bar{A}$
B	0	1
$\bar{B}$	1	1

AND-Gatter



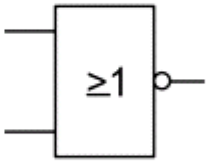
	A	$\bar{A}$
B	1	0
$\bar{B}$	0	0

NOR-Gatter



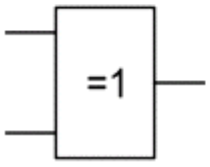
	A	$\bar{A}$
B	0	0
$\bar{B}$	0	1

OR-Gatter



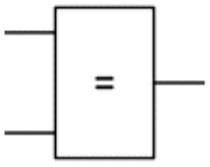
	A	$\bar{A}$
B	1	1
$\bar{B}$	1	0

XOR-Gatter



	A	$\bar{A}$
B	0	1
$\bar{B}$	1	0

XNOR-Gatter

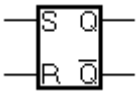


	A	$\bar{A}$
B	1	0
$\bar{B}$	0	1

Flipflops

Das Basis-Flipflop

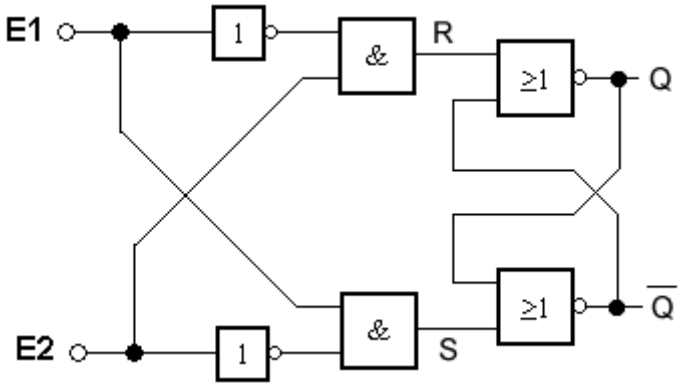
Das RS-Flipflop (Rücksetz-/Setz-FF, synonym zu SR-FF) bildet das einfachste Basis-Flipflop. Für das SR-FF werden die Eingänge mit "S" (Setzen) und "R" (Rücksetzen) sowie der Ausgang mit "Q" bezeichnet. Es gilt dann:

S	R	Q	Schaltzeichen
0	0	Speichern	
1	0	1, Setzen	
0	1	0, Rücksetzen	
1	1	Verboten	

Charakteristische Gleichung: $Q_{t+1} = S \vee (Q_t \wedge \neg R)$

Das E-Flipflop

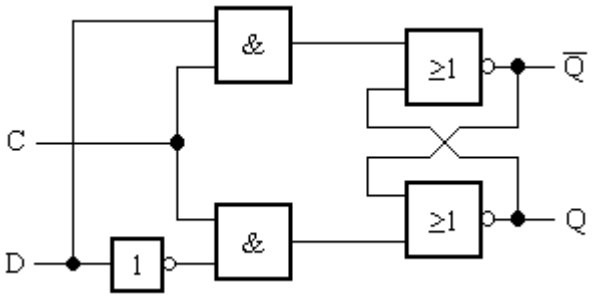
Das E-Flipflop ist eine Sonderausführung des RS-Flipflops.



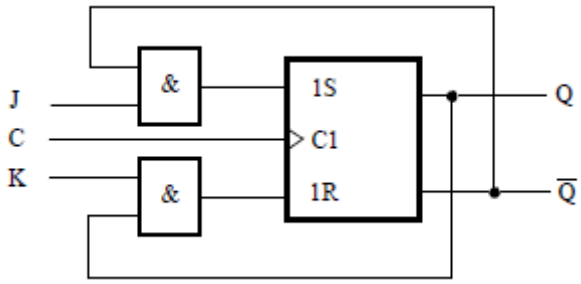
D-Flipflop (Latch)

Der R-Eingang hat immer den invertierten Wert des S-Eingangs. Der verbotene bzw. nicht definierte Zustand S = R = 1 des RS-Flip-Flops wird damit verhindert.

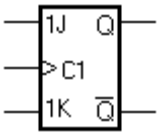
Die charakteristische Gleichung lautet: $Q_{+1} = D$



Einflankengesteuertes JK-Flipflop



Wahrheitstabelle:

J	K	Q_{n+1}	Schaltzeichen
0	0	Qn	
1	0	1	
0	1	0	
1	1	$\neg Q_n$	

Die charakteristische Gleichung lautet dann: $Q_{+1} = (\neg K \wedge Q) \vee (J \wedge \neg Q)$

Zweiflankengesteuertes JK-Flipflop

Das zweiflankengesteuerte JK-Flipflop, auch "Master-Slave-JK-Flipflop" genannt: