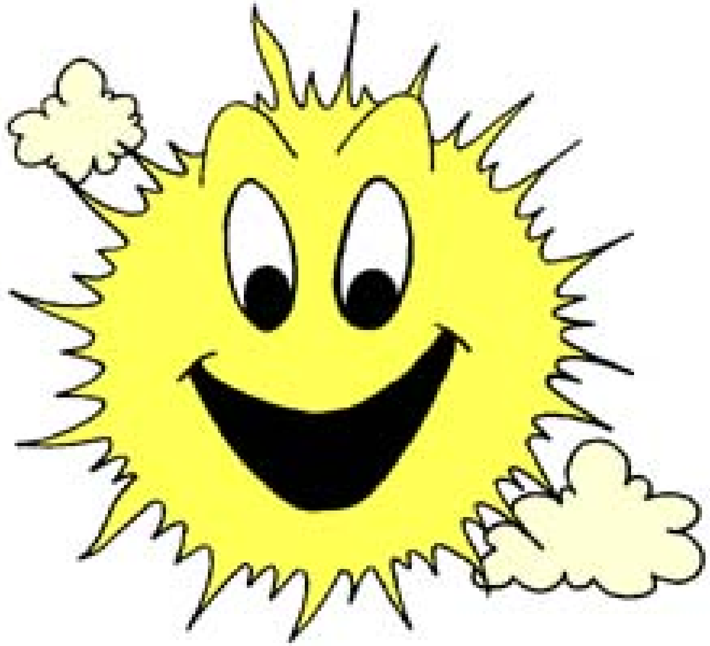
Wiederholung



1.2 Nachricht und Information

- Eine **Nachricht** ist eine **Zeichenfolge**, die **nach bestimmten Regeln gebildet** wird, **Syntax**
  - Bsp. Glockenschlag der Kirchturmuhr

- **Information** ist die **Bedeutung einer Nachricht für den Empfänger**, **Semantik**
  - Bsp. Tageszeit

1.3 Bits und Bitfolgen (2)

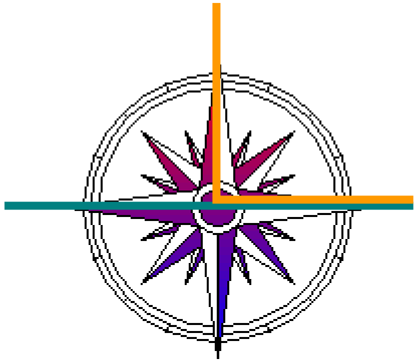
Maßeinheit der Information ist ein Bit.

1 Bit ist die kleinst-mögliche Informationsmenge, d.h die Informationsmenge einer Nachricht, die in einer Antwort auf eine Frage enthalten ist, die nur zwei mögliche Antworten zulässt:

z.B.:

- ja – nein
- wahr – falsch
- schwarz – weiss
- 0 - 1

1.3 Bits und Bitfolgen (4) Beispiel Wind



1. weht Wind aus NORD oder OST?
2. weht Wind aus OST oder WEST?

Antwort ja = 1, nein = 0.

Antworten auf
beide Fragen:

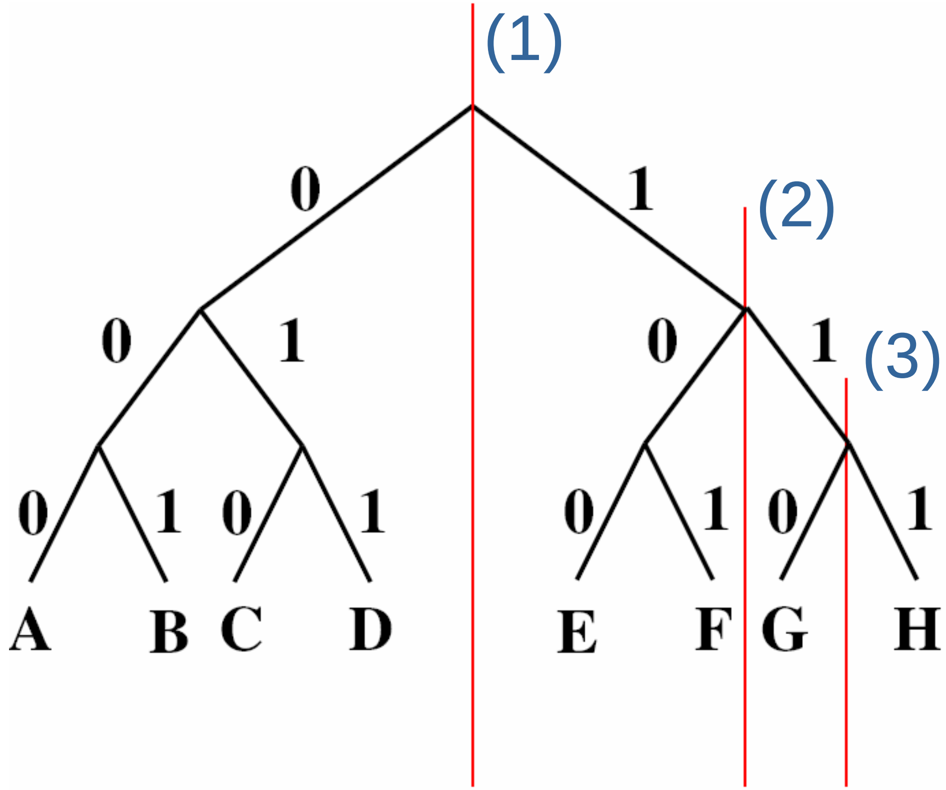
```
1 0 : Nord
1 1 : Ost
0 1 : West
0 0 : Süd
```

Frage nach der Windrichtung kennt **vier** Antworten, **Informationsgehalt** ist aber nur **2** Bit, 2 Elementar-Fragen.

Mit 2 Bit können vier Möglichkeiten dargestellt werden.

1.3 Buchstaben A-H (Beispiel)

Informatik



(1) E ≤ Buchstabe ≤ H ?

(2) G ≤ Buchstabe ≤ H ?

(3) Buchstabe = H ?

Buchstaben A-H können mit 3 Bit dargestellt werden:

Beisp:

0 0 0 : A

1 0 1 : F usw.

1.4 Codierung / Decodierung

- **Zeichenvorrat**: endliche Menge von unterscheidbaren Dingen
- **Zeichen**: ein Element des Zeichenvorrats
- **Code**: Vorschrift für die eindeutige Zuordnung der Zeichen eines Zeichenvorrats zu denjenigen eines anderen Zeichenvorrats

1.5 Präfixfreie Codes (4)

Eine Zeichenkette Z heißt Präfix der Zeichenkette W genau dann, wenn die Länge von Z kleiner als die von W ist und der Anfang von W (v.l.) mit Z identisch ist.

Präfixfreie Codes

Codewörter und Sequenzen von Codewörtern eines präfixfreien Codes können eindeutig decodiert werden.

1.8 Darstellung ganzer Zahlen (2)

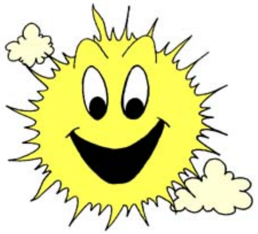
$$Z = \sum_i x_i Y^i,$$

wobei *Y* die Basis des Zahlensystems bezeichnet, *i* stellt die Stelle der Ziffern dar und x_i den Wert der *i*-ten Ziffer

Beispiel:

$1011_{(2)} = 1*8 + 0*4 + 1*2 + 1*1 = 11_{(10)}$

Ende der Wiederholung



→ → → → → → → → → → → → → → → → →

2 Boolesche Algebra – Inhalt

1. Die Boolesche Algebra
2. Boolesche Funktionen
3. Die Booleschen Postulate
4. De Morgansche Gesetz
5. Dualitätsprinzip, Shannonsche Gesetz

2 Boolesche Algebra

Die Boolesche Algebra (Schaltalgebra) dient der Beschreibung digitaler Funktionen.

Ähnlich wie in der Algebra werden Variablen über Operatoren verknüpft. Es lassen sich Gleichungen aufstellen und umformen.

Bei der Booleschen Algebra nehmen die Variablen nur die Werte 0 und 1 an.

1854: Theorie durch Mathematiker George Boole

1940: Claude E. Shannon: Anwendung auf Schaltnetze

3 Boolesche Funktionen – Inhalt

1. Funktionen mit 1 Eingang und 1 Ausgang
2. Funktionen mit 2 Eingängen und 1 Ausgang
3. Boolesche Funktionen mit mehreren Variablen

3.1 Schließer und Öffner

Schließer

X = 0: Schalter offen

X = 1: Schalter geschlossen

Öffner

X = 0: Schalter geschlossen

X = 1: Schalter offen

3.1 Negation (NOT)

- „Wenn meine Schwiegermutter zu Besuch kommt, gehe ich heute Abend nicht ins Theater."
- Wenn die Aussage A „Schwiegermutter kommt zu Besuch" wahr ist, kann die Aussage X „Theaterbesuch" nicht wahr sein.
- Unäre Operation

3.2 UND Verknüpfung (AND)

- „Wenn morgen schönes Wetter ist und mein Bruder Zeit hat, gehen wir segeln.“
- Aussage A „schönes Wetter“ und Aussage B „mein Bruder Zeit hat“ müssen zutreffen, damit die Aussage X „segeln gehen“ wahr wird.
- Binäre Operation

3.2 ODER-Verknüpfung (OR)

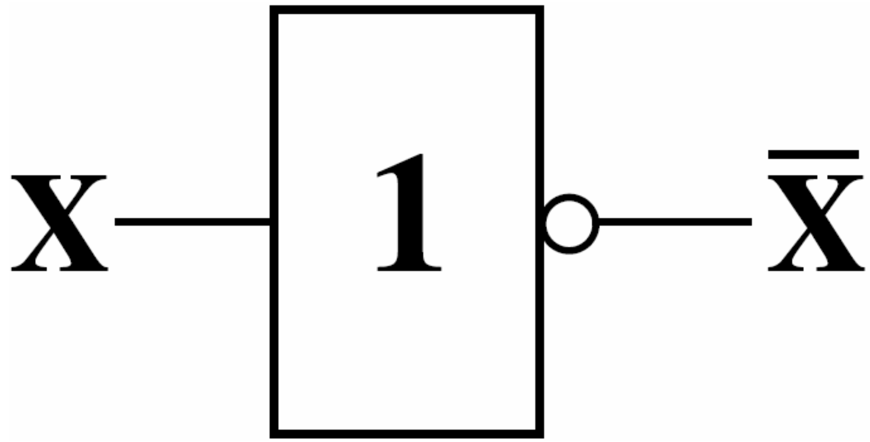
- „Wenn ich eine Erbschaft mache oder im Lotto gewinne, mache ich eine Weltreise."
- Wenn Aussage A „Erbschaft" oder Aussage B „Lottogewinn" zutrifft, oder beide Aussagen zutrefffen, wird Aussage X „Weltreise machen" wahr.
- Binäre Operation

3. Symbole und Zeichen logischer Funktionen

- Realisierung durch Schalter
- Schaltsymbole nach DIN 40900
- Formelzeichen der Booleschen Algebra
- Wahrheitstabellen (Funktion in Tabellenform)
- Weitere Schaltsymbole nach DIN und ANSI / ISO (teils veraltete Normen)

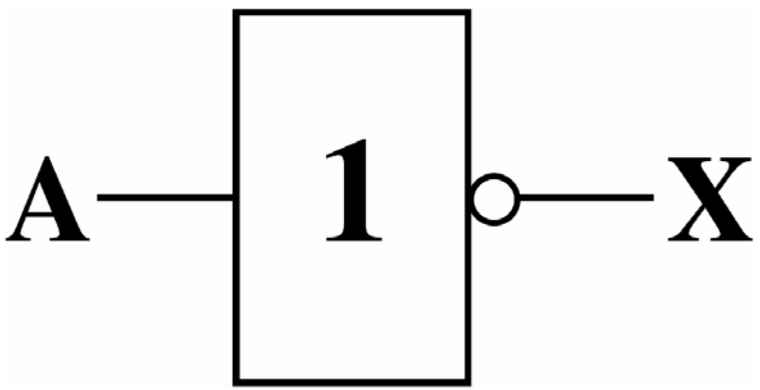
3.1 Negation (NOT)

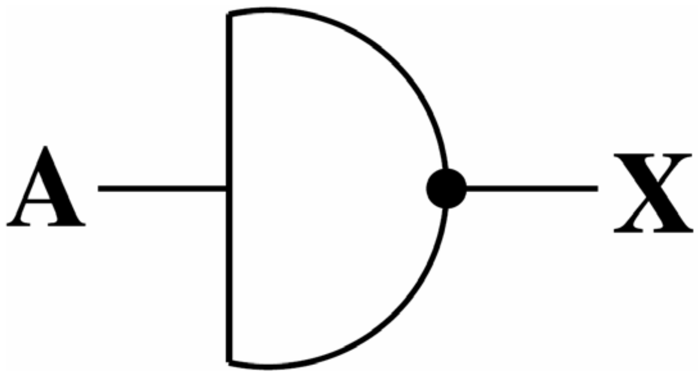
X	$\overline{X}$
0	1
1	0

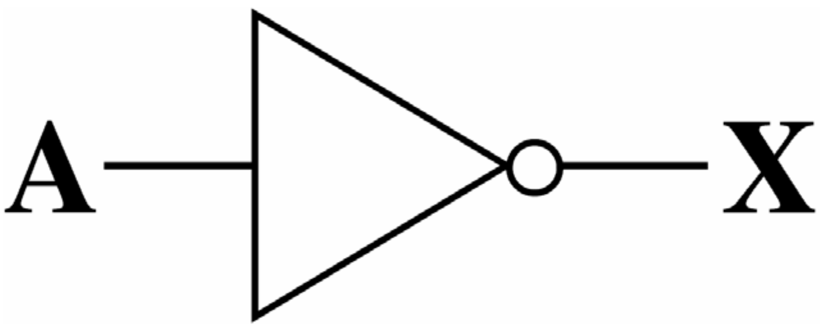


Negation: $\neg X$, $\overline{X}$, $!X$

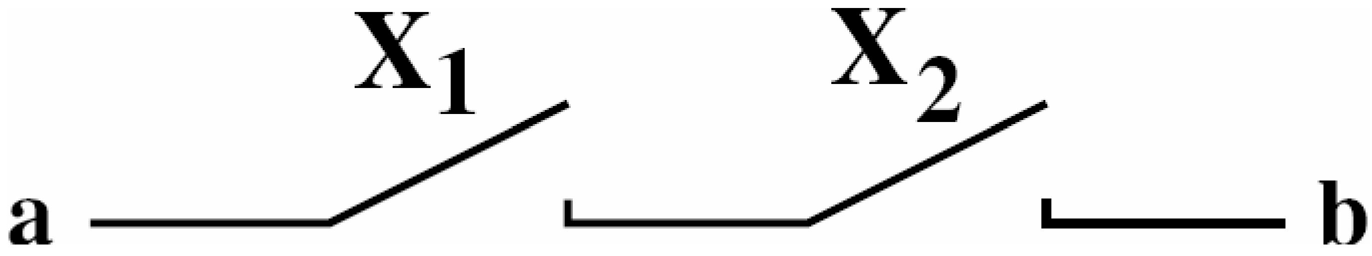
3.1 Schaltzeichen Negation (NOT)



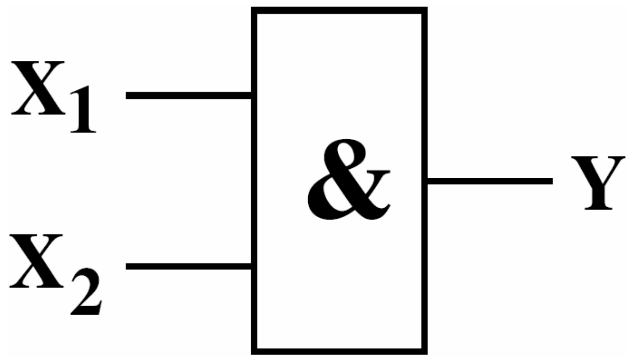




3.2 UND-Verknüpfung (AND)

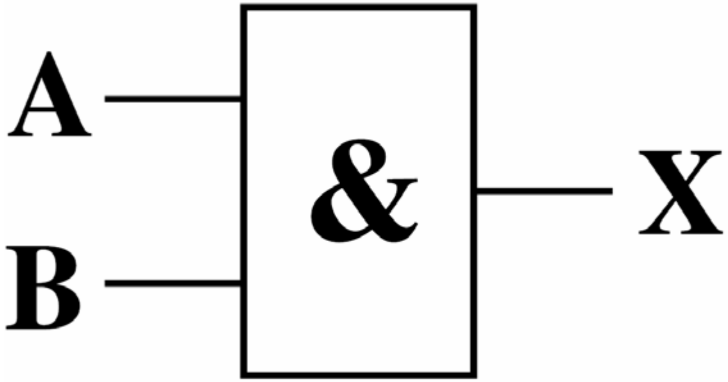


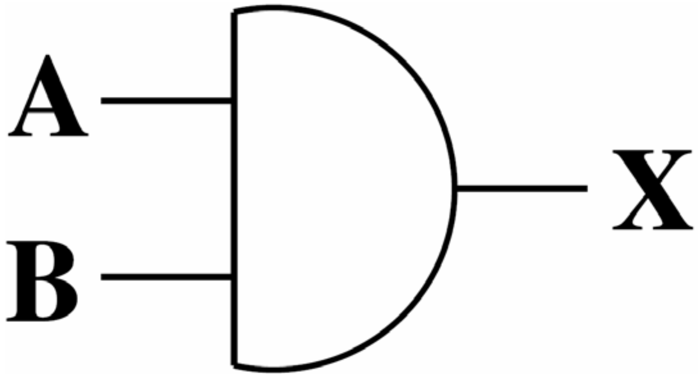
X_1	X_2	$X_1 \wedge X_2$
0	0	0
0	1	0
1	0	0
1	1	1

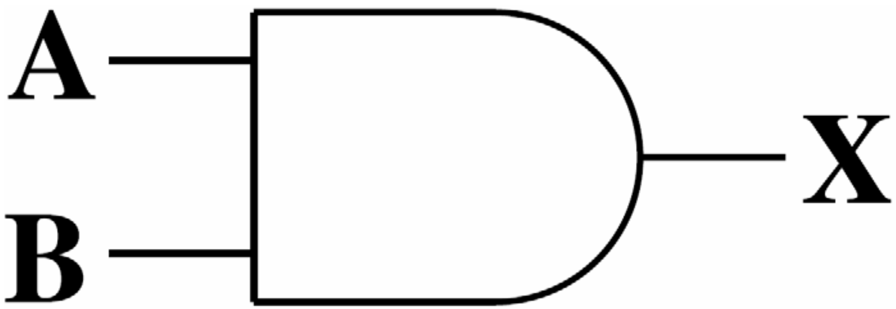


UND: $\wedge$, $\bullet$

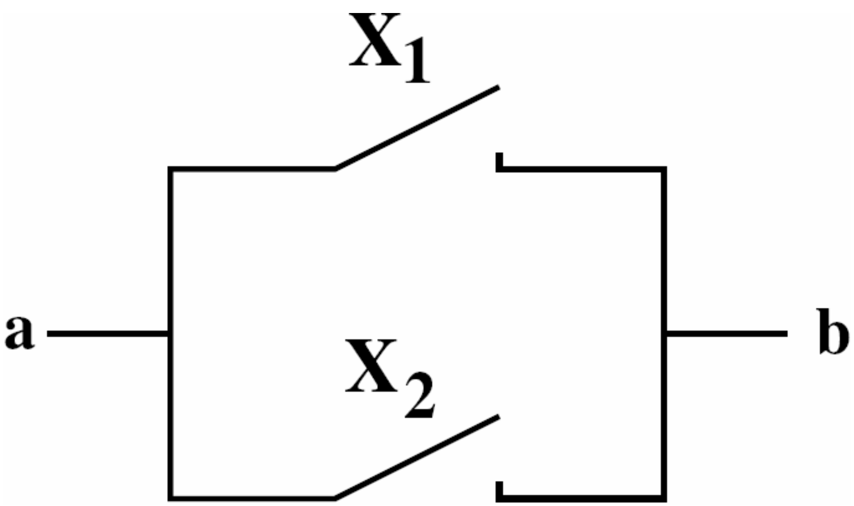
3.2 Schaltzeichen UND (AND)

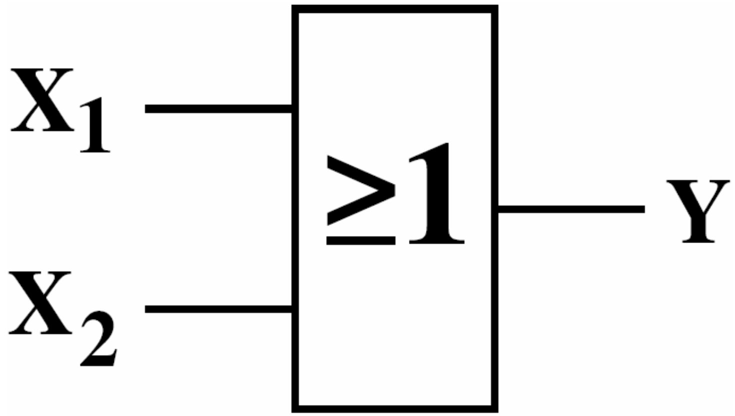






3.2 ODER-Verknüpfung (OR)

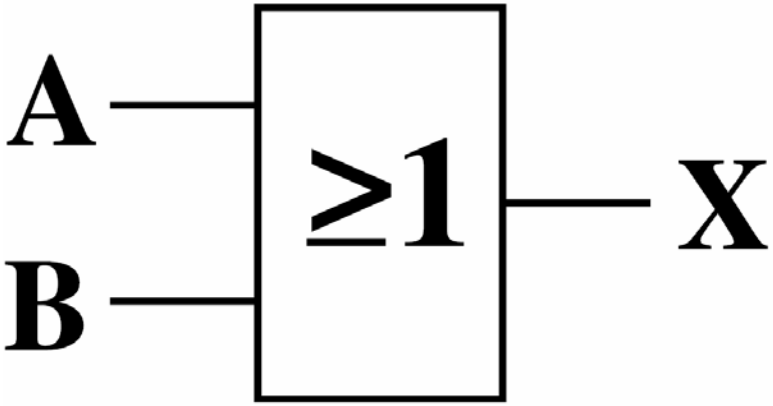


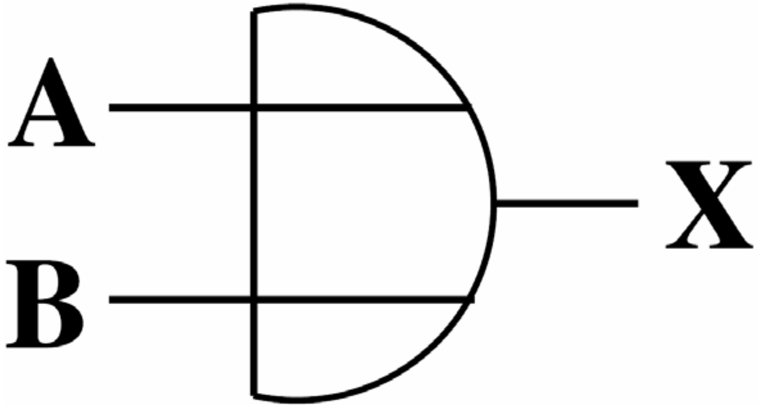


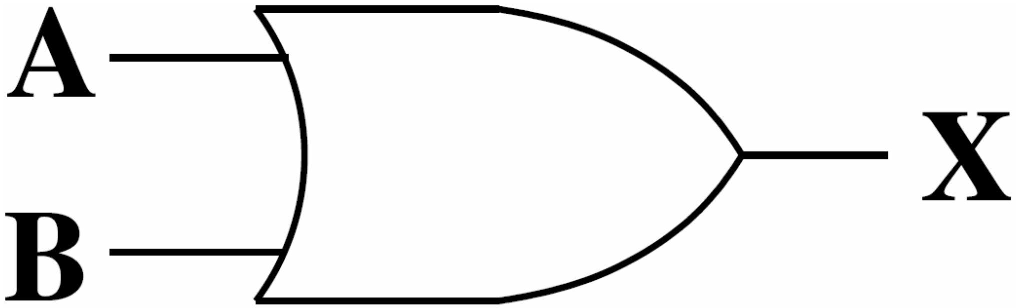
X_1	X_2	$X_1 \vee X_2$
0	0	0
0	1	1
1	0	1
1	1	1

ODER: $\vee$, $+$

3.2 Schaltzeichen ODER (OR)

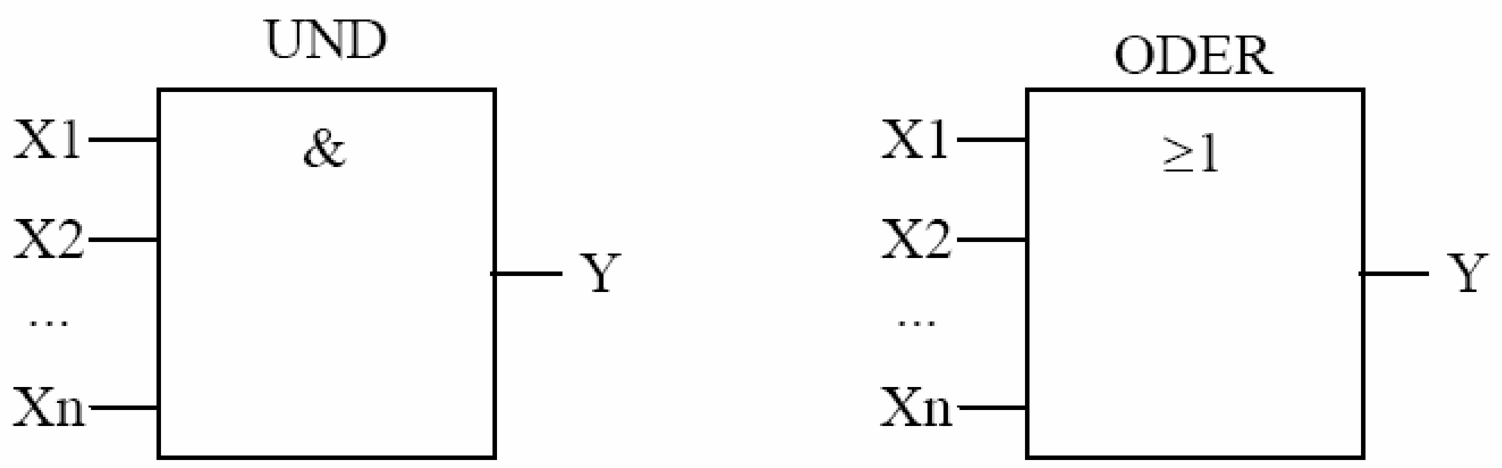






3.3 Funktionen mehrere Variablen

- Neben UND und ODER gibt es natürlich weitere Funktionen
- Funktionen können mehrere Variablen haben: $Y_B(X_1, X_2, \ldots, X_n)$

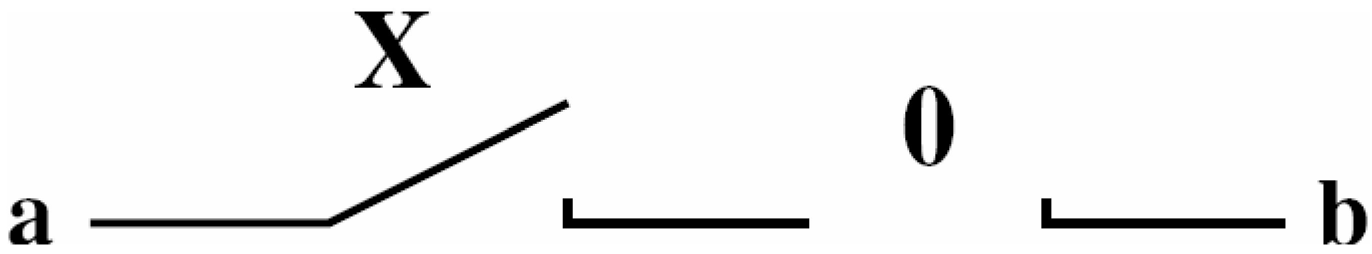


3. Boolesche Postulate

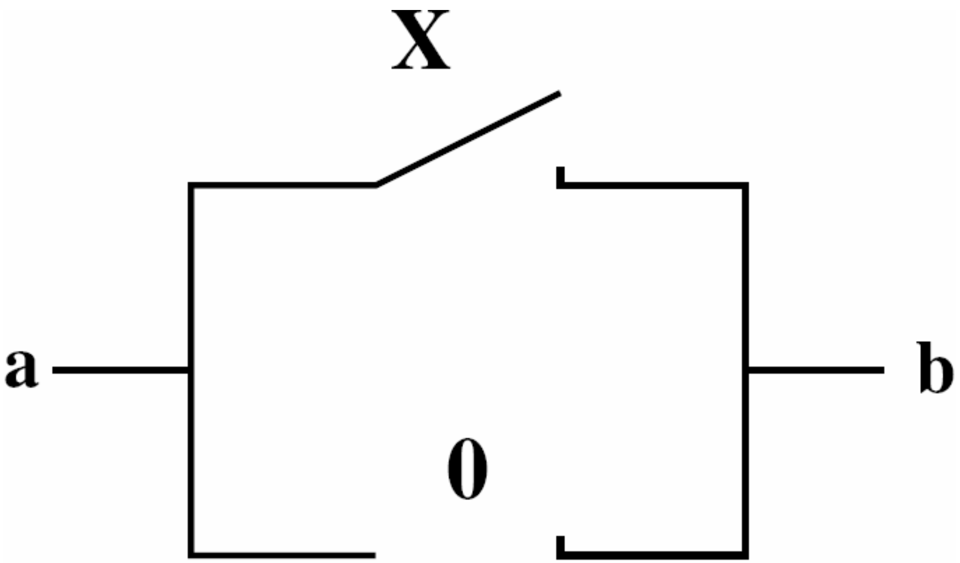
Menge A mit den Elementen 0 und 1 und den Operationen UND, ODER und Negation.

P1	a=0 oder a=1	P5	$1 \vee 1 = 1$
P2	$0 \wedge 0 = 0$	P6	$1 \wedge 0 = 0,\ 0 \wedge 1 = 0$
P3	$1 \wedge 1 = 1$	P7	$1 \vee 0 = 1,\ 0 \vee 1 = 1$
P4	$0 \vee 0 = 0$	P8	$\neg 1 = 0,\ \neg 0 = 1$

3.1 Das NULL Gesetz

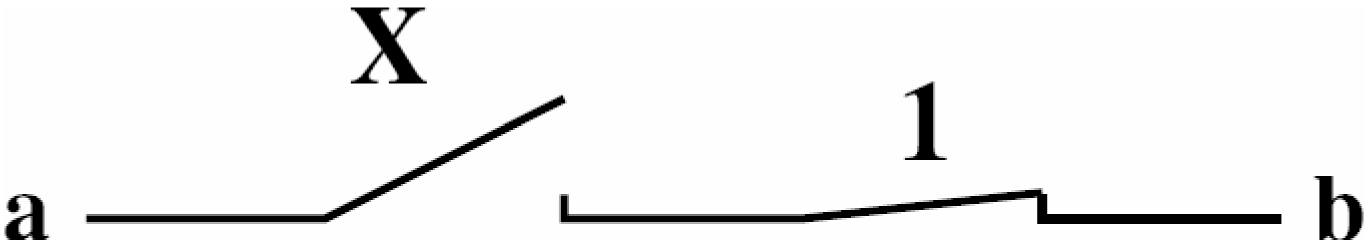


$$X \wedge 0 = 0$$

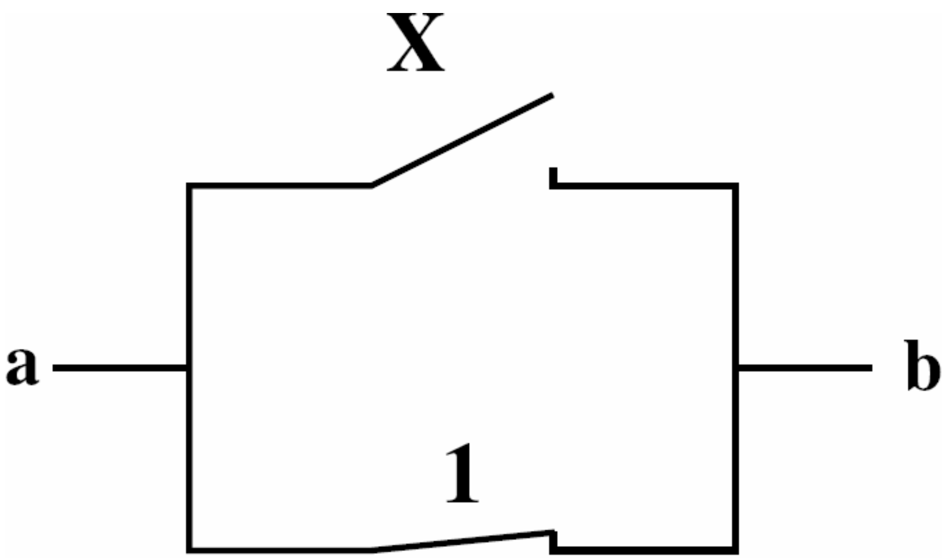


$$X \vee 0 = X$$

3.1 Das EINS Gesetz



$$X \wedge 1 = X$$



$$X \vee 1 = 1$$

4 Vollst. Systeme aus UND, ODER, NICHT

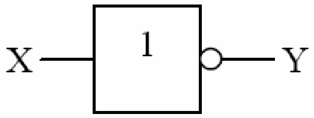
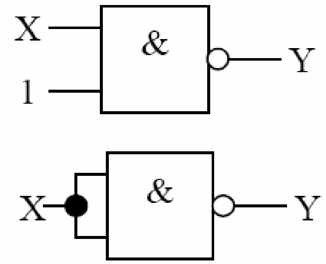
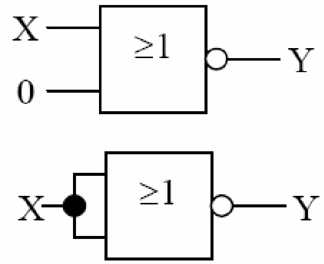
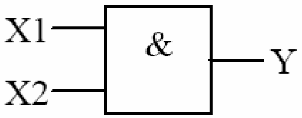
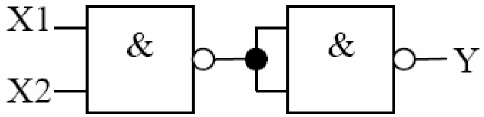
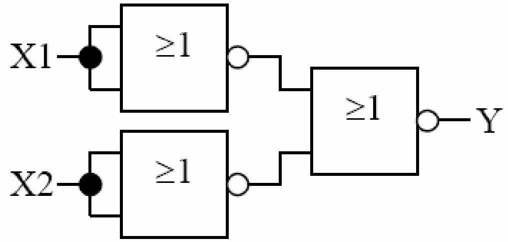
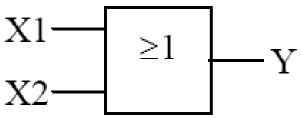
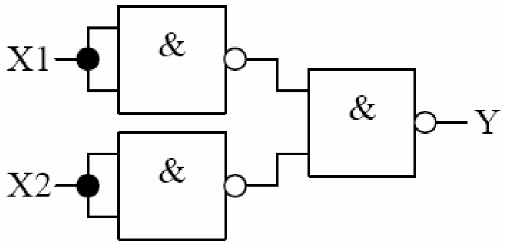
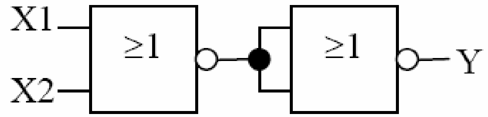
Informatik

Bisher wurden die booleschen Funktionen durch die Grundverknüpfungen UND, ODER und NICHT dargestellt. Diese Verknüpfungen besitzen eine besondere Bedeutung, da man mathematisch nachweisen kann, daß sich mit diesen drei Operationen alle booleschen Funktionen darstellen lassen. Man bezeichnet daher diese Verknüpfungen auch als vollständiges System.

4 Vollst. Syst. aus UND, ODER, NICHT (2)

Für den Schaltungsentwurf hat dies die Attraktivität, dass lediglich drei Grundschaltungen benötigt werden, welche diese drei Operationen in Hardware realisieren. Durch Replizieren und Kombination dieser drei Grundschaltungen kann jede boolesche Schaltung aufgebaut werden.

5 Vollst. Systeme mit NAND und NOR

Grundverknüpfung	Realisiert mit NAND	Realisiert mit NOR
Negation, NICHT  $Y = \overline{X}$	(i) (ii)  (i) $Y = \overline{X \wedge 1}$ (ii) $Y = \overline{X \wedge X}$	(i) (ii)  (i) $Y = \overline{X \vee 0}$ (ii) $Y = \overline{X \vee X}$
Konjunktion, UND  $Y = X1 \wedge X2$	 $Y = \overline{\overline{(X1 \wedge X2)} \wedge \overline{(X1 \wedge X2)}}$	 $Y = \overline{\overline{(X1 \vee X1)} \vee \overline{(X2 \vee X2)}}$
Disjunktion, ODER  $Y = X1 \vee X2$	 $Y = \overline{\overline{(X1 \wedge X1)} \wedge \overline{(X2 \wedge X2)}}$	 $Y = \overline{\overline{(X1 \vee X2)} \vee \overline{(X1 \vee X2)}}$

5 Vollst. Systeme mit NAND und NOR (2)

Ebenso wie die AND-, OR-Verknüpfung und NOT bilden NAND und NOR, jede für sich, ebenfalls vollständige Systeme.

Somit reicht es für die Entwicklung einer booleschen Hardware aus, entweder eine Grundschaltung für NAND oder für NOR zur Verfügung zu stellen.

Daraus kann durch Replizieren und Kombinieren jede boolesche Schaltung entwickelt werden.