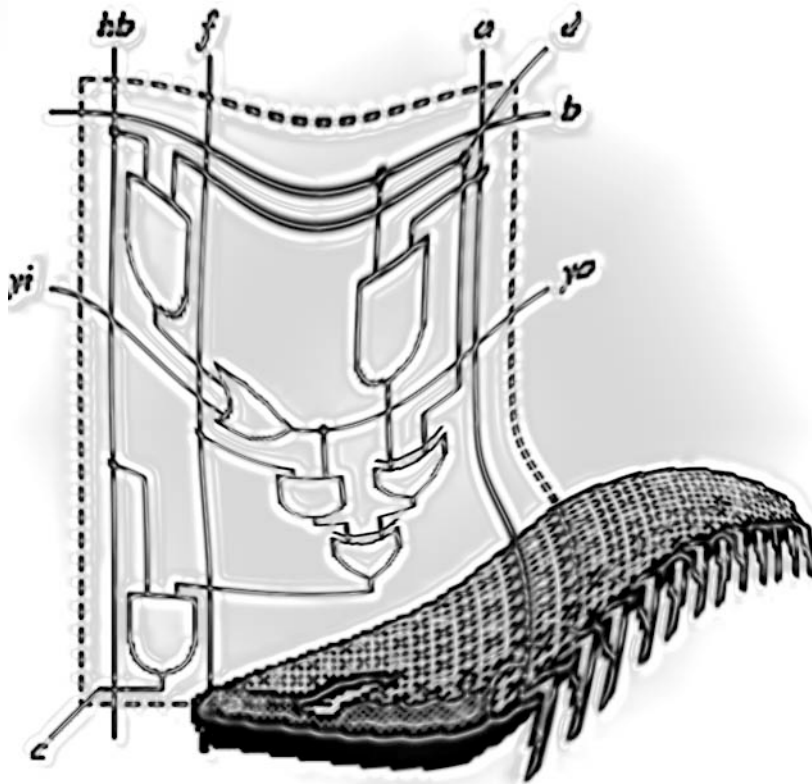


# Technische Informatik I

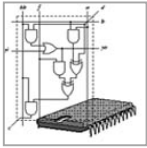
## Kapitel 2

### Boolesche Algebra



Prof. Dr. Dirk W. Hoffmann





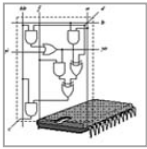
# Boolesche Algebra



Gegeben: Menge  $V$ , Operatoren  $\bullet, + : V \times V \rightarrow V$

$V$  heißt Boolesche Algebra,  
wenn die folgenden vier Huntington'schen Axiome gelten:

- Kommutativgesetze (K):  $a \bullet b = b \bullet a$   
 $a + b = b + a$
- Distributivgesetze (D):  $a \bullet (b + c) = (a \bullet b) + (a \bullet c)$   
 $a + (b \bullet c) = (a + b) \bullet (a + c)$
- Neutrale Elemente (N): Es existieren  $e, n \in V$  mit  
 $a \bullet e = a$  und  $a + n = a$
- Inverse Elemente (I): Für alle  $a \in V$  existiert ein  $a'$  mit  
 $a \bullet a' = n$  und  $a + a' = e$



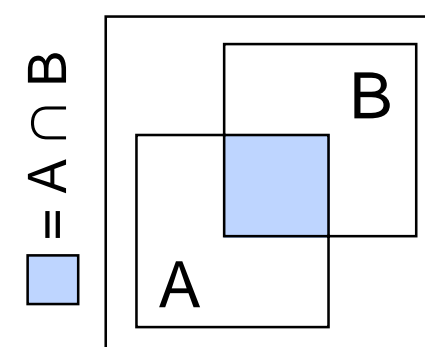
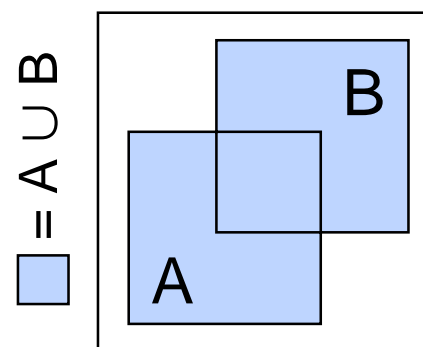
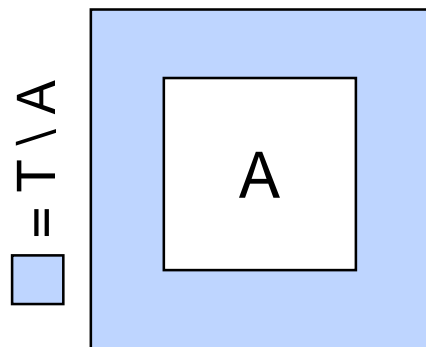
# Boolesche Algebra: Beispiele

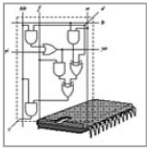


## ■ Beispiel 1: Mengenalgebra ( $T$ = Trägermenge )

Boolesche Algebra	Mengenalgebra	
$V$	$\wp(T)$	Potenzmenge der Trägermenge $T$
$\cdot$	$\cap$	Durchschnitt
$+$	$\cup$	Vereinigung
$n$	$\emptyset$	Leere Menge
$e$	$T$	Trägermenge
$a'$	$T \setminus A$	Komplementärmenge

## ■ Veranschaulichung durch Venn-Diagramme

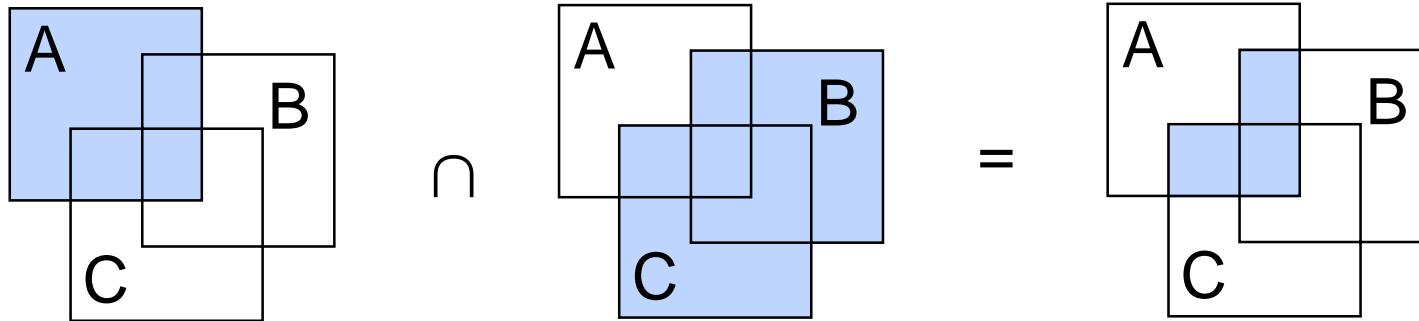




# Boolesche Algebra: Beispiele

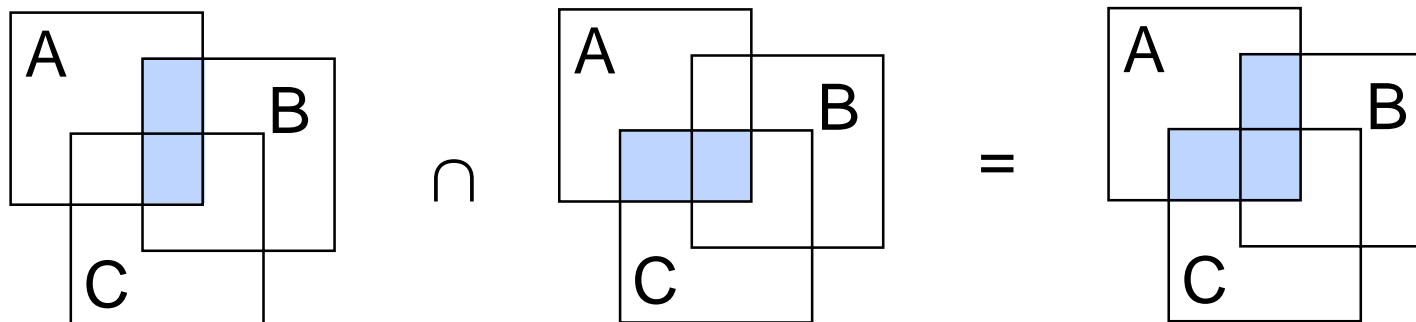


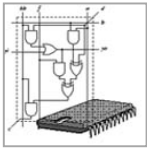
- $A \cap (B \cup C)$



II

- $(A \cap B) \cup (B \cap C)$

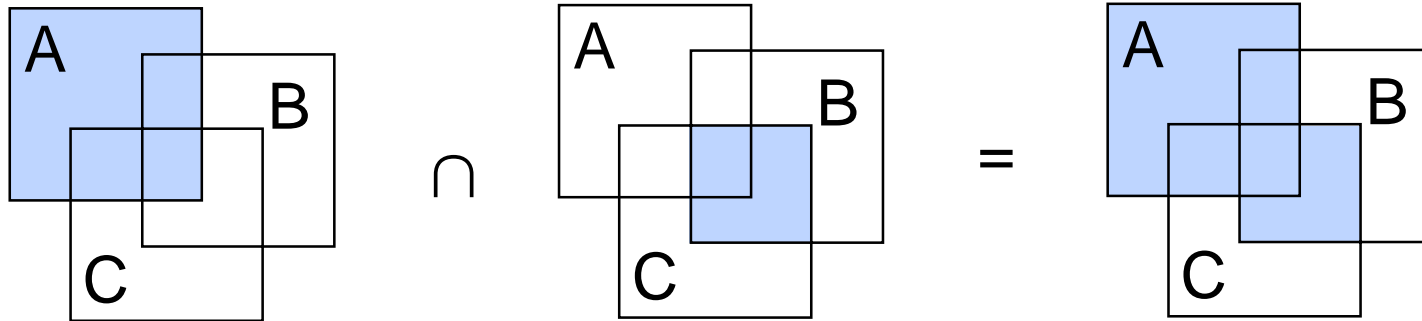




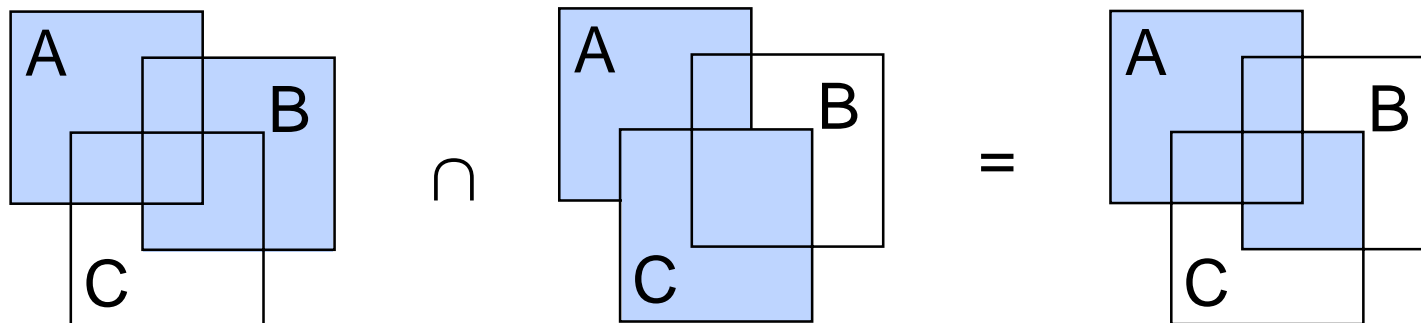
# Boolesche Algebra: Beispiele



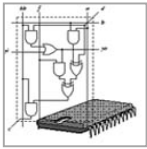
- $A \cup (B \cap C)$



- $(A \cup B) \cap (A \cup C)$



II



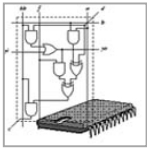
# Boolesche Algebra: Beispiele



## ■ Beispiel 2: Schaltalgebra

Boolesche Algebra	Schaltalgebra	
$V$	$\{ 1, 0 \}$	Wahrheitswerte (TRUE, FALSE)
$\bullet$	$\wedge$	Konjunktion (UND-Operator)
$+$	$\vee$	Disjunktion (ODER-Operator)
$n$	$0$	„Falsch“ (FALSE)
$e$	$1$	„Wahr“ (TRUE)
$a'$	$\neg a$	Negation (Verneinung)



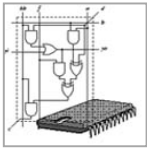


# Notation und Operatorenbindung



- Abgeleitete Operatoren („*syntactic sugar*“)
  - $(a \rightarrow b)$  für  $(\neg a \vee b)$  (Implikation)
  - $(a \leftarrow b)$  für  $(b \rightarrow a)$  (Inv. Implikation)
  - $(a \leftrightarrow b)$  für  $(a \rightarrow b) \wedge (a \leftarrow b)$  (Äquivalenz)
  - $(a \oplus b)$  für  $\neg(a \leftrightarrow b)$  (Antivalenz)
  - $(a \underline{\vee} b)$  für  $\neg(a \vee b)$  (NOR)
  - $(a \underline{\wedge} b)$  für  $\neg(a \wedge b)$  (NAND)





- Boolesche Funktionen

- $\neg$  ist eine einstellige Boolesche Funktion

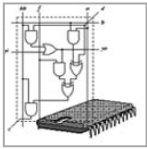
$$\neg : \{0,1\} \rightarrow \{0,1\}$$

- Alle anderen Operatoren sind zweistellige Boolesche Funktionen

$$\wedge, \vee, \dots : \{0,1\} \times \{0,1\} \rightarrow \{0,1\}$$

- Wie viele zweistellige Boolesche Funktionen gibt es insgesamt?

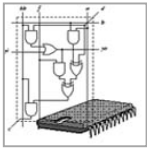




# Die zweistelligen Booleschen Funktionen



		$f_0 = 0$	$f_1 = a \wedge b$	$f_2 = \neg a \wedge b$	$f_3 = b$	$f_4 = \neg b \wedge a$	$f_5 = a$	$f_6 = a \oplus b$	$f_7 = a \vee b$	$f_8 = \neg(a \vee b)$	$f_9 = a \leftrightarrow b$	$f_{10} = \neg a$	$f_{11} = a \rightarrow b$	$f_{12} = \neg b$	$f_{13} = a \leftarrow b$	$f_{14} = \neg(a \wedge b)$	$f_{15} = 1$
b	a	$f_0$	$f_1$	$f_2$	$f_3$	$f_4$	$f_5$	$f_6$	$f_7$	$f_8$	$f_9$	$f_{10}$	$f_{11}$	$f_{12}$	$f_{13}$	$f_{14}$	$f_{15}$
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
Nullfunktion																	
Konjunktion																	
Antivalenz																	
Disjunktion																	
Äquivalenz																	
Implikation																	
Inverse Implikation																	
Einsfunktion																	



# Notation und Operatorenbindung



- Alternative Notation der Booleschen Operatoren
  - $(a \cdot b)$  bzw.  $(ab)$  anstelle  $(a \wedge b)$
  - $(a + b)$  anstelle  $(a \vee b)$
  - $\overline{a}$  anstelle  $\neg a$
  - $(a \leftrightarrow b)$  anstelle  $(a \oplus b)$
  - $(a \overline{\wedge} b)$  anstelle  $(a \underline{\wedge} b)$
  - $(a \overline{\vee} b)$  anstelle  $(a \underline{\vee} b)$
- Bindung der Operatoren
  - $\wedge$  bindet stärker als  $\vee$
  - $\neg$  bindet stärker als  $\wedge$
- Klammerung
  - Gleiche binäre Operatoren werden linksassoziativ zusammengefasst, z.B.  $a \wedge b \wedge c = (a \wedge b) \wedge c$

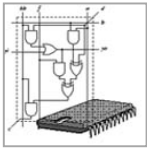


Kommutativgesetze	$a \wedge b = b \wedge a$ $a \vee b = b \vee a$	(K)
Distributivgesetze	$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$ $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$	(D)
Neutrale Elemente	$a \wedge 1 = a$ $a \vee 0 = a$	(N)
Inverse Elemente	$a \wedge \neg a = 0$ $a \vee \neg a = 1$	(I)

In jeder Booleschen Algebra, so auch in der Schaltalgebra, gelten die vier oben gezeigten Huntington'schen Axiome

Aus den Huntington'schen Axiomen lassen sich weitere praktische Rechenregeln ableiten...

Kommutativgesetze	$a \wedge b = b \wedge a$ $a \vee b = b \vee a$	(K)
Distributivgesetze	$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$ $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$	(D)
Neutrale Elemente	$a \wedge 1 = a$ $a \vee 0 = a$	(N)
Inverse Elemente	$a \wedge \neg a = 0$ $a \vee \neg a = 1$	(I)
Assoziativgesetze	$a \wedge (b \wedge c) = (a \wedge b) \wedge c = a \wedge b \wedge c$ $a \vee (b \vee c) = (a \vee b) \vee c = a \vee b \vee c$	(A)
Idempotenzgesetze	$a \wedge a = a$ $a \vee a = a$	(ID)
Absorptionsgesetze	$a \vee (a \wedge b) = a$ $a \wedge (a \vee b) = a$	(AB)
Gesetze von DeMorgan	$\neg(a \vee b) = \neg a \wedge \neg b$ $\neg(a \wedge b) = \neg a \vee \neg b$	(M)
Auslöschungsgesetze	$a \wedge 0 = 0$ $a \vee 1 = 1$	(L)
Gesetz der Doppelnegation	$\neg \neg a = a$	(DN)

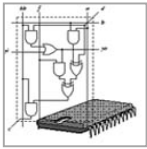


# Anwendung der Regeln



- Vereinfachung von Ausdrücken
  - Beispiel 1:  $Y = (A \wedge B \wedge C \wedge \neg D) \vee (A \wedge B) \vee \neg(\neg B \vee (A \wedge \neg A))$
  - Beispiel 2:  $Y = \neg((\neg A \vee \neg B \vee \neg C) \wedge (\neg D \vee \neg E \vee \neg F))$

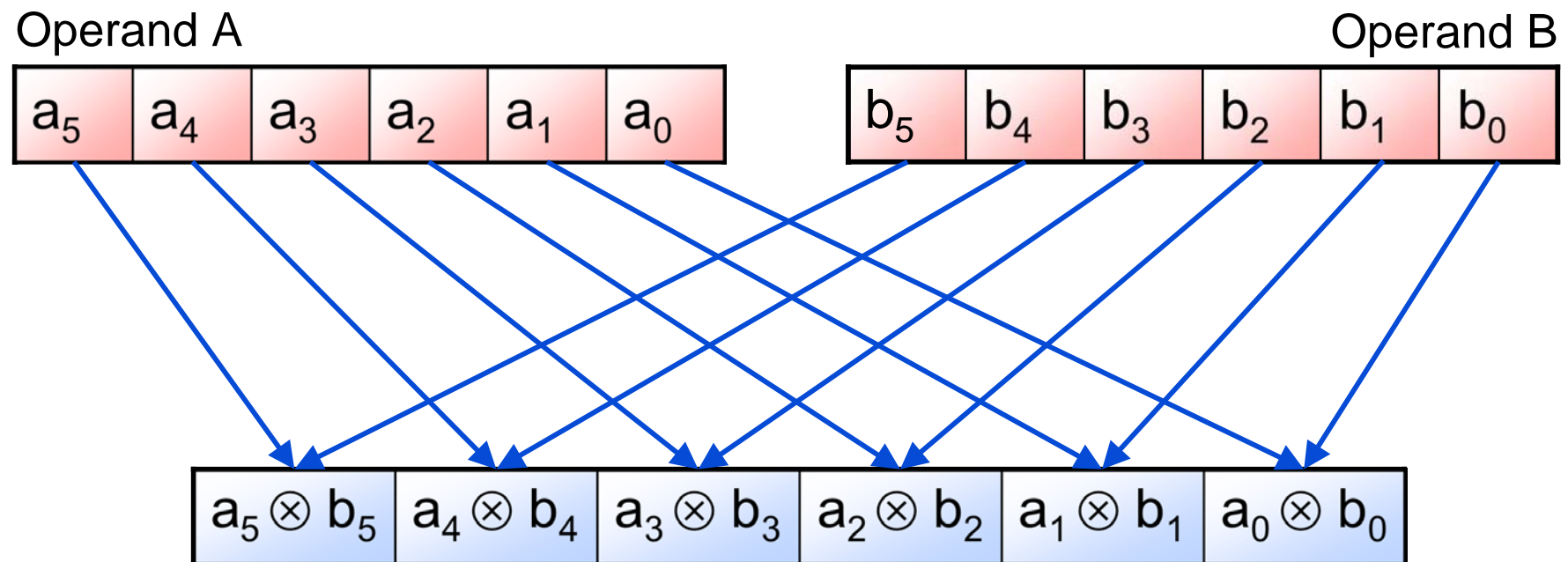




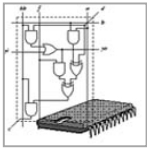
# Bitweise Logische Operationen



A, B seien Bitvektoren,  $\otimes$  eine beliebige Verknüpfung



Ergebnis  $E = A \otimes B$



# Bitweise Logische Operationen



- UND, ODER und XOR wirken wie spezielle Bit-Masken

$$\begin{array}{r} 1 \ 1 \ 0 \ 1 \ 0 \ 0 \\ \wedge \ 0 \ 1 \ 1 \ 1 \ 0 \ 0 \\ \hline = \ 0 \ 1 \ 0 \ 1 \ 0 \ 0 \end{array}$$

**UND** wird verwendet, um Bits gezielt auf **0** zu setzen. Dazu hat die Maske an allen Bitpositionen, die übernommen werden sollen, eine **1** und an den Stellen, die auf **0** gesetzt werden sollen, eine **0**.

$$\begin{array}{r} 1 \ 1 \ 0 \ 1 \ 0 \ 0 \\ \vee \ 0 \ 1 \ 1 \ 1 \ 0 \ 0 \\ \hline = \ 1 \ 1 \ 1 \ 1 \ 0 \ 0 \end{array}$$

**ODER** wird verwendet, um Bits gezielt auf **1** zu setzen. Dazu hat die Maske an allen Bitpositionen, die übernommen werden sollen, eine **0** und an den Stellen, die auf **1** gesetzt werden sollen, eine **1**.

$$\begin{array}{r} 1 \ 1 \ 0 \ 1 \ 0 \ 0 \\ \oplus \ 0 \ 1 \ 1 \ 1 \ 0 \ 0 \\ \hline = \ 1 \ 0 \ 1 \ 0 \ 0 \ 0 \end{array}$$

**XOR** wird verwendet, um Bits gezielt zu **kippen**. Dazu hat die Maske an allen Bitpositionen, die übernommen werden sollen, eine **0** und an den Stellen, die gekippt werden sollen, eine **1**.