Vorlesung 11 - Logik 2 am 26.10.2022

•

Logische Operationen

Konjunktion $\wedge$ AND

A	B	$A \wedge B$
w	w	w
w	f	f
f	w	f
f	f	f

Disjunktion $\vee$ OR

A	B	$A \vee B$
w	w	w
w	f	w
f	w	w
f	f	f

Negation $\neg$ NOT

A	$\neg A$
w	f
f	w

$\neg A \quad \bar{A}$

$\uparrow$

Sheffer-Operator | NAND

A	B	$A \wedge B$	$\neg(A \wedge B)$
f	f	f	w
f	w	f	w
w	f	f	w
w	w	w	f

Peirce-Operator $\downarrow$ NOR

A	B	$A \vee B$	$\neg(A \vee B)$
f	f	f	w
f	w	w	f
w	f	w	f
w	w	w	f

A	B	$\neg A$	$\neg A \vee B$
w	w	f	w
w	f	f	f
f	w	w	w
f	f	w	w

Exklusiv OR $\veebar$ XOR

A	B	A XOR B
f	f	f
f	w	w
w	f	w
w	w	f

$(A \wedge \neg B) \vee (\neg A \wedge B)$

$\Updownarrow$

Implikation $\Rightarrow$

A	B	$A \Rightarrow B$
w	w	w
w	f	f
f	w	w
f	f	w

Äquivalenz $\Leftrightarrow$

A	B	$A \Leftrightarrow B$
w	w	w
w	f	f
f	w	f
f	f	w

$(A \Rightarrow B) \wedge (B \Rightarrow A)$

$(\neg A \vee B) \wedge (\neg B \vee A)$

3.1.3 Logische Verbindungen

Logische Verbindungen (oder Aussagenverbindungen) entstehen durch Verkettung mehrerer Aussagen mittels logischer Operatoren.

Zur Vereinfachung der Schreibweise und zum Verzicht einer notwendigen Klammerung sind Vorrangregeln festgelegt. In der folgenden Reihenfolge bindet jede logische Operation stärker, d.h. wird zuerst ausgewertet:

- zuerst Negation,
- dann Konjunktion,
- dann Disjunktion,
- dann Implikation und
- dann Äquivalenz

Ist das Setzen von Klammern notwendig, so werden sie von innen nach außen interpretiert.

Beispiel

Die Aussageverbindung $\boxed{C := (A \wedge (B \Rightarrow \neg A)) \Rightarrow \neg B}$ wird durch Auswertung der einzelnen Komponenten über eine Wahrheitstafel ausgewertet.

A	B	$\neg A$	$\neg B$	$(B \Rightarrow \neg A)$	$A \wedge (B \Rightarrow \neg A)$	$C := (A \wedge (B \Rightarrow \neg A)) \Rightarrow \neg B$
w	w	f	f	f	f	w
w	f	f	w	w	w	w
f	w	w	f	w	f	w
f	f	w	w	w	f	w

f f f f

Die Aussageverbindung C ist also stets wahr.

Definition 3.4: Tautologie

stets wahr

Eine Aussage, die stets wahr ist, heißt **Tautologie**, z.B. $A \vee \neg A$ $= w$ (Satz vom ausgeschlossenen Dritten).

Tautologien liefern auch Vorgehensweisen zu Beweistechniken von Aussagen.

Definition 3.5: Kontradiktion

stets falsch

Eine Aussage, die stets falsch ist, heißt **Kontradiktion**, z.B. $A \wedge \neg A$ $= f$ (Satz vom ausgeschlossenen Widerspruch).

Dritten

Rangfolge bei der Auswertung von logischen Operatoren

https://de.wikipedia.org/wiki/Operatorrangfolge

Die Operatorrangfolge ist keine strikte Reihenfolge zwischen allen Operatoren geben muss. Es können auch mehrere Operatoren auf demselben Rang stehen.

Zum Beispiel ist in der Arithmetik der Rang von Multiplikation und Division gleich, aber höher als der Rang von Addition und Subtraktion („Punktrechnung vor Strichrechnung“).

Eine Klammerung bietet die Möglichkeit der Bevorrangung eines Teilstücks einer Formel: Der eingeklammerte, also von einem Klammerpaar „(...)“ eingeschlossene Bereich ist rechnerisch zuerst auszuführen und durch das entsprechende Teilergebnis zu ersetzen.

In der Logik ist es nicht immer üblich, eine Operatorrangfolge zu definieren.

Wo das geschieht, wird meistens (in absteigender Priorität) folgende gewählt:

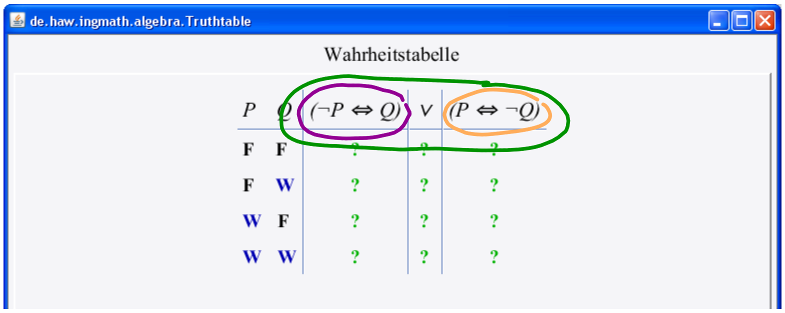
1. Negation
2. Konjunktion
3. Disjunktion
4. Implikation
5. Äquivalenz

Wahrheitswerte von Logischen Ausdrücke

über

(1) Wahrheitstabelle
(2) Vereinfachung mit Rechenregeln

Aufgabe: (1) Wahrheitstabelle aus letztem Vorlesungs-pdf



P	Q	$(\neg P \Leftrightarrow Q)$	$\vee$	$(P \Leftrightarrow \neg Q)$
F	F	?	?	?
F	W	?	?	?
W	F	?	?	?
W	W	?	?	?

P	Q	$\neg P$	$\neg Q$	$\neg P \Leftrightarrow Q$	$P \Leftrightarrow \neg Q$	$(\neg P \Leftrightarrow Q) \vee (P \Leftrightarrow \neg Q)$	
f	f	w	w	f	f	f	
f	w	w	f	w	w	w	
w	f	f	w	w	w	w	
w	w	f	f	f	f	f	

P XOR Q
$\Leftrightarrow$
$(P \wedge \neg Q) \vee (\neg P \wedge Q)$

(2) Lösung über Umformung

$$(\neg P \Leftrightarrow Q) \vee (P \Leftrightarrow \neg Q)$$

Auflösen Äquivalenz
$\Leftrightarrow \to \Rightarrow\Leftarrow$

$$\Rightarrow ((\neg P \Rightarrow Q) \wedge (Q \Rightarrow \neg P)) \vee ((P \Rightarrow \neg Q) \wedge (\neg Q \Rightarrow P))$$

Auflösen Implikation
$A \Rightarrow B \to \neg A \vee B$

$$\Rightarrow ((\neg(\neg P) \vee Q) \wedge (\neg Q \vee \neg P)) \vee ((\neg P \vee \neg Q) \wedge (\neg(\neg Q) \vee P))$$

doppelte Negation

$$\Rightarrow ((P \vee Q) \wedge (\neg Q \vee \neg P)) \vee ((\neg P \vee \neg Q) \wedge (Q \vee P))$$

Idempotenzgesetz
$A \vee A = A$
Kommutativgesetz
$A \vee B = B \vee A$
$A \wedge B = B \wedge A$

$$\Rightarrow \underbrace{(P \vee Q)}_{A} \wedge (\neg Q \vee \neg P)$$

Distributivgesetz

$$\Rightarrow ((\underbrace{P \vee Q}_{A}) \wedge \neg Q) \vee ((\underbrace{P \vee Q}_{A}) \wedge \neg P)$$

Distributivgesetz $\Rightarrow \big((P \land \neg Q) \lor \underbrace{(Q \land \neg Q)}_{f} \big) \lor \big(\underbrace{(P \land \neg P)}_{f} \lor (Q \land \neg P) \big)$

Satz vom ausgeschl. Widerspruch $\Rightarrow \big((P \land \neg Q) \lor f \big) \lor \big(f \lor (Q \land \neg P) \big)$

Dominanzgesetz $\Rightarrow (P \land \neg Q) \lor (Q \land \neg P)$

Definition XOR $\Rightarrow P \text{ XOR } Q$

DNF Disjunktive Normalform $(P \land \neg Q) \lor (\neg P \land Q)$

$\Updownarrow$

KNF Konjunktive Normalform $(P \lor Q) \land (\neg Q \lor \neg P)$

$\Updownarrow$

$P \text{ XOR } Q$

verschiedene Darstellungsformen des gleichen logischen Ausdrucks

Grundgesetze der Aussagenlogik

Satz 3.1: Grundgesetze der Aussagenlogik

Für beliebige logische Ausdrücke A, B, C gelten folgende Behauptungen:

(1) **Kommutativgesetz**

$\boxed{A \wedge B = B \wedge A}$ *und* $\boxed{A \vee B = B \vee A}$

(2) **Assoziativgesetz** für gleiche Operatoren

$(A \wedge B) \wedge C = A \wedge (B \wedge C)$ *und* $(A \vee B) \vee C = A \vee (B \vee C)$

(3) **Distributivgesetz** für verschiedene Operatoren

$A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$ *und* $A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$

„Ausklammern"

„Ausmultiplizieren"

(4) **Satz von der doppelten Negation:**

$\neg(\neg A) = A$

(5) **Idempotenzgesetze**

$A \wedge A = A$ *und* $A \vee A = A$

A	f	$A \vee f$
w	f	w
f	f	f

$= A$

(6) **Dominanzgesetz**

$A \wedge f = f,\ A \wedge w = A$ *und* $A \vee f = A,\ A \vee w = w$

A	w	$A \vee w$
w	w	w
f	w	w

$= w$

(7) **Satz vom ausgeschlossenen Dritten**

$A \wedge \neg A = f$ *und* $A \vee \neg A = w$ (... Widerspruch)

(8) **Satz von der Transitivität** Grundlage für direkten Beweis

$[(A \Rightarrow B) \wedge (B \Rightarrow C)] \Rightarrow (A \Rightarrow C)$

Wenn dann Wenn dann Wenn dann

(9) **Satz von der Kontraposition** Grundlage für indirekten Beweis

$(A \Rightarrow B) \Leftrightarrow (\neg B \Rightarrow \neg A)$

Ist eine Zahl durch 2 teilbar, dann ist sie gerade.
$\Leftrightarrow$
Ist eine Zahl nicht gerade, dann ist sie nicht durch 2 teilbar.

(10) **De Morganschen Regeln**

a) $\neg(A \wedge B) \Leftrightarrow \neg A \vee \neg B$

(Negation der Konjunktion ist die Disjunktion der negierten Elemente)

$\neg(A \vee B) \Leftrightarrow \neg A \wedge \neg B$

(Negation der Disjunktion ist die Konjunktion der negierten Elemente)

Vorgehen beim Vereinfachen logischer Ausdrücke

1. Implikationen und Äquivalenzen auflösen und durch Grundoperationen ersetzen
2. NAND, NOR, XOR auflösen und durch Grundoperationen ersetzen
3. De Morgansche Regeln anwenden
4. Rechengesetze geeignet anwenden

Beispiel

$$\neg(A \vee \neg(A \vee B)) \wedge (A \vee B)$$

de Morgansche Regel

$$= (\neg A \wedge \neg\neg(A \vee B)) \wedge (A \vee B)$$

doppelte Negation

$$= (\neg A \wedge (A \vee B)) \wedge (A \vee B)$$

Assoziativgesetz

$$= \neg A \wedge ((A \vee B) \wedge (A \vee B))$$

Idempotenzgesetz

$$= \neg A \wedge (A \vee B)$$

Distributivgesetz

$$= (\neg A \wedge A) \vee (\neg A \wedge B)$$

Satz vom ausgeschlossenen Dritten

$$= f \vee (\neg A \wedge B)$$

Dominanzgesetz

$$= \boxed{\neg A \wedge B}$$

Vereinfachter Ausdruck

Lösung: Weiteres Beispiel als Aufgabe
Formen Sie den logischen Ausdruck mit Hilfe der Rechenregeln um!

Beispiel

Die Aussageverbindung $\boxed{C := (A \wedge (B \Rightarrow \neg A)) \Rightarrow \neg B}$ wird durch Auswertung der einzelnen Komponenten über eine Wahrheitstafel ausgewertet.

A	B	$\neg A$	$\neg B$	$(B \Rightarrow \neg A)$	$A \wedge (B \Rightarrow \neg A)$	$C := (A \wedge (B \Rightarrow \neg A)) \Rightarrow \neg B$
w	w	f	f	f	f	w
w	f	f	w	w	w	w
f	w	w	f	w	f	w
f	f	w	w	w	f	w

Die Aussageverbindung C ist also stets wahr.

$$(A \wedge (B \Rightarrow \neg A)) \Rightarrow \neg B$$

$$= \neg(A \wedge (\neg B \vee \neg A)) \vee \neg B$$

Auflösen der Implikationen

$$= (\neg A \vee \neg(\neg B \vee \neg A)) \vee \neg B$$

deMorgansche Regel

$$= (\neg A \vee (\neg\neg B \wedge \neg\neg A)) \vee \neg B$$

$$= (\neg A \vee (B \wedge A)) \vee \neg B$$

doppelte Negation

$$= ((\neg A \vee B) \wedge (\neg A \vee A)) \vee \neg B$$

Distributivgesetz

$$= ((\neg A \vee B) \wedge w) \vee \neg B$$

Satz vom ausgeschlossenen Dritten

$$= (\neg A \vee B) \vee \neg B$$

Dominanzgesetz

$$= \neg A \vee (B \vee \neg B)$$

Assoziativgesetz

$$= \neg A \vee w$$

Satz vom ausgeschlossenen Dritten

$$= w$$

Dominanzgesetz

Vorgehen bei der Auflösung logischer Ausdrücke:

1. Implikationen und Äquivalenzen auflösen und durch Grundoperationen ersetzen
2. NAND, NOR, XOR auflösen und durch Grundoperationen ersetzen
3. De Morgansche Regeln anwenden
4. Rechengesetze geeignet anwenden

Satz 3.2

Jeder logische Ausdruck kann durch einen Ausdruck ersetzt werden, der nur die Operatoren $\neg, \wedge, \vee$ enthält.

Zum Beweis:

Durch die Richtigkeit der nachfolgenden Aussagen kann gezeigt werden, dass dieses möglich ist.

- XOR: $A \text{ XOR } B = (A \wedge \neg B) \vee (\neg A \wedge B)$
- Implikation: $A \Rightarrow B = \neg A \vee B$
- Äquivalenz: $A \Leftrightarrow B = (A \Rightarrow B) \wedge (B \Rightarrow A) = (\neg A \vee B) \wedge (A \vee \neg B)$
- NOR: $A \downarrow B = \neg(A \vee B)$
- NAND: $A \mid B = \neg(A \wedge B)$

Satz 3.3

Die drei booleschen Grundoperationen $\neg, \wedge, \vee$ können als Hintereinanderausführung von ausschließlich NAND-Funktionen oder ausschließlich NOR-Funktionen geschrieben werden.

Zum Beweis für NAND:

$$A \wedge B \overset{①}{=} (A \wedge B) \vee f \overset{②}{=} \neg(\neg(A \wedge B) \wedge w) \overset{③}{=} (A \text{ NAND } B) \text{ NAND } w$$

$$A \vee B \overset{①}{=} (A \wedge w) \vee (B \wedge w) \overset{②}{=} \neg(\neg(A \wedge w) \wedge \neg(B \wedge w)) \overset{③}{=} (A \text{ NAND } w) \text{ NAND } (B \text{ NAND } w)$$

$$\neg A \overset{①}{=} \neg(A \wedge w) = A \text{ NAND } w$$

① Dominanzgesetz

② doppelte Negation
und einmal DeMorgansche Regel

③ Definition von NAND

Wofür braucht man eine Realisierung von logischen Ausdrücken nur mit NAND oder NOR?

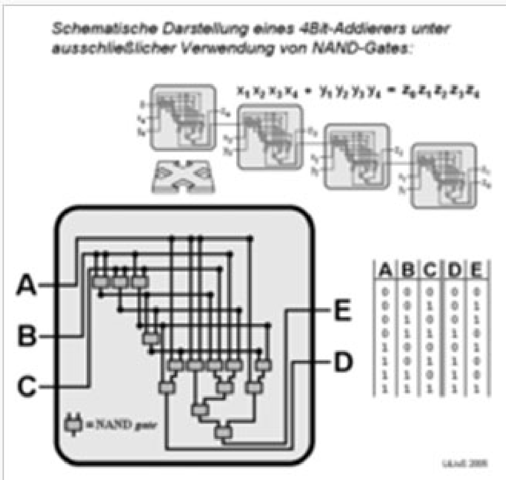
NAND-Gatter

Ein **NAND-Gatter** (von englisch: **n**ot **and** – *nicht und*) ist ein Logikgatter mit zwei oder mehr Eingängen x, y, … und einem Ausgang Q, zwischen denen die logische Verknüpfung **NICHT UND** besteht. Ein NAND-Gatter gibt am Ausgang nur dann 0 aus, wenn alle Eingänge 1 sind, oder gibt nur dann 1 aus, wenn mindestens ein Eingang 0 ist.

http://de.wikipedia.org/wiki/NAND-Gatter

Verwendung [Bearbeiten]

NAND-Gatter spielen in der Digitaltechnik die Rolle eines Standardbausteins, da sich allein mit ihnen alle logischen Verknüpfungen und somit auch komplexere Schaltungen (wie Addierer, Multiplexer usw.) zusammenstellen lassen. Dadurch, dass sich mit diesem Baustein alle anderen ersetzen lassen, wird eine Schaltung wesentlich preisgünstiger. Dies liegt daran, dass ein sogenanntes IC immer mehrere Gatter enthält und demzufolge eine bestimmte Anzahl Ein- und Ausgänge bereitstellen kann. Soll etwa ein Eingangssignal lediglich negiert werden, muss kein neues IC benutzt werden, sondern man legt die Eingangspins (Anschlüsse) zusammen, so dass nur noch ein Eingang zur Verfügung steht. Damit ist ein Nicht-Gatter entstanden. Mit einer geringeren IC-Anzahl können also Schaltungen umgesetzt werden, da die Hardware-Bausteine komplett ausgenutzt werden können.



Schematische Darstellung eines 4Bit-Addierers unter ausschließlicher Verwendung von NAND-Gattern

Logische Verknüpfungen und deren Umsetzung mittels NAND-Gattern:

Verknüpfung	Umsetzung
NOT x	x NAND x
x AND y	(x NAND y) NAND (x NAND y)
x NAND y	x NAND y
x OR y	(x NAND x) NAND (y NAND y)
x NOR y	((x NAND x) NAND (y NAND y)) NAND ((x NAND x) NAND (y NAND y))
x XOR y	(x NAND (y NAND y)) NAND ((x NAND x) NAND y) ((x NAND y) NAND y) NAND ((x NAND y) NAND x)
x XNOR y	(x NAND y) NAND ((x NAND x) NAND (y NAND y)) ≡ x ⇔ y
x ⇒ y	x NAND (y NAND y)
x ⇐ y	(x NAND x) NAND y
x ⇔ y	(x NAND y) NAND ((x NAND x) NAND (y NAND y)) ≡ x XNOR y
verum	(x NAND x) NAND x
falsum	((x NAND x) NAND x) NAND ((x NAND x) NAND x)

Logische Verknüpfungen haben ihre Entsprechung im Schaltungsaufbau:

- $\wedge$ entspricht einer Reihenschaltung von zwei Schaltern
- $\vee$ entspricht einer Serienschaltung von zwei Schaltern

wahr bedeutet "geschlossener Schalter"
falsch bedeutet "offener Schalter"
Negation bedeutet die Umkehr offen <-> geschlossen

Verknüpfung	Schaltzeichen	Funktionsgleichnung	Funktionstabelle	Prinzipschaltung
NOT (NICHT) Negation	A –[1]o– Q	$Q=\bar{A}$ $= \neg A$	A Q 0 1 1 0	Öffner
AND (UND) Konjunktion	A, B –[&]– Q	$Q=A\wedge B$	A B Q 0 0 0 0 1 0 1 0 0 1 1 1	Schließer in Reihenschaltung
OR (ODER) Disjunktion	A, B –[≥1]– Q	$Q=A\vee B$	A B Q 0 0 0 0 1 1 1 0 1 1 1 1	Schließer in Parallelschaltung
NAND (NICHT-UND)	A, B –[&]o– Q	$Q=\overline{A\wedge B}$ $= \neg(A\wedge B)$	A B Q 0 0 1 0 1 1 1 0 1 1 1 0	Öffner in Reihenschaltung
NOR (NICHT-ODER)	A, B –[≥1]o– Q	$Q=\overline{A\vee B}$ $= \neg(A\vee B)$	A B Q 0 0 1 0 1 0 1 0 0 1 1 0	Öffner in Parallelschaltung
XOR (Exklusiv-ODER) Antivalenz	A, B –[=1]– Q	$Q=(\bar{A}\wedge B)\vee(A\wedge\bar{B})$	A B Q 0 0 0 0 1 1 1 0 1 1 1 0	Wechsler in Parallelschaltung
XNOR (Exklusiv-NICHT-ODER) Äquivalenz	A, B –[=1]o– Q	$Q=(A\wedge B)\vee(\bar{A}\wedge\bar{B})$	A B Q 0 0 1 0 1 0 1 0 0 1 1 1	Wechsler in Reihenschaltung

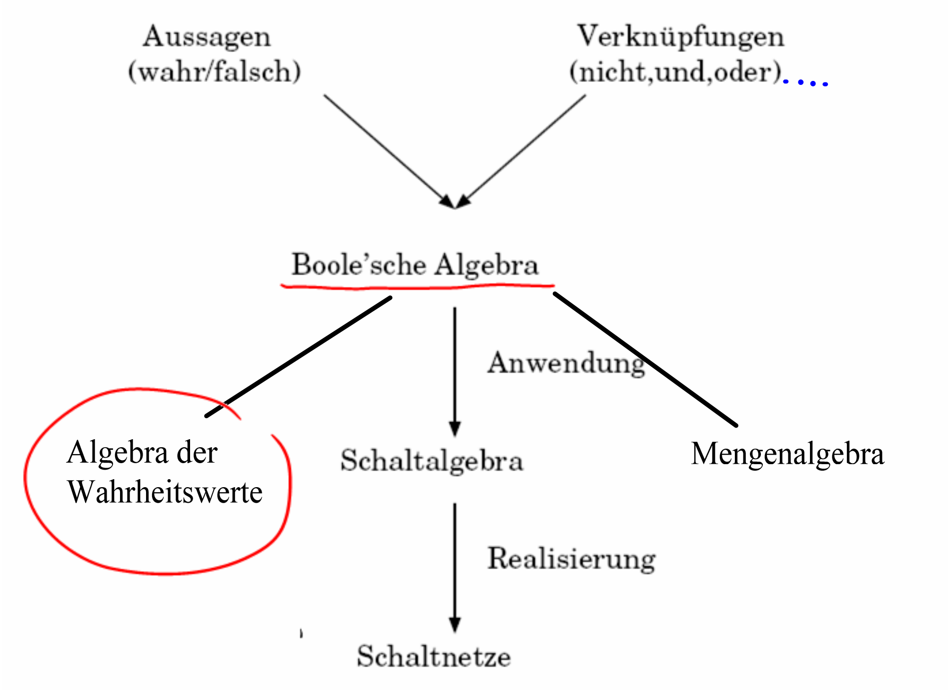
$A=1$, $B=1$ } wenn Schalter geöffnet wird

Abbildungen sind vertauscht

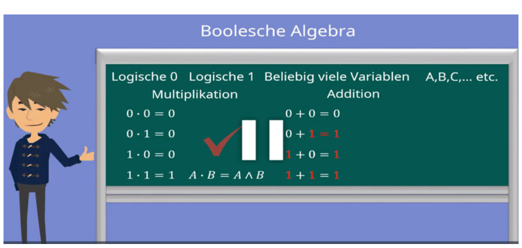
$\wedge$ und, $\vee$ oder, ¯ über der Aussage bedeutet Negation der Aussage, ∘ beim Ausgangsglied weist auf Negation hin

www.tafelwerk-interaktiv.de

Zusammenhänge "Boolesche Algebra"



Operation	Boolesche Algebra	Aussagenlogik	Mengenlehre	Schaltzeichen	Kontaktschalter	Wahrheitstabelle
UND	* (Boolesches Produkt) oder ·	∧ (Konjunktion)	∩	&	— x — y — Reihenschaltung	x y x*y 0 0 0 0 1 0 1 0 0 1 1 1
ODER	+ (Boolesche Summe) oder #	∨ (Disjunktion)	∪	≥1	x y Parallelschaltung	x y x+y 0 0 0 0 1 1 1 0 1 1 1 1
NICHT	x' (Boolesches Komplement)	¬ (Negation)	G \ A	1	— $\bar{x}$ — Ruhekontakt	x x' 0 1 1 0



https://studyflix.de/informatik/boolesche-algebra-977

3.3 Boolesche Algebra

Definition 3.11: Boolesche Algebra

Eine **Boolesche Algebra** (A,#, *, ¬ ,0,1) ist eine algebraische Struktur, die aus einer nicht leeren Menge A besteht, für deren Elemente folgende Operationen erklärt sind:

2-stellig:	$*$	:	$A \times A \to A$	(Boolesches Produkt)
2-stellig:	$\#$	:	$A \times A \to A$	(Boolesche Summe)
1-stellig:	$\neg$	:	$A \to A$	(Boolesches Komplement)
0-stellig:	0,1	:	A	(feste Elemente aus A mit $0 \neq 1$)

so dass für alle x; y; z ∊ A die folgenden Gleichungen (Axiome) gelten:

(1) Kommutativität	$x * y = y * x$	$x \# y = y \# x$
(2) Distributivität	$x * (y \# z) = (x * y) \# (x * z)$	$x \# (y * z) = (x \# y) * (x \# z)$
(3) neutrales Element	$x * 1 = x$	$x \# 0 = x$
(4) komplementäre Elemente (Negation)	$x * \neg x = 0$	$x \# \neg x = 1$

Bemerkung:

- Eine **algebraische Struktur** ist eine Menge mit auf ihr definierten Verknüpfungen, deren Ergebnisse wieder in dieser Menge liegen.

Gesetze der Booleschen Algebra

Satz 3.5: Gesetze einer Booleschen Algebra

Folgende **Sätze** gelten für eine Boolesche Algebra:

(1) **Idempotenzgesetze**

Die Operatoren #, * sind idempotent,
d.h. für alle $x \in A$ gilt $x \# x = x$ und $x * x = x$.

(2) **Dominanzgesetz**

Für alle $x \in A$ gilt: $x \# 1 = 1$ und $x * 0 = 0$

(3) **Absorptionsgesetze**

Für alle $x, y \in A$ gilt: $x * (x \# y) = x$ und $x \# (x * y) = x$

(4) **Assoziativgesetze**

Für alle $x, y, z \in A$ gilt: $x * (y * z) = (x * y) * z$ $\quad$ $x \# (y \# z) = (x \# y) \# z$

(5) **DeMorgansche Gesetze**

Für alle $x, y \in A$ gilt: $\neg(x \# y) = \neg x * \neg y$ und $\neg(x * y) = \neg x \# \neg y$.

(6) **Doppelte Komplementbildung**

Für alle $x, y \in A$ gilt: $\neg\neg x = x$

(7) **Komplementarität der neutralen Elemente**

Die neutralen Elemente (0, 1) sind wechselseitig komplementär.
Es gilt: $\neg 0 = 1$ und $\neg 1 = 0$

Beispiele für eine Boolesche Algebra

Beispiele: Boolesche Algebra - Potenzmengenalgebra (Mengenalgebra)

Satz 3.6:

Ist E eine beliebige Menge und P(E) die Potenzmenge von E, so machen die Operationen

Vereinigung $\cup$	als	#	
Schnitt $\cap$	als	*	
Komplement $\overline{}$	als	$\neg$	mit $\overline{M} = E \setminus M = \{x \in E : x \notin M\}$

sowie die Konstanten

leere Menge $\emptyset$	als	0	(neutrales Element der Vereinigung)
Menge E	als	1	(neutrales Element der Schnittbildung)

die Potenzmenge P(E) zu einer Booleschen Algebra (P(E), $\cup, \cap, \neg, \emptyset$, E).

Beispiele: Boolesche Algebra - Algebra der Wahrheitswerte (Aussagenalgebra)

Satz 3.7:

Ist B die Menge der Wahrheitswerte $\{w, f\}$, so machen die Operationen

Disjunktion $\vee$	als	#
Konjunktion $\wedge$	als	*
Komplement $\neg$	als	$\neg$

sowie die logischen Konstanten

falsch f	als	0	(neutrales Element der Disjunktion)
wahr w	als	1	(neutrales Element der Konjunktion)

die Menge B zu einer Booleschen Algebra (B, $\vee, \wedge, \neg, w, f$).

Beispiele: Boolesche Algebra - Schaltalgebra

Satz 3.8:

Sei B die Menge $\{oS, gS\}$ mit *oS* := „offener Schalter" und *gS*:= „geschlossener Schalter" und den Operationen

Parallelschaltung *P*	als	#
Serienschaltung *S*	als	*

und der Komplementbildung Komplement $\overline{oS} = gS \;\; und \;\; \overline{gS} = oS$

sowie die logischen Konstanten

oS	als	0	(neutrales Element der Parallelschaltung)
gS	als	1	(neutrales Element der Serienschaltung)

so gelten die Axiome der Booleschen Algebra. Diese Algebra wird Schaltalgebra genannt.

Operation	Boolesche Algebra	Aussagenlogik	Mengenlehre	Schaltzeichen	Kontaktschalter	Wahrheitstabelle
UND	* (Boolesches Produkt)	$\wedge$ (Konjunktion)	$\cap$	&	— x — y — Reihenschaltung	x y x*y 0 0 0 0 1 0 1 0 0 1 1 1
ODER	+ (Boolesche Summe)	$\vee$ (Disjunktion)	$\cup$	≥1	x y Parallelschaltung	x y x+y 0 0 0 0 1 1 1 0 1 1 1 1
NICHT	x' (Boolesches Komplement)	$\neg$ (Negation)	G \ A	1	— $\bar{x}$ — Ruhekontakt	x x' 0 1 1 0

Anwendung: Schaltnetze

Beispiel:

Beschreibung einer Schaltfunktion

Gesamtzusammenhänge

- Wertetabelle

x_0	x_1	x_2	$y = f(x_0, x_1, x_2)$
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

3

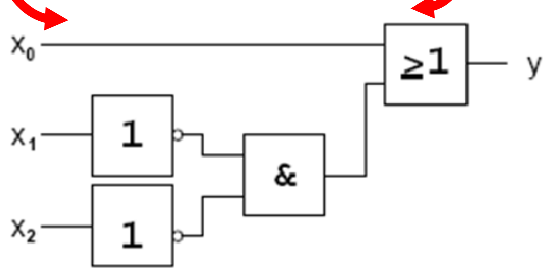
- Funktionsgleichung

2

$$y = x_0 \vee (\bar{x}_1 \wedge \bar{x}_2)$$

1

- Schaltnetz



http://www.uni-koblenz.de/~physik/informatik

Aufgabe: 2
Aus einer gegebenen Schaltung eine Wahrheits-tafel erstellen!

Aufgabe: 3
Aus einer gegebenen Wahrheitstafel eine logische Formel erstellen!

Aufgabe: 1
Aus einer gegebenen logischen Formel ein Schaltnetz erstellen!

Aufgabe: 1
Aus einem gegebenen Schaltnetz eine logische Formel ableiten!

Anwendung: Schaltnetze

Operator	Schaltzeichen	Benennung	Beispiel-IC
$\overline{}$	1	NOT	7404
$\wedge$	&	AND	7408
$\vee$	$\geq$	OR	7432

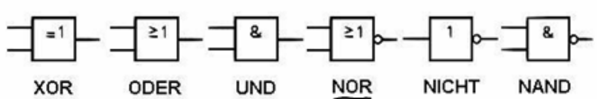
http://www.uni-koblenz.de/~physik/informatik

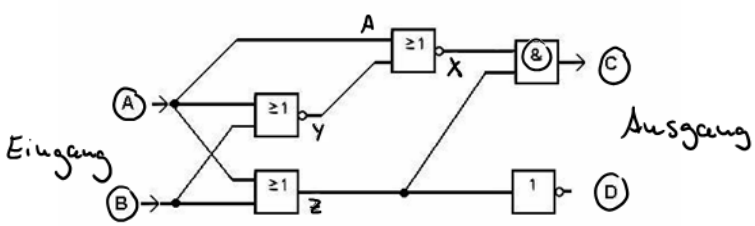
Beispiel:

Aufgabe: 1
Aus einem gegebenen Schaltnetz eine logische Formel ableiten!

Beispiel: Anwendung Schaltkreise

- Schaltkreise haben n Eingänge und m Ausgänge.
- Jeder Eingang (Ausgang) kann entweder wahr oder falsch sein.
- Ausgänge hängen logisch von den Eingängen ab.
- Schaltkreise werden aus logischen Komponenten zusammengesetzt:





$C = X \wedge Z$

$= (A \text{ NOR } Y) \wedge Z$

$= (A \text{ NOR } (A \text{ NOR } B)) \wedge Z$

$= (A \text{ NOR } (A \text{ NOR } B)) \wedge (A \vee B)$

$= \neg(A \vee (A \text{ NOR } B)) \wedge (A \vee B)$

$= \neg(A \vee \neg(A \vee B)) \wedge (A \vee B)$

⋮ Umformungen

$= \neg A \wedge B$

$D = \neg Z$

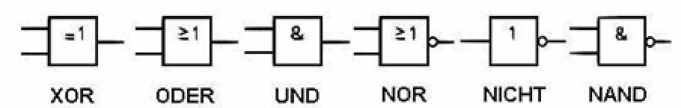
$= \neg(A \vee B)$

$= \neg A \wedge \neg B$

Anwendung: Schaltnetze

Beispiel:

Beschreibung einer Schaltfunktion



- Wertetabelle

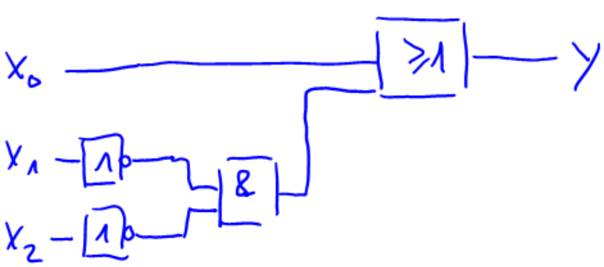
Eingänge

- Funktionsgleichung

$$\boxed{y = x_0 \vee (\bar{x}_1 \wedge \bar{x}_2)} = X_0 \vee (\neg X_1 \wedge \neg X_2)$$

Ausgang

- Schaltnetz



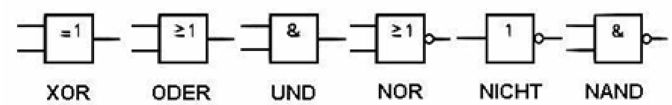
Aufgabe: 1
Aus einer gegebenen logischen Formel ein Schaltnetz erstellen!

http://www.uni-koblenz.de/~physik/informatik

Anwendung: Schaltnetze

Beispiel:

Beschreibung einer Schaltfunktion



- Wertetabelle

x_0	x_1	x_2	$y = f(x_0, x_1, x_2)$
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

- Funktionsgleichung

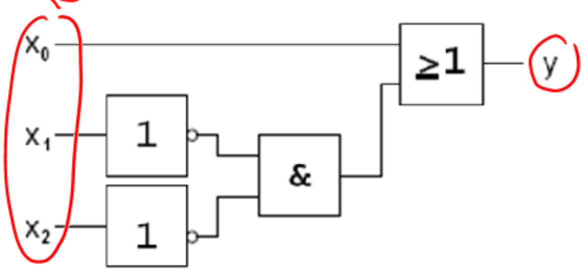
siehe vorher

Messen

logische Formel

- Schaltnetz

1



Aufgabe: 2
Aus einer gegebenen Schaltung eine Wahrheitstafel erstellen!

http://www.uni-koblenz.de/~physik/informatik

Anwendung: Schaltnetze

Beispiel:

Beschreibung einer Schaltfunktion

- Wertetabelle

x_0	x_1	x_2	$y = f(x_0, x_1, x_2)$
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

?

DNF/KNF

(auch später in Digitaltechnik)

- Funktionsgleichung

- Schaltnetz

Aufgabe: 3
Aus einer gegebenen Wahrheitstafel eine logische Formel erstellen!

http://www.uni-koblenz.de/~physik/informatik

3.1.5 Normalformen

Unterschiedliche Formen der gleichen Aussage erschweren oft die Lesbarkeit und führen zum Wunsch nach Standardformen. Normalformen sind für die Entwicklung logischer Schaltkreise wichtig. Insbesondere konjunktive Normalformen spielen in der logischen Programmierung eine Rolle.

Definition 3.7: Literal

Ein **Literal** ist ein Ausdruck, der entweder aus einer einzelnen logischen Aussagenvariable (atomare Aussage) oder ihrer Negation besteht.

A oder $\neg A$

Definition 3.8: disjunktive Normalform (DNF)

Ein logischer Ausdruck befindet sich in **disjunktiver Normalform**, wenn er ausschließlich Disjunktionen von Konjunktionen von Literalen enthält.

$$K_1 \vee K_2 \vee K_3 \vee \ldots \vee K_n \quad mit \quad K_i = L_{i1} \wedge L_{i2} \wedge \ldots \wedge L_{im_i} \quad L_{ij} \; Literal$$

Klammern mit $\wedge$-Verknüpfungen

Definition 3.9: konjunktive Normalform (KNF)

Ein logischer Ausdruck befindet sich in **konjunktiver Normalform**, wenn er ausschließlich Konjunktionen von Disjunktionen von Literalen enthält.

$$D_1 \wedge D_2 \wedge D_3 \wedge \ldots \wedge D_n \quad mit \quad D_i = L_{i1} \vee L_{i2} \vee \ldots \vee L_{im_i}, \quad L_{ij} \; Literal$$

Klammern mit $\vee$-Verknüpfungen

Beispiele:

(1) $A, \neg A, w, f$ sind Beispiele für Literale

(2) Der Ausdruck $(A \vee B) \wedge (\neg A \vee B \vee w)$ ist in konjunktiver Normalform.

(3) Der Ausdruck $(A \wedge B) \vee (\neg A \wedge C) \vee C$ ist in disjunktiver Normalform.

Satz 3.4:

Zu jeder aussagenlogischen Verbindung gibt es (mindestens) eine logisch äquivalente disjunktive Normalform und (mindestens) eine logisch äquivalente konjunktive Normalform.

DNF

KNF

Umwandlung in Normalform:

(1) Systematische Umwandlung in eine KNF/ DNF in 3 Schritten:

(1) Elimination von $\Rightarrow$ mittels $A \Rightarrow B \;=\; \neg A \vee B$ und
$\Leftrightarrow$ mittels $A \Leftrightarrow B \;=\; (\neg A \vee B) \wedge (A \vee \neg B)$

(2) Verteilung von $\neg$ auf atomare Ausdrücke mittels DeMorgansche Regeln:
$\neg(A \wedge B) \Leftrightarrow \neg A \vee \neg B$, $\neg(A \vee B) \Leftrightarrow \neg A \wedge \neg B$ sowie $\neg(\neg A) = A$

(3) Umwandlung in eine Konjunktion von Disjunktionen (KNF) bzw. Disjunktion von Konjunktionen (DNF) mittels Distributivgesetze: $A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$,
$A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$

Beispiel:

$A \Rightarrow \neg(B \Rightarrow C)$ *Umwandlung in konjunktive Normalform*

Implikat. auflösen — *Schritt 1:* $= \neg A \vee \neg(\neg B \vee C)$

DeMorgan + doppelte Neg — *Schritt 2:* $= \neg A \vee (\neg\neg B \wedge \neg C)$

$= \neg A \vee (B \wedge \neg C)$ ist DNF

Distributivgesetze — *Schritt 3:* $= (\neg A \vee B) \wedge (\neg A \vee \neg C)$ ist KNF

(2) Bestimmung von KNF/ DNF aus einer Wahrheitstafel

Ist eine Wahrheitstafel mit Eingangsvariablen $x_1, x_2, \ldots, x_n$ und einer Ausgangsvariablen $f(x_1, x_2, \ldots, x_n)$ gegeben, so kann aus ihr direkt die KNF bzw. DNF für $f(x_1, x_2, \ldots, x_n)$ abgelesen werden:

DNF:

- Zu jeder Zeile der Wahrheitstafel, die den Funktionswert *w* liefert, konstruiert man einen Booleschen Ausdruck, der nur die Operationen $\wedge$ und $\neg$ enthält und genau für die Eingabedaten dieser Zeile den Wert *w* annimmt.
- Die so erhaltenen Ausdrücke werden dann mit $\vee$ verknüpft.

KNF:

- Zu jeder Zeile der Wahrheitstafel, die den Funktionswert *f* liefert, konstruiert man einen Booleschen Ausdruck, der nur die Operationen $\vee$ und $\neg$ enthält und genau für die Eingabedaten dieser Zeile den Wert *f* annimmt.
- Die so erhaltenen Ausdrücke werden dann mit $\wedge$ verknüpft.

Beispiel:

A	*B*	*C*	*f(A,B,C)*	DNF	KNF
w	w	w	w	$(A \wedge B \wedge C) \vee$	
w	w	f	w	$(A \wedge B \wedge \neg C) \vee$	
w	f	w	f		$(\neg A \vee B \vee \neg C) \wedge$
w	f	f	f		$(\neg A \vee B \vee C) \wedge$
f	w	w	f		$(A \vee \neg B \vee \neg C) \wedge$
f	w	f	w	$(\neg A \wedge B \wedge \neg C) \vee$	
f	f	w	w	$(\neg A \wedge \neg B \wedge C)$	
f	f	f	f		$(A \vee B \vee C)$

DNF: $f(A,B,C) = (A \wedge B \wedge C) \vee (A \wedge B \wedge \neg C) \vee (\neg A \wedge B \wedge \neg C) \vee (\neg A \wedge \neg B \wedge C)$

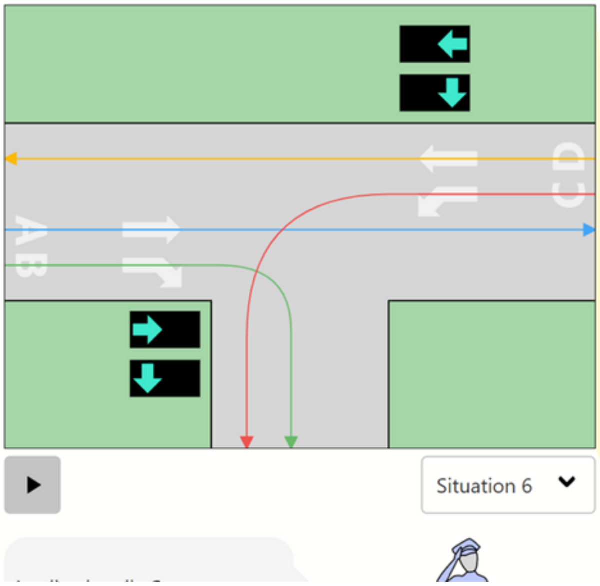
KNF: $f(A,B,C) = (\neg A \vee B \vee \neg C) \wedge (\neg A \vee B \vee C) \wedge (A \vee \neg B \vee \neg C) \wedge (A \vee B \vee C)$

Da in diesen Teilausdrücken jeweils alle Booleschen Variablen vorkommen, nennt man diese Darstellungen eine **kanonische DNF bzw. KNF**.

Bemerkung:

- Mit der beschriebenen Ermittlung von DNF und KNF aus der Wahrheitstafel bekommt man nicht immer die kürzeste Schreibweise. Für eine kürzere Schreibweise wird die Ermittlung von DNF und KNF über Karnaugh-Diagramme verwendet. Diese Methode soll hier aber nicht weiter vertieft werden.

Einführung konjunktive/ disjunktive Normalform



A	B	C	D	sicher
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

optimal

$(\neg A \wedge \neg B \wedge \neg C \wedge \neg D) \vee (\neg A \wedge \neg B \wedge \neg C \wedge D) \vee (\neg A \wedge \neg B \wedge C \wedge \neg D) \vee (\neg A \wedge \neg B \wedge C \wedge D) \vee (\neg A \wedge B \wedge \neg C \wedge \neg D) \vee (\neg A \wedge B \wedge \neg C \wedge D) \vee (A \wedge \neg B \wedge \neg C \wedge \neg D) \vee (A \wedge \neg B \wedge \neg C \wedge D) \vee (A \wedge B \wedge \neg C \wedge \neg D) \vee (A \wedge B \wedge \neg C \wedge D)$

KDNF | In Formeleditor laden

$(\neg B \vee \neg C) \wedge (\neg A \vee \neg C)$

KNF | In Formeleditor laden

$(\neg A \wedge \neg B \wedge \neg C \wedge \neg D) \vee (\neg A \wedge \neg B \wedge \neg C \wedge D) \vee (\neg A \wedge \neg B \wedge C \wedge \neg D) \vee (\neg A \wedge \neg B \wedge C \wedge D) \vee (\neg A \wedge B \wedge \neg C \wedge \neg D) \vee (\neg A \wedge B \wedge \neg C \wedge D) \vee (A \wedge \neg B \wedge \neg C \wedge \neg D) \vee (A \wedge \neg B \wedge \neg C \wedge D) \vee (A \wedge B \wedge \neg C \wedge \neg D) \vee (A \wedge B \wedge \neg C \wedge D)$

KDNF | In Formeleditor laden

$(\neg A \wedge \neg B) \vee (\neg C)$

DNF | In Formeleditor laden

$C \rightarrow (\neg A \wedge \neg B)$

Implikation | In Formeleditor laden

Weiteres Beispiel für das Ableiten einer DNF/ KNF

A	B	C	Ergebnis
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

→ zu Hause

http://de.wikipedia.org/wiki/Disjunktive_Normalform

Weiteres Beispiel für das Ableiten einer DNF/ KNF

A	B	C	Ergebnis	Klausel
0	0	0	0	$A \vee B \vee C$
0	0	1	0	$A \vee B \vee \neg C$
0	1	0	1	$\neg A \wedge B \wedge \neg C$
0	1	1	1	$\neg A \wedge B \wedge C$
1	0	0	0	$\neg A \vee B \vee C$
1	0	1	1	$A \wedge \neg B \wedge C$
1	1	0	0	$\neg A \vee \neg B \vee C$
1	1	1	1	$A \wedge B \wedge C$

$\wedge$

$\vee$

DNF: $(\neg A \wedge B \wedge \neg C) \vee (\neg A \wedge B \wedge C) \vee (A \wedge \neg B \wedge C) \vee (A \wedge B \wedge C)$

KNF: $(A \vee B \vee C) \wedge (A \vee B \vee \neg C) \wedge (\neg A \vee B \vee C) \wedge (\neg A \vee \neg B \vee C)$

http://de.wikipedia.org/wiki/Disjunktive_Normalform

Anwendung: Schaltnetze

Beispiel:

Beschreibung einer Schaltfunktion

- Wertetabelle

x_0	x_1	x_2	$y = f(x_0, x_1, x_2)$
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

zu Hause

- Funktionsgleichung

- Schaltnetz

Aufgabe:
Aus einer gegebenen Wahrheitstafel eine logische Formel mit Hilfe DNF/KNF erstellen!

http://www.uni-koblenz.de/~physik/informatik

Anwendung: Schaltnetze

Aufgabe 3 für die Übungen

Beispiel:

Beschreibung einer Schaltfunktion

Beschreibung einer Schaltfunktion

Darstellungen können mit Rechengesetzen umgeformt werden

- Wertetabelle

Stellt die komplette Wahrheitstafel dar

Stellt die komplette Wahrheitstafel dar

KNF: $y = (x_0 \lor x_1 \lor \neg x_2) \land (x_0 \lor \neg x_1 \lor x_2) \land (x_0 \lor \neg x_1 \lor \neg x_2)$

DNF: $y = (\neg x_0 \land \neg x_1 \land \neg x_2) \lor (x_0 \land \neg x_1 \land \neg x_2) \lor (x_0 \land \neg x_1 \land x_2)$
$\lor (x_0 \land x_1 \land \neg x_2) \lor (x_0 \land x_1 \land x_2)$

x_0	x_1	x_2	$y = f(x_0, x_1, x_2)$	
0	0	0	1	→ DNF
0	0	1	0	→ KNF $(x_0 \lor x_1 \lor \neg x_2)$
0	1	0	0	→ KNF $(x_0 \lor \neg x_1 \lor x_2)$
0	1	1	0	→ KNF $(x_0 \lor \neg x_1 \lor \neg x_2)$
1	0	0	1	→ DNF
1	0	1	1	→ DNF
1	1	0	1	→ DNF
1	1	1	1	→ DNF

- Funktionsgleichung

Siehe Vorlesung

$\boxed{y = x_0 \lor (\neg x_1 \land \neg x_2)}$

- Schaltnetz

Aufgabe:
Aus einer gegebenen Wahrheitstafel eine logische Formel mit Hilfe DNF/KNF erstellen!