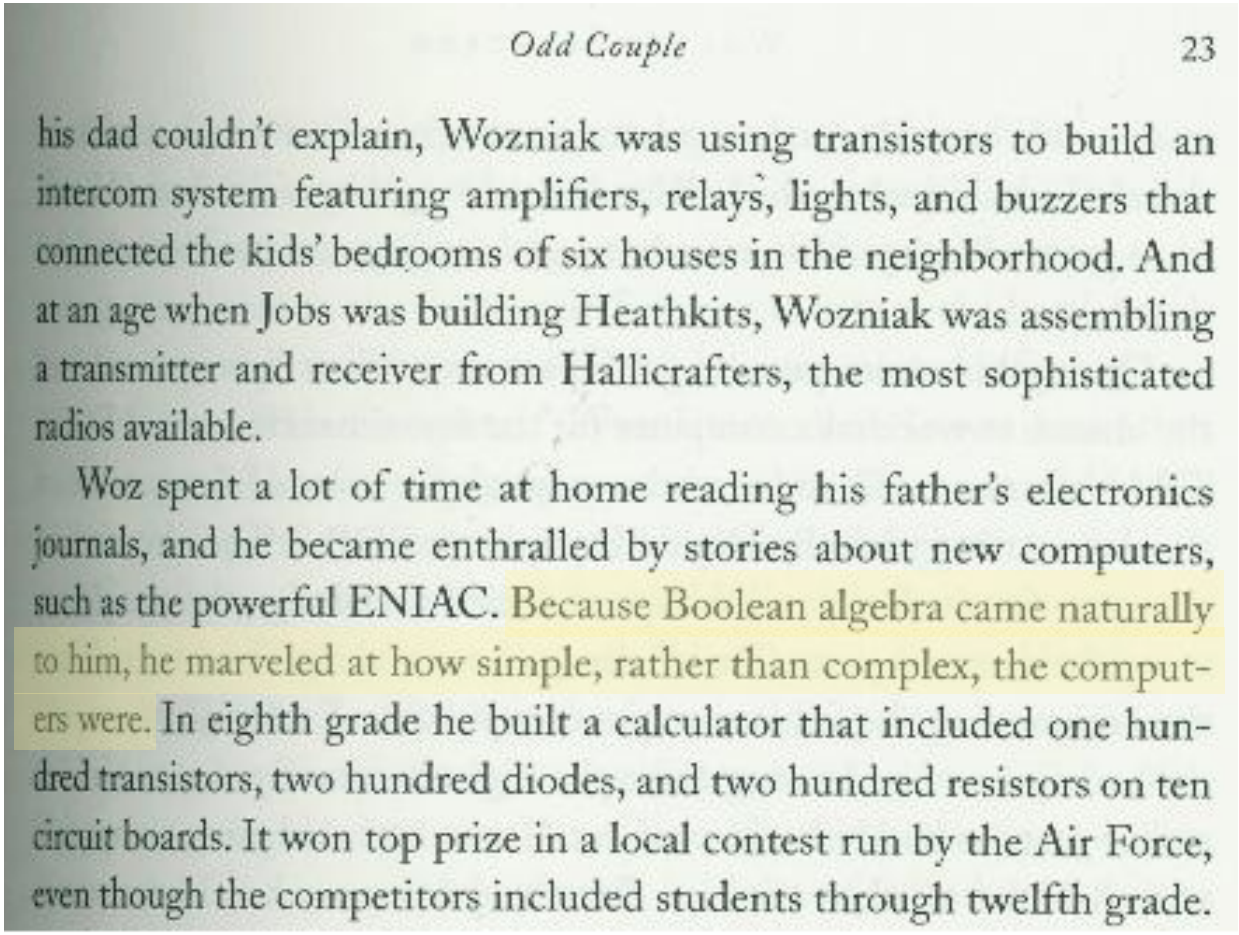
3. Boolesche Algebra und Schaltnetze

Boolesche Algebra



Odd Couple 23

his dad couldn't explain, Wozniak was using transistors to build an intercom system featuring amplifiers, relays, lights, and buzzers that connected the kids' bedrooms of six houses in the neighborhood. And at an age when Jobs was building Heathkits, Wozniak was assembling a transmitter and receiver from Hallicrafters, the most sophisticated radios available.

Woz spent a lot of time at home reading his father's electronics journals, and he became enthralled by stories about new computers, such as the powerful ENIAC. Because Boolean algebra came naturally to him, he marveled at how simple, rather than complex, the computers were. In eighth grade he built a calculator that included one hundred transistors, two hundred diodes, and two hundred resistors on ten circuit boards. It won top prize in a local contest run by the Air Force, even though the competitors included students through twelfth grade.

Walter Isacsson, "Steve Jobs", Simon & Schuster, New York, 2011

Boolesche Algebra

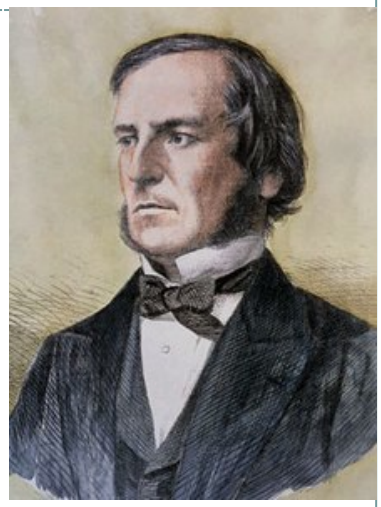
Definition 3.1 (Boolesche Algebra)
Eine *boolesche Algebra* besteht aus einer nichtleeren Menge B, sowie den beiden binären Operatoren ***: B x B → B** und **+: B x B → B**.
Das Tripel (B, *, +) ist genau dann eine boolesche Algebra, wenn für beliebige x,y,z ϵ B die folgenden Huntington ‘schen Axiome erfüllt sind:

1.Die Menge B enthält mindestens zwei Elemente x, y für die gilt:
$x \neq y$ (A1)
2.Geschlossenheit:
Für jedes x, y ϵ B gilt:
- $(x * y) \in B$ (A2)
- $(x + y) \in B$ (A3)

3.Kommutativgesetze:
Die Reihenfolge der Operanden darf vertauscht werden:
- $x * y = y * x$ (A4)
- $x + y = y + x$ (A5)

Boolesche Algebra

4. Distributivgesetze:
   Gleiche Elemente dürfen ausgeklammert werden:
   - x * (y + z) = (x * y) + (x * z) (A6)
   - x + (y * z) = (x + y) * (x + z) (A7)
5. Neutrale Elemente:
   Es existieren Elemente e, n ϵ B, für die gilt:
   - x * e = x (A8)
   - x + n = x (A9)
6. Komplement:
   Für alle x ϵ B existiert ein x ‘ ϵ B, für das gilt:
   - x * x ‘ = n (A10)
   - x + x ‘ = e (A11)



George Boole
1815-1864

Theoreme der booleschen Algebra

Satz 3.1 (Komplement der neutralen Elemente)
n' = e (T1)
e' = n (T2)

Satz 3.2 (Idempotenz)
x * x = x (T3)
x + x = x (T4)

Satz 3.3 (Elimination)
x * n = n (T5)
x + e = e (T6)

Satz 3.4 (Assoziativität)
x * (y * z) = (x * y) * z = x * y * z (T7)
x + (y + z) = (x + y) + z = x + y + z (T8)

Schaltalgebra

Definition 3.2 (Schaltalgebra)
Die *Schaltalgebra* (switching algebra) ist eine boolesche Algebra über der Menge B = {0,1}.

Die beiden binären Operatoren * und + der Schaltalgebra sind die logischen Operatoren Konjunktion und Disjunktion.

Schaltalgebra hat in der Informatik eine überragende Bedeutung, da heute alle digitalen Schaltungen auf ihr beruhen. Die zweiwertige Logik kann leicht durch Transistoren umgesetzt werden (Schalterprinzip).

Schaltfunktion, Schaltvariable

Definition 3.3 (Schaltvariable)
Eine Schaltvariable ist ein Symbol für die Elemente {0, 1} der Schaltalgebra.

Definition 3.4 (Schaltfunktion)
Eine Schaltfunktion **f** ist eine Funktion der Schaltalgebra, die die Abhängigkeit einer Schaltvariablen y von einer oder mehreren unabhängigen Schaltvariablen x_1, x_2,..., x_n beschreibt.

Durch die Schaltfunktion wird der abhängigen Variablen bei jeder Wertekombination der unabhängigen Variablen ein eindeutiger Funktionswert zugeordnet:

$y = f(x_1, x_2, ..., x_n) \ \epsilon \{0,1\}$

Wahrheitstabelle

Eine Schaltfunktion kann durch eine Funktionstabelle, meist Wahrheitstabelle genannt, dargestellt werden. Für alle Wertekombinationen der Eingangsvariablen ist der Wert der Ausgangsvariablen festgelegt.

x_3	x_2	x_1	$y = f(x_1,x_2,x_3)$
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

Wahrheitstabelle - verkürzt

Wenn bei bestimmten Eingangskombinationen der Wert einer oder mehrerer Variablen keinen Einfluss auf den Wert der Ausgangsvariablen hat, so kann die Wahrheitstabelle in verkürzter Darstellung angegeben werden. Für die Variablen ohne Einfluss wird das – Zeichen (don't care) benutzt.

x_3	x_2	x_1	$y = f(x_1,x_2,x_3)$
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

x_3	x_2	x_1	$y = f(x_1,x_2,x_3)$

verkürzte Darstellung

Wahrheitstabelle - verkürzt

Wenn bei bestimmten Eingangskombinationen der Wert einer oder mehrerer Variablen keinen Einfluss auf den Wert der Ausgangsvariablen hat, so kann die Wahrheitstabelle in verkürzter Darstellung angegeben werden. Für die Variablen ohne Einfluss wird das – Zeichen (don't care) benutzt.

x_3	x_2	x_1	$y = f(x_1,x_2,x_3)$
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

x_3	x_2	x_1	$y = f(x_1,x_2,x_3)$
0	0	–	0
0	1	–	1
1	–	0	0
1	–	1	1

verkürzte Darstellung

Wahrheitstabelle - verkürzt

Eine Wahrheitstabelle kann auch verkürzt werden, in dem die Eingangsvariable auf die Ausgangsseite gebracht wird. Die Funktion nimmt dann für diese Eingangskombination den Wert der Variablen an.

x_3	x_2	x_1	$y = f(x_1,x_2,x_3)$
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

	$y = f(x_1,x_2,x_3)$

verkürzte Darstellung

Wahrheitstabelle - verkürzt

Eine Wahrheitstabelle kann auch verkürzt werden, in dem die Eingangsvariable auf die Ausgangsseite gebracht wird. Die Funktion nimmt dann für diese Eingangskombination den Wert der Variablen an.

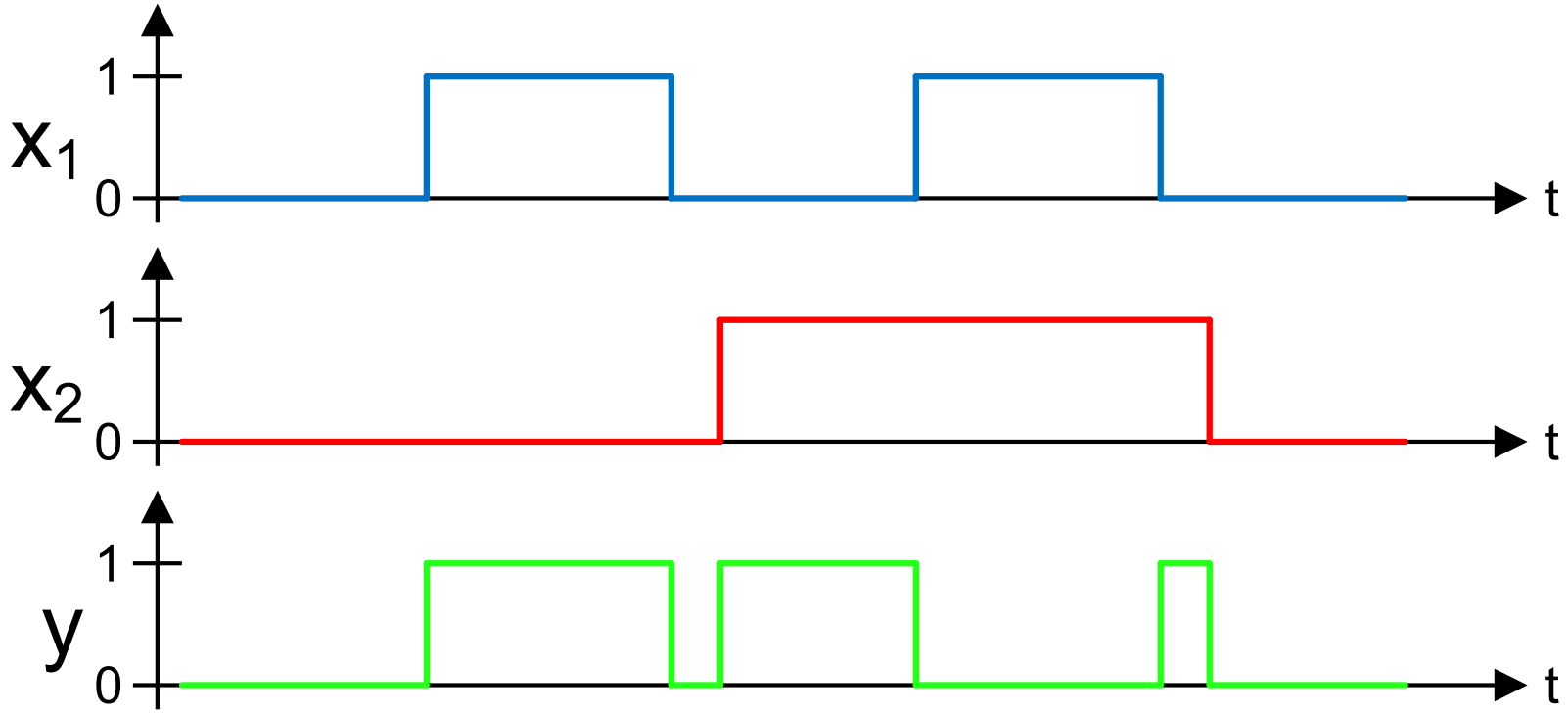
x_3	x_2	x_1	$y = f(x_1,x_2,x_3)$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

x_3	$y = f(x_1,x_2,x_3)$
0	x_2
1	x_1

verkürzte Darstellung

Zeitdiagramm

Eine Schaltfunktion kann auch durch ein Zeitdiagramm dargestellt werden. Hierbei wird der zeitliche Verlauf der Ausgangsvariable in Abhängigkeit von den Eingangsvariablen angegeben.



Boolesche Ausdrücke

Boolesche Ausdrücke stellen eine weitere Möglichkeit dar, um Schaltfunktionen darzustellen. Dabei werden die Schaltfunktionen durch Formeln ausgedrückt, welche Variablen und Operatoren enthalten.

Definition 2.5 (Boolesche Ausdrücke)
Sei V = { x_1, x_2, ..., x_n} eine Menge von Schaltvariablen. Die Menge aller boolescher Ausdrücke ist wie folgt definiert:

1. 0, 1, x_i sind boolesche Ausdrücke
2. Mit A ist auch A ‘ ein boolescher Ausdruck
3. Mit A und B sind auch A * B und A + B boolesche Ausdrücke
4. Mit A ist auch (A) ein boolescher Ausdruck

In Literatur auch andere Schreibweisen gängig:

$a * b$: $a \wedge b$; $a \cdot b$; $a\,b$;

$a + b$: $a \vee b$

a': $\overline{a}$; $\neg a$

elementare Schaltfunktionen

Nicht-Schaltfunktion (Negation)
Die Ausgangsvariable y hat dann den Wert 1, wenn x den Wert 0 hat und umgekehrt.

$$y = x'$$

x	y
0	1
1	0

UND-Schaltfunktion (Konjunktion)
Die Ausgangsvariable y hat nur dann den Wert 1, wenn x_1 **und** x_2 den Wert 1 haben.

$$y = x_1 * x_2$$

x_2	x_1	y
0	0	0
0	1	0
1	0	0
1	1	1

ODER-Schaltfunktion (Disjunktion)
Die Ausgangsvariable y hat den Wert 1, wenn x_1 **oder** x_2 den Wert 1 haben.

$$y = x_1 + x_2$$

x_2	x_1	y
0	0	0
0	1	1
1	0	1
1	1	1

Theoreme der Schaltalgebra

Satz 2.5 (Absorption)
$x * (x+y) = x$ (T9)
$x + (x*y) = x$ (T10)

Satz 2.6 (Gesetze von De Morgan)
$(x * y)' = x' + y'$ (T11)
$(x + y)' = x' * y'$ (T12)

Satz 2.7 (Negationstheorem)
Sei $f(0, 1, x_1, x_2, ..., x_n, x_1', x_2', ..., x_n', *, +)$ ein boolescher Ausdruck, in dem die Konstanten 0 und 1, die Variablen $x_1, x_2, ...,x_n$ oder die negierten Variablen $x_1', x_2', ..., x_n'$, sowie die Operatoren * und + vorkommen. Den negierten Ausdruck **f'** erhält man durch Vertauschen der Konstanten und Operatoren, sowie der Ersetzung der Variablen durch ihre Negierung und umgekehrt:

$\mathbf{f}'(0, 1, x_1, x_2, ..., x_n, x_1', x_2', ..., x_n',*, +) = f(1, 0, x_1', x_2', ..., x_n', x_1, x_2, ..., x_n, +, *)$

Dualität

Zu jedem booleschen Ausdruck f(0, 1, x_1, x_2, ..., x_n, *, +) kann man einen dualen Ausdruck $\mathbf{f^d}$ erhalten, indem man die Operatoren *, + vertauscht und die Konstanten 0 und 1 tauscht.

$\mathbf{f^d}(0, 1, x_1, x_2, ..., x_n, *, +) := f(1, 0, x_1, x_2, ..., x_n, +, *)$

Satz 2.8 (Dualität boolescher Ausdrücke)
Sind zwei boolesche Ausdrücke f und g äquivalent (f = g), dann sind auch die zugehörigen dualen Ausdrücke f^d und g^d äquivalent ($f^d = g^d$).

Literal, Monom, Minterm

Definition 2.6 (Literal)
Ein *Literal* ist eine Schaltvariable oder eine negierte Schaltvariable.

Beispiel:
x, x'

Definition 2.7 (Monom)
Ein *Monom* oder *Produktterm* ist eine Formel von einer der folgenden Formen:

- 1
- ein Literal
- eine Konjunktion von Literalen, wobei keine Variable mehr als einmal auftritt

Beispiel:
$x_1 * x_2 * x_3'$

Definition 2.8 (Minterm)
Ein *Minterm* ist ein Monom, das für jede Variable von f ein Literal enthält. Ein solches Monom wird auch als vollständiges Monom bezeichnet.

Beispiel:
$f(x_1, x_2, x_3, x_4)$
$x_1 * x_2 * x_3' * x_4'$ ist Minterm
$x_1 * x_2 * x_3'$ ist kein Minterm

Minterme

Beispiel: Minterme von einer Funktion von 2 Variablen $f(x_1, x_2)$

		m_0	m_1	m_2	m_3
x_2	x_1	$x_1' * x_2'$	$x_1 * x_2'$	$x_1' * x_2$	$x_1 * x_2$
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Jeder Minterm evaluiert bei genau einer Variablenkombination zum Wert 1!

Darstellung durch Minterme

Beispiel: Funktion von 3 unabhängigen Variablen $f(x_1, x_2, x_3)$

x_3	x_2	x_1	y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

x_3	x_2	x_1	m_1	m_2	m_4	m_6	$m_1+m_2+m_4+m_6$
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	1
0	1	0	0	1	0	0	1
0	1	1	0	0	0	0	0
1	0	0	0	0	1	0	1
1	0	1	0	0	0	0	0
1	1	0	0	0	0	1	1
1	1	1	0	0	0	0	0

$f(x_1, x_2, x_3) = m_1+m_2+m_4+m_6 = (x_1*x_2\text{‘}*x_3\text{‘}) + (x_1\text{‘}*x_2*x_3\text{‘}) + (x_1\text{‘}*x_2\text{‘}*x_3) + (x_1\text{‘}*x_2*x_3)$

Disjunktive Normalform

Definition 3.9 (Disjunktive Normalform)
Eine *disjunktive Normalform* (DNF) oder eine *Summe von Produkten* (sum of products, SOP) ist eine Formel von einer der folgenden Formen:

- 0
- ein Produktterm
- eine Disjunktion von Produkttermen

Eine DNF, die ausschließlich aus Mintermen besteht, heißt ausgezeichnete oder kanonische disjunktive Normalform (KDNF).

Die Anzahl der Terme der disjunktiven Normalform wächst mit der Anzahl der Einsen des Funktionswerts!

Satz 3.9 (Hauptsatz der Schaltalgebra 1)
Jede Schaltfunktion kann mit der disjunktiven Normalform dargestellt werden.

Klausel, Maxterm

Definition 3.10 (Klausel)
Eine *Klausel* oder *Summenterm* ist eine Formel von einer der folgenden Formen:

- 0
- ein Literal
- eine Disjunktion von Literalen, wobei keine Variable mehr als einmal auftritt

Beispiel:
$x_1 + x_2 ' + x_3 '$

Definition 3.11 (Maxterm)
Ein *Maxterm* ist eine Klausel, die für jede Variable von f ein Literal enthält. Eine solche Klausel wird auch als vollständige Klausel bezeichnet.

Beispiel: $f(x_1, x_2, x_3, x_4)$

$x_1 + x_2 + x_3 ' + x_4 '$ ist Maxterm
$x_1 + x_2 + x_3 '$ ist kein Maxterm

Konjunktive Normalform

Definition 3.12 (Konjunktive Normalform)
Eine *konjunktive Normalform* (KNF) oder ein *Produkt von Summen* (product of sums, POS) ist eine Formel von einer der folgenden Formen:

- 1
- ein Summenterm
- eine Konjunktion von Summentermen

Eine KNF, die ausschließlich aus Maxtermen besteht, heißt ausgezeichnete oder kanonische konjunktive Normalform (KKNF).

Die Anzahl der Terme der konjunktiven Normalform wächst mit der Anzahl der Nullen des Funktionswerts!

Satz 3.10 (Hauptsatz der Schaltalgebra 2)
Jede Schaltfunktion kann mit der konjunktiven Normalform dargestellt werden.

DNF und KNF

x_3	x_2	x_1	y	Terme
0	0	0	0	$M_0 = x_1 + x_2 + x_3$
0	0	1	1	$m_1 = x_1 * x_2' * x_3'$
0	1	0	1	$m_2 = x_1' * x_2 * x_3'$
0	1	1	1	$m_3 = x_1 * x_2 * x_3'$
1	0	0	1	$m_4 = x_1' * x_2' * x_3$
1	0	1	0	$M_5 = x_1' + x_2 + x_3'$
1	1	0	1	$m_6 = x_1' * x_2 * x_3$
1	1	1	0	$M_7 = x_1' + x_2' + x_3'$

kanonische DNF:

$f(x_1, x_2, x_3) = m_1+m_2+m_3+m_4+m_6$

$= \sum m(1,2,3,4,6)$

kanonische KNF:

$f(x_1, x_2, x_3) = M_0*M_5*M_7$

$= \prod M(0,5,7)$

Umwandlung DNF und KNF

1. **Umwandlung einer kanonischen DNF in kanonische KNF:**
   Ersetze die Summe von Mintermen durch ein Produkt aus Maxtermen. Verwende hierbei alle Maxterme, deren Index bei den Mintermen nicht auftreten.

   $f(x_1, x_2, x_3) = m_0 + m_5 + m_7 = M_1 * M_2 * M_3 * M_4 * M_6$

2. **Umwandlung einer kanonischen KNF in kanonische DNF:**
   Ersetze das Produkt aus Maxtermen durch eine Summe von Mintermen. Verwende hierbei alle Minterme, deren Index bei den Maxtermen nicht auftreten.

   $f(x_1, x_2, x_3) = M_0 * M_1 * M_2 = m_3 + m_4 + m_5 + m_6 + m_7$

Shannon-Theorem

Satz 3.11 (Entwicklungssatz von Shannon)
Sei f eine beliebige n-stellige boolesche Funktion.
Dann gilt:

$$f(x_1, x_2, ..., x_n) = (x_i \text{ ‘ } * f_{xi=0}) + (x_i * f_{xi=1})$$

$$f(x_1, x_2, ..., x_n) = (x_i + f_{xi=0}) * (x_i \text{ ‘ } + f_{xi=1})$$

mit
$f_{xi=0} = f(x_1, x_2, ..., x_{i-1}, 0, x_{i+1}, ..., x_n)$
$f_{xi=1} = f(x_1, x_2, ..., x_{i-1}, 1, x_{i+1}, ..., x_n)$

Claude
Shannon
1916-2001

Die iterative Anwendung der oberen Gleichung erzeugt die kanonische DNF.
Die iterative Anwendung der unteren Gleichung erzeugt die kanonische KNF.

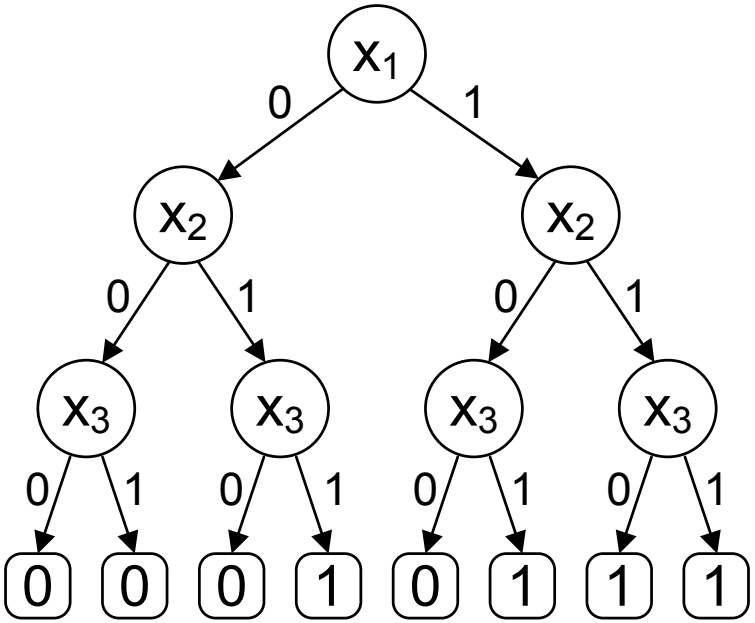
Binäre Entscheidungsdiagramme

Definition 3.13 (Binäres Entscheidungsdiagramm)
Ein *binäres Entscheidungsdiagramm* (Binary Decision Diagram, BDD) der Schaltfunktion $f(x_1, x_2, ..., x_n)$ ist ein gewurzelter, azyklischer und gerichteter Graph G = (V, E) mit der Knotenmenge V und der Kantenmenge E. Die Knotenmenge V besteht aus Entscheidungsknoten und terminalen Knoten mit folgenden Eigenschaften:

- Für Entscheidungsknoten u gilt:
  - u ist mit einer Schaltvariablen x_i markiert, der Entscheidungsvariablen des Knotens.
  - u hat genau zwei ausgehende Kanten (beschriftet mit 0 und 1), die zu Teilgraphen führen bei denen $f_{xi=0}$ und $f_{xi=1}$ gilt.
- Terminale Knoten werden mit 0 oder 1 markiert und haben keine ausgehenden Kanten.

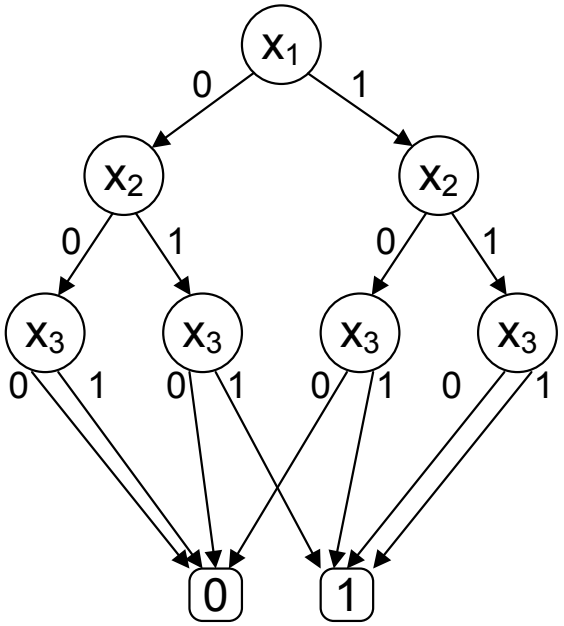
Binäre Entscheidungsdiagramme

Definition 3.14 (geordnetes binäres Entscheidungsdiagramm)
Ein *geordnetes binäres Entscheidungsdiagramm* (Ordered Binary Decision Diagram, OBDD) ist ein binäres Entscheidungsdiagramm, bei dem alle Variablen auf dem Pfad von der Wurzel zu den Blättern durchlaufen werden und die Reihenfolge der Variablen auf allen Pfaden von der Wurzel zu den Blättern gleich ist.



Binäre Entscheidungsdiagramme

Ein Binäres Entscheidungsdiagramm kann weiter vereinfacht werden. Dazu werden zunächst alle Blätter auf zwei Blätter reduziert, das Blatt mit dem Wert 0 und das andere mit dem Wert 1.

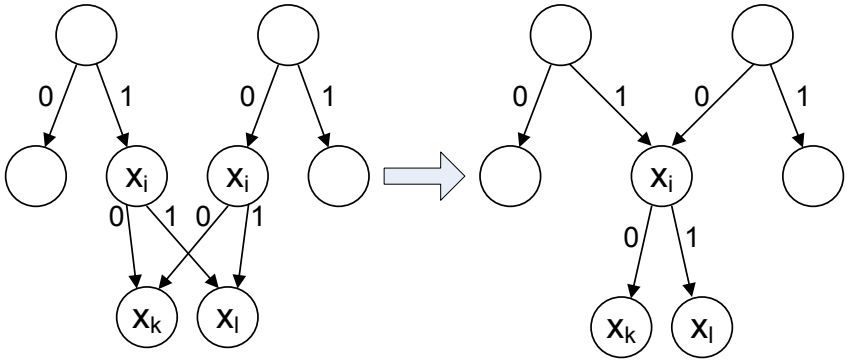


Ausgehend von diesen beiden Blätter wird der Graph in Richtung Wurzel durchlaufen und folgende beiden Vereinfachungsregeln angewandt:

Binäre Entscheidungsdiagramme

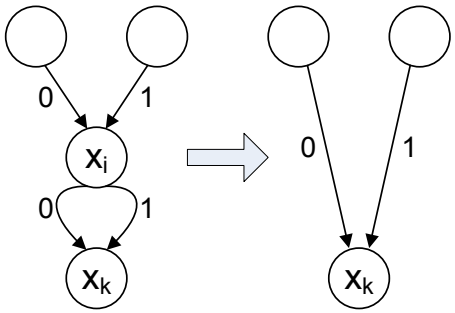
- **Verschmelzung gleicher Teilbäume**

Zwei Knoten, deren linke und rechte Kanten auf die gleichen Nachfolgeknoten verweisen, können zu einem Knoten zusammengefasst werden.



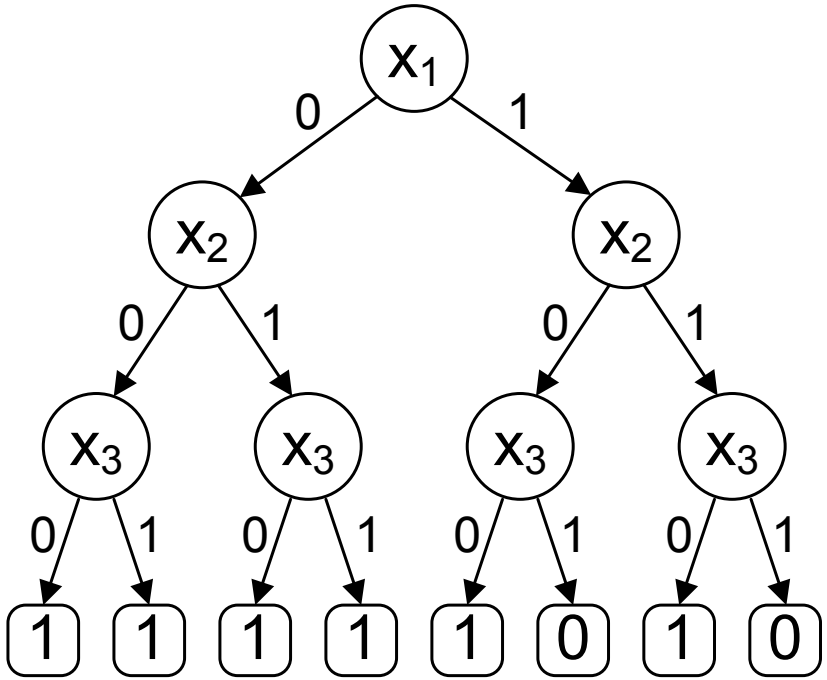
- **Löschen von Knoten mit gleichen Nachfolgern**

Knoten, deren ausgehende Kanten auf den gleichen Knoten zeigen, können gelöscht werden, da der Wert der Variablen keinen Einfluss auf den Funktionswert hat.



Beispiel Vereinfachungen

Gegeben folgendes BDD:



Binäre Entscheidungsdiagramme

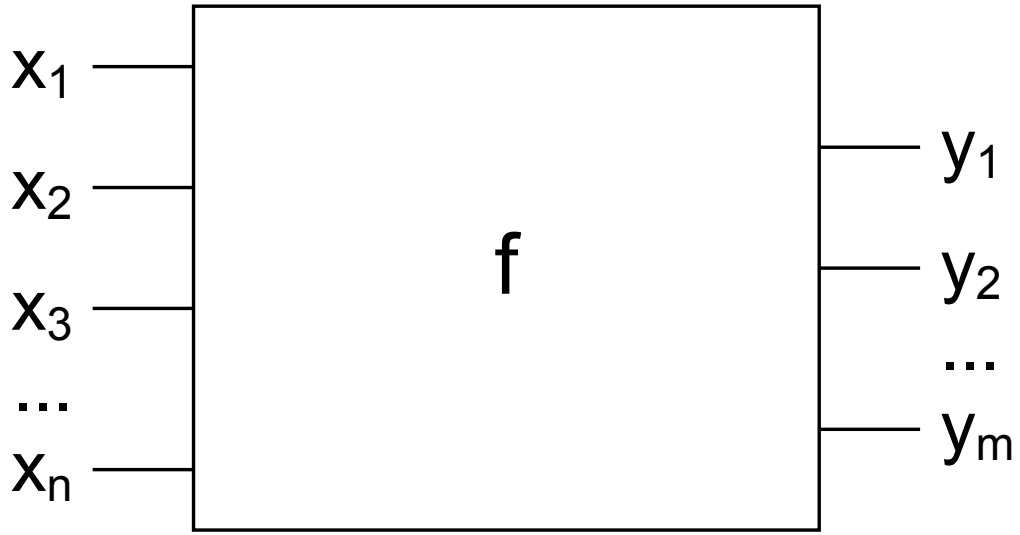
Definition 3.15 (reduziertes geordnetes binäres Entscheidungsdiagramm)
Ein *reduziertes geordnetes binäres Entscheidungsdiagramm* (Reduced Ordered Binary Decision Diagram, ROBDD) ist ein geordnetes Entscheidungsdiagramm, bei dem keine weiteren Vereinfachung mehr möglich sind.

Durch ROBDDs können Schaltfunktionen eindeutig dargestellt werden. Sie sind weitere kanonische Normalformdarstellungen von Schaltfunktionen.
Gilt nur, wenn die Reihenfolge der Variablen festgelegt ist!

Schaltnetze

Definition 3.16 (Schaltnetz)
Ein *Schaltnetz* ist ein System mit n Eingängen $\underline{X} = \{x_1, x_2, ..., x_n\}$ und m Ausgängen $\underline{Y} = \{y_1, y_2, ..., y_m\}$, welche die Schaltfunktion $\underline{Y} = f(\underline{X})$ realisiert, und bei der die Ausgangssignale $\underline{Y}$ in jedem Zeitpunkt nur von den gegenwärtigen Eingangssignalen $\underline{X}$ abhängen.

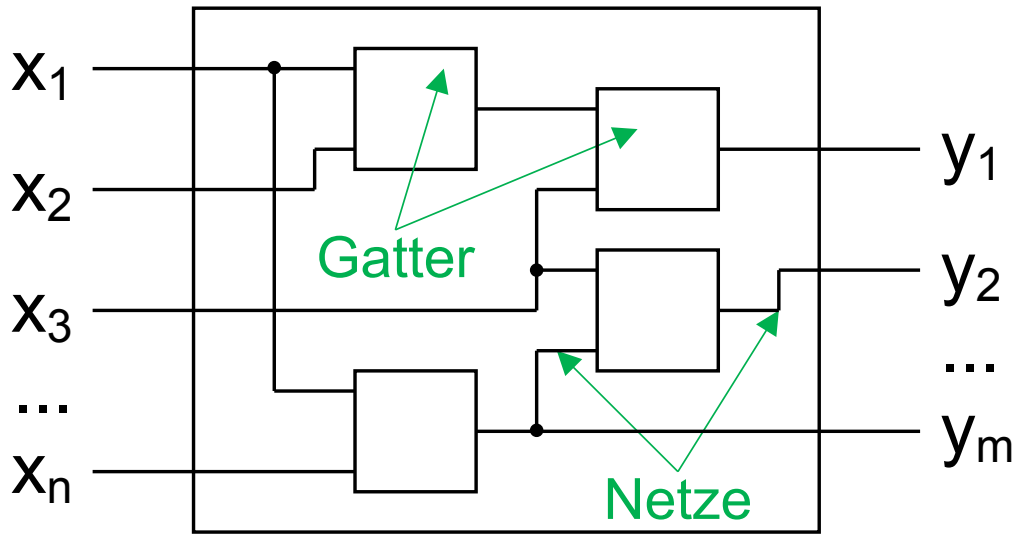
Black-Box-Ansicht eines Schaltnetzes
Interface-Sicht (Eingänge, Ausgänge – ohne inneren Aufbau)



Schaltnetze

White-Box-Ansicht eines Schaltnetzes
Berücksichtigung des inneren Aufbaus

Ein Schaltnetz ist ein rückwirkungsfreies (azyklisches) Netz aus logischen Schaltgliedern (Gattern), welche durch *Netze* (Leitungen mit ≥ 2 Anschlüssen) verbunden sind. Es wird auch als *kombinatorische Schaltung* (*combinatorial circuit)* bezeichnet.



Bündelfunktionen

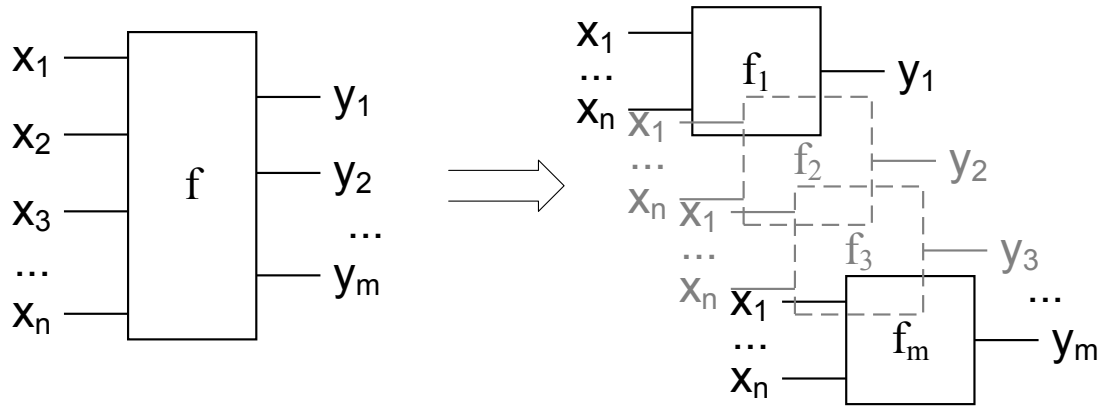
Schaltnetze mit mehreren Ausgängen können durch eine einzige sog. *Bündelfunktion* beschrieben werden:

$$f : \{0,1\}^n \rightarrow \{0,1\}^m$$

$$\underline{Y} = f(\underline{X}) \quad (y_1, y_2, \ldots, y_m) = f(x_1, x_2, \ldots, x_n)$$

Meist werden die Schaltnetze jedoch durch mehrere, für jede Ausgangsvariable eigenständige, Funktionen beschrieben:

$$y_i = f_i(x_1, x_2, \ldots, x_n) \quad i = 1 \ldots m$$



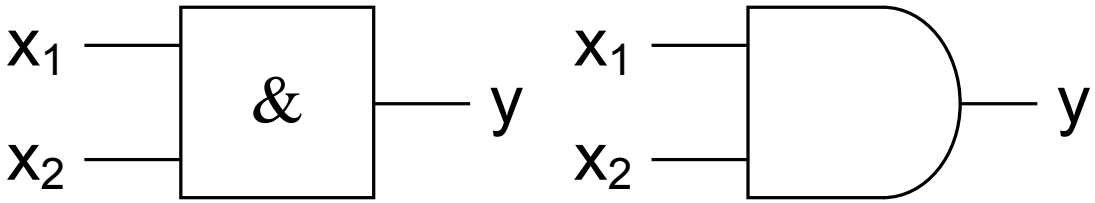
Grundgatter

UND-Gatter (AND)

Realisierung der Konjunktion $y = x_1 * x_2$ DIN-Norm

US-Norm

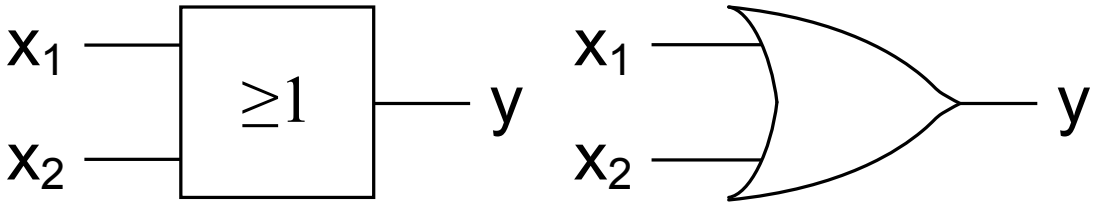
x_2	x_1	y
0	0	0
0	1	0
1	0	0
1	1	1



ODER-Gatter (OR)

Realisierung der Disjunktion $y = x_1 + x_2$

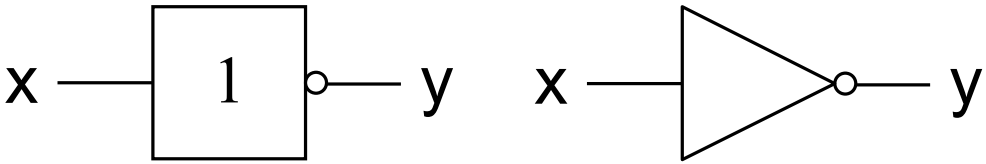
x_2	x_1	y
0	0	0
0	1	1
1	0	1
1	1	1



Inverter (NOT)

Realisierung der Negation $y = x'$

x	y
0	1
1	0



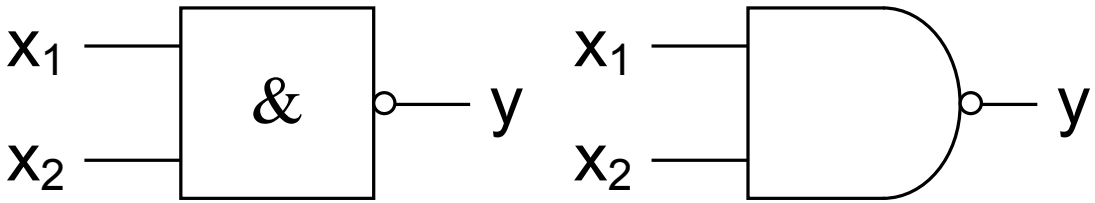
Grundgatter

NAND-Gatter
Realisierung der Funktion $y = (x_1 * x_2)'$

DIN-Norm

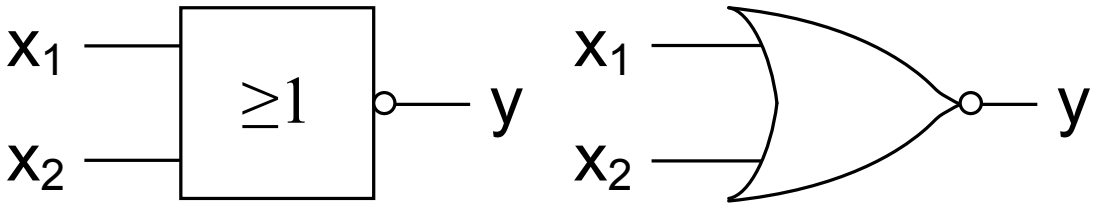
US-Norm

x_2	x_1	y
0	0	1
0	1	1
1	0	1
1	1	0



NOR-Gatter
Realisierung der Funktion $y = (x_1 + x_2)'$

x_2	x_1	y
0	0	1
0	1	0
1	0	0
1	1	0



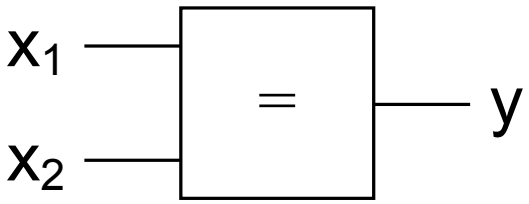
Grundgatter

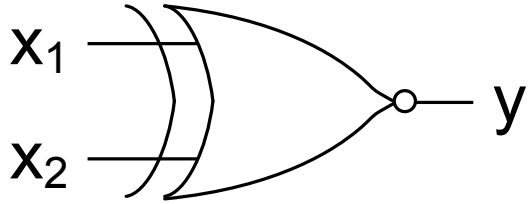
Äquivalenz-Gatter (XNOR)
Realisierung der Funktion $y = (x_1 \equiv x_2)$

DIN-Norm

US-Norm

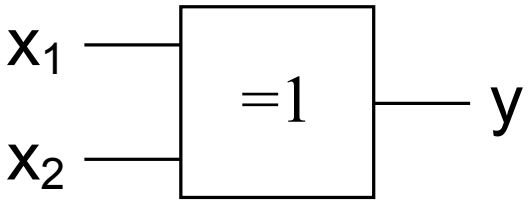
x_2	x_1	y
0	0	1
0	1	0
1	0	0
1	1	1

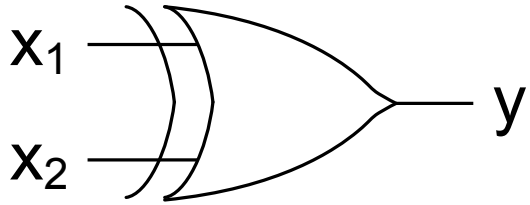




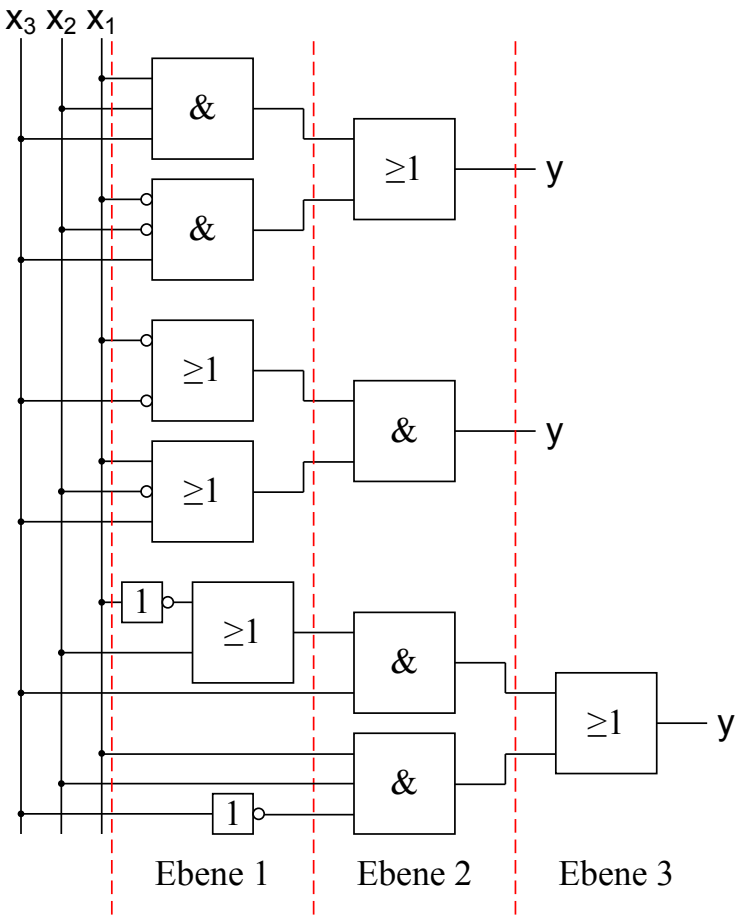
Antivalenz-Gatter (XOR)
Realisierung der Funktion $y = x_1 \oplus x_2$

x_2	x_1	y
0	0	0
0	1	1
1	0	1
1	1	0



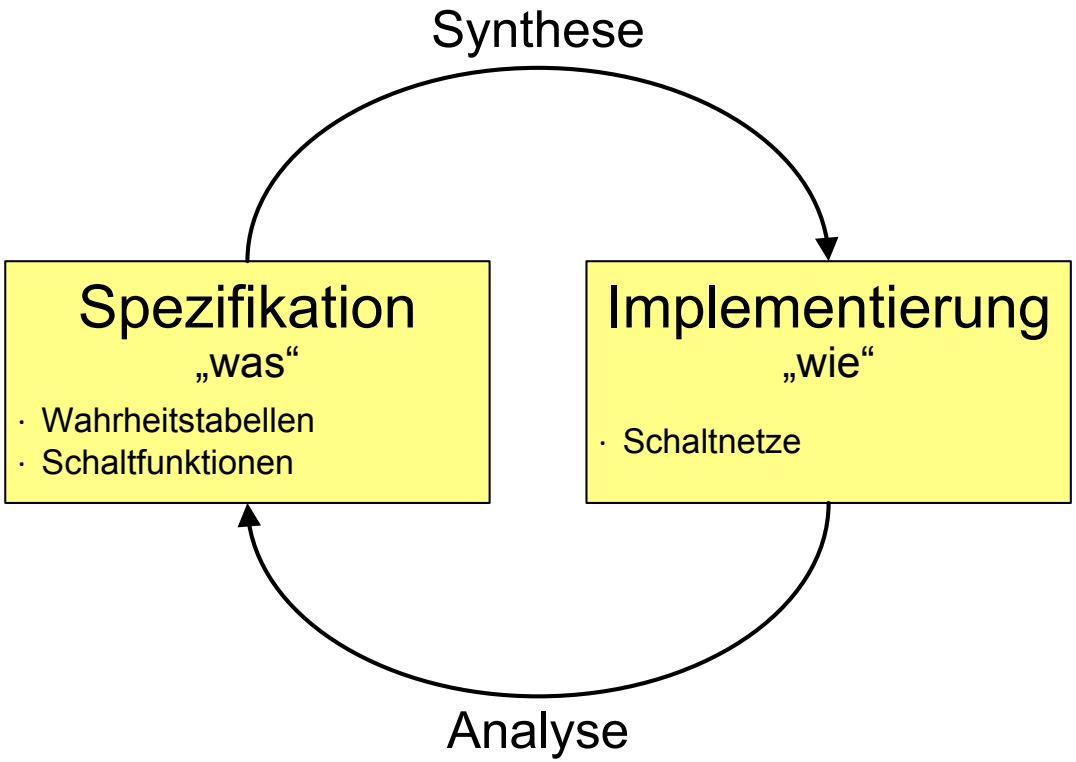


zweistufige und mehrstufige Schaltnetze



- zweistufige Schaltnetze besitzen nur zwei Gatterebenen
- zweistufige Schaltnetze können direkt aus DNF oder KNF abgeleitet werden
- Inverter zählen bei Ebenenbetrachtung nicht mit
- mehrstufige Schaltnetze haben mehr als zwei Gatterebenen

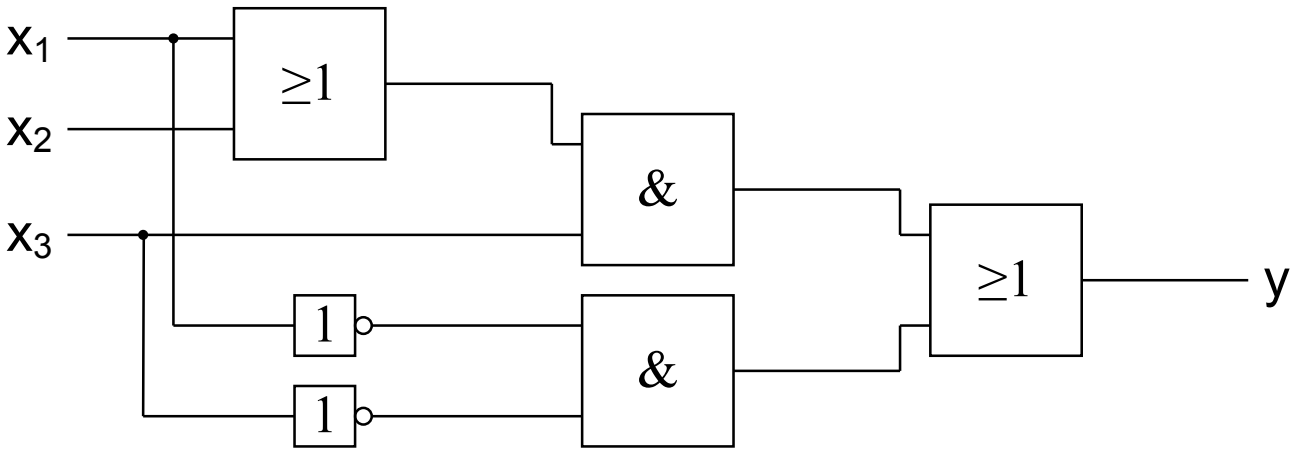
Analyse und Synthese



Schaltungsanalyse

Die Aufgabe der *Schaltungsanalyse* ist es, zu einem gegebenen Schaltnetz eine formale Beschreibung der Schaltfunktion zu erhalten. Möglichkeiten hierfür sind:

• Aufstellung einer Wahrheitstabelle, durch Bestimmung der Ausgangswerte bei allen Eingangskombinationen

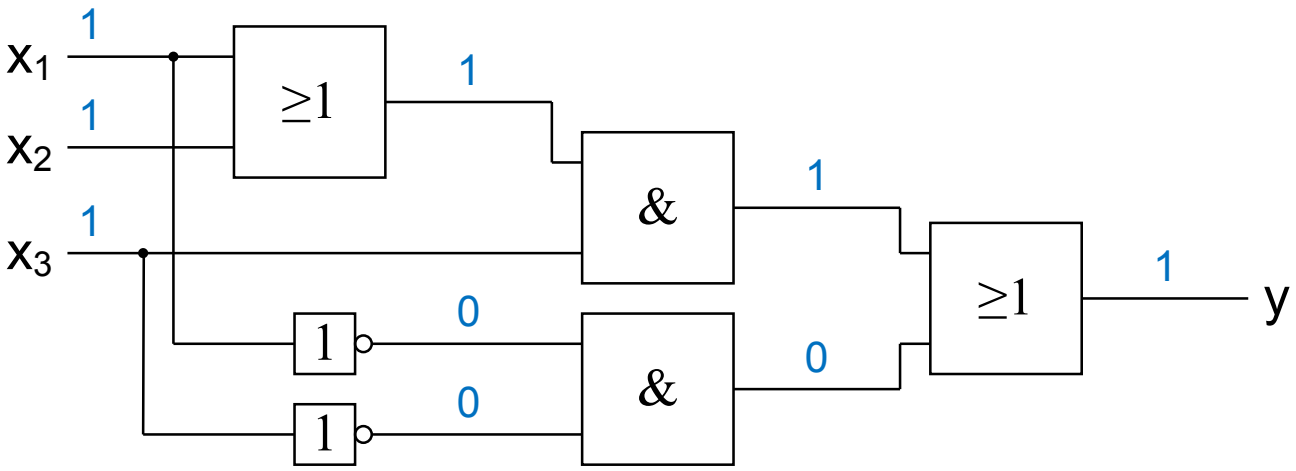


x_3	x_2	x_1	y

Schaltungsanalyse

Die Aufgabe der *Schaltungsanalyse* ist es, zu einem gegebenen Schaltnetz eine formale Beschreibung der Schaltfunktion zu erhalten. Möglichkeiten hierfür sind:

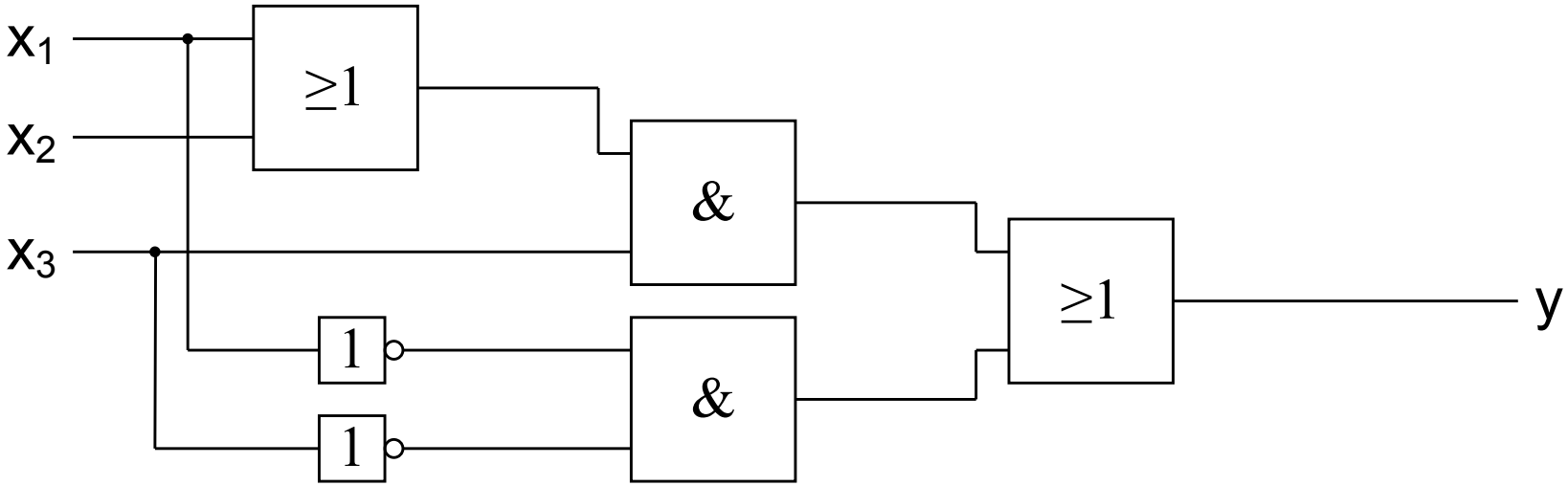
• Aufstellung einer Wahrheitstabelle, durch Bestimmung der Ausgangswerte bei allen Eingangskombinationen



x_3	x_2	x_1	y
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

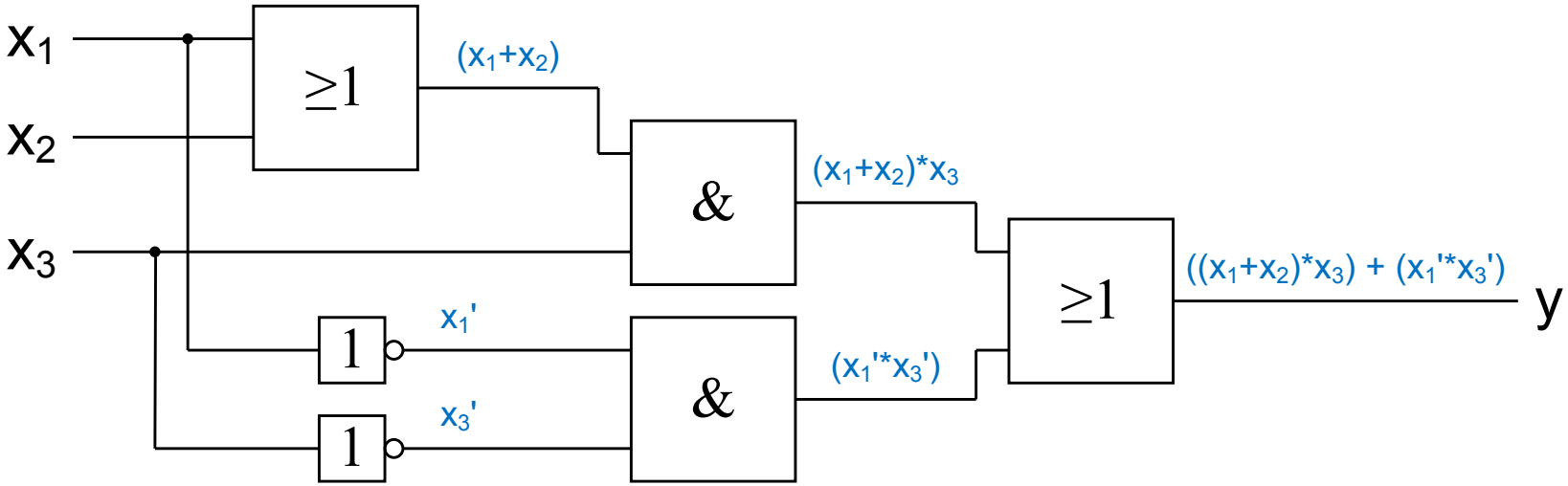
Symbolische Schaltungsanalyse

• Aufstellung eines booleschen Ausdrucks durch schrittweises Durchlaufen des Schaltnetzes und Generierung von geklammerten Ausdrücken

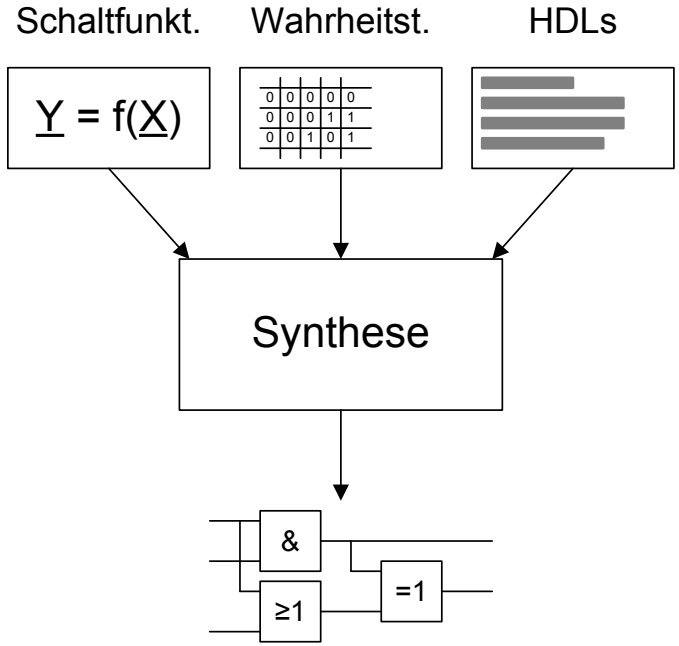


Symbolische Schaltungsanalyse

• Aufstellung eines booleschen Ausdrucks durch schrittweises Durchlaufen des Schaltnetzes und Generierung von geklammerten Ausdrücken



Schaltungssynthese



Die Umsetzung (Implementierung) einer Schaltfunktion in ein Schaltnetz wird *Schaltungsynthese* oder kurz *Synthese* genannt.

Heute ein automatisierter Vorgang, der ausgehend von Beschreibungen mit Wahrheitstabellen, Schaltfunktionen oder mit Hardwarebeschreibungssprachen (engl. hardware description languages, HDLs) hin zu Netzlisten vom Rechner durchgeführt wird, inklusive möglicher Optimierungen.

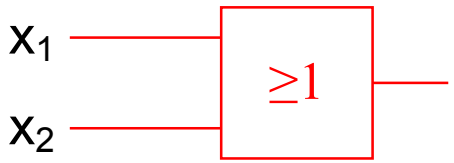
Optimierungen werden in diesem Kapitel jedoch noch nicht betrachtet; dies folgt in Kapitel 9.

Umsetzung einer Schaltfunktion

$$y = \overline{\overline{((x_1+x_2) * (x_3+x_4'))} + ((x_2'*x_3) + x_4)}$$

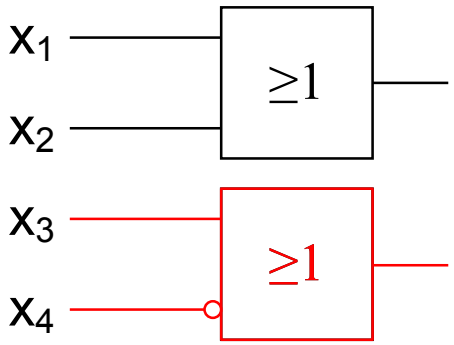
Umsetzung einer Schaltfunktion

$$y = \overline{\overline{((x_1+x_2) * (x_3+x_4'))} + ((x_2'*x_3) + x_4)}$$



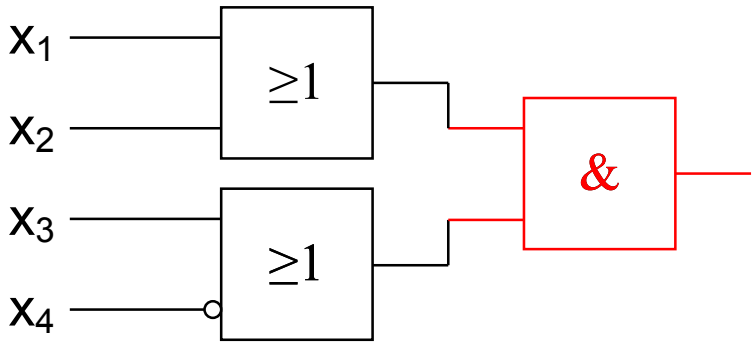
Umsetzung einer Schaltfunktion

$$y = \overline{\overline{((x_1+x_2) * (x_3+x_4'))} + ((x_2'*x_3) + x_4)}$$



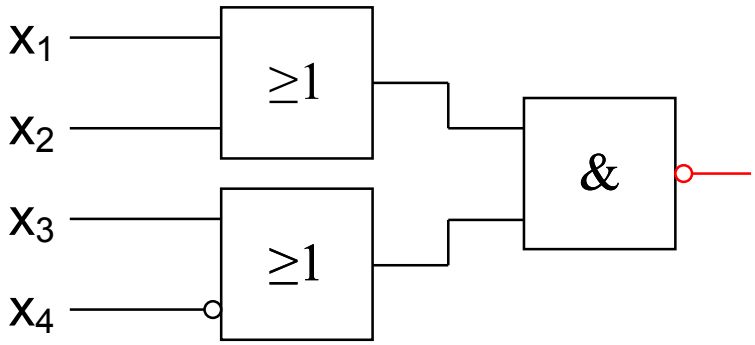
Umsetzung einer Schaltfunktion

$$y = \overline{((x_1+x_2) * (x_3+x_4'))} + ((x_2'*x_3) + x_4)$$



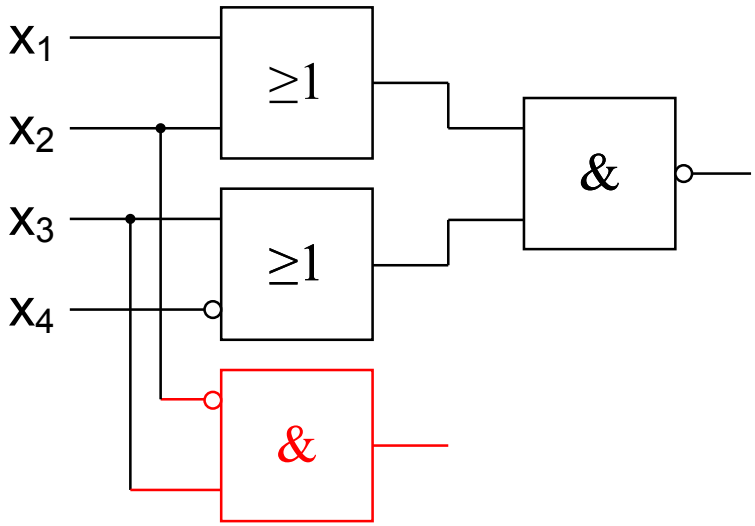
Umsetzung einer Schaltfunktion

$$y = \overline{((x_1+x_2) * (x_3+x_4')) + ((x_2'*x_3) + x_4)}$$



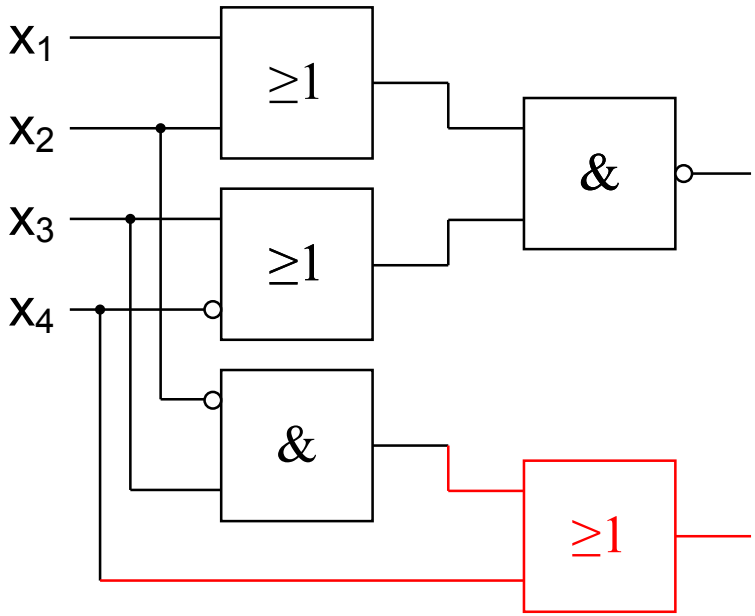
Umsetzung einer Schaltfunktion

$$y = \overline{\overline{((x_1+x_2) * (x_3+x_4'))} + ((x_2'*x_3) + x_4)}$$



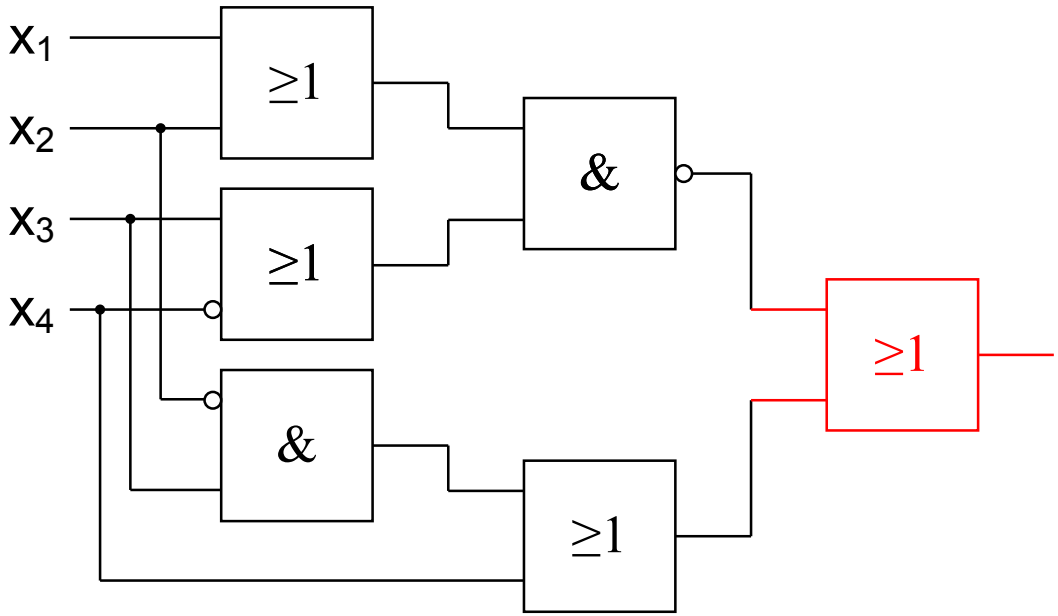
Umsetzung einer Schaltfunktion

$$y = \overline{\overline{((x_1+x_2) * (x_3+x_4'))} + ((x_2'*x_3) + x_4)}$$



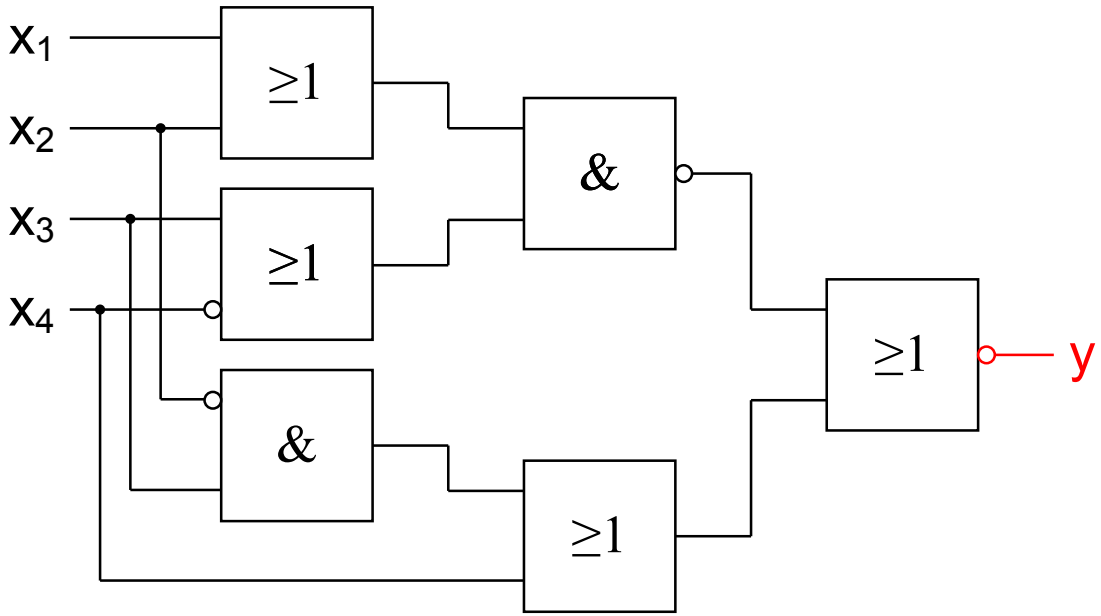
Umsetzung einer Schaltfunktion

$$y = \overline{\overline{((x_1+x_2) * (x_3+x_4'))} + ((x_2'*x_3) + x_4)}$$



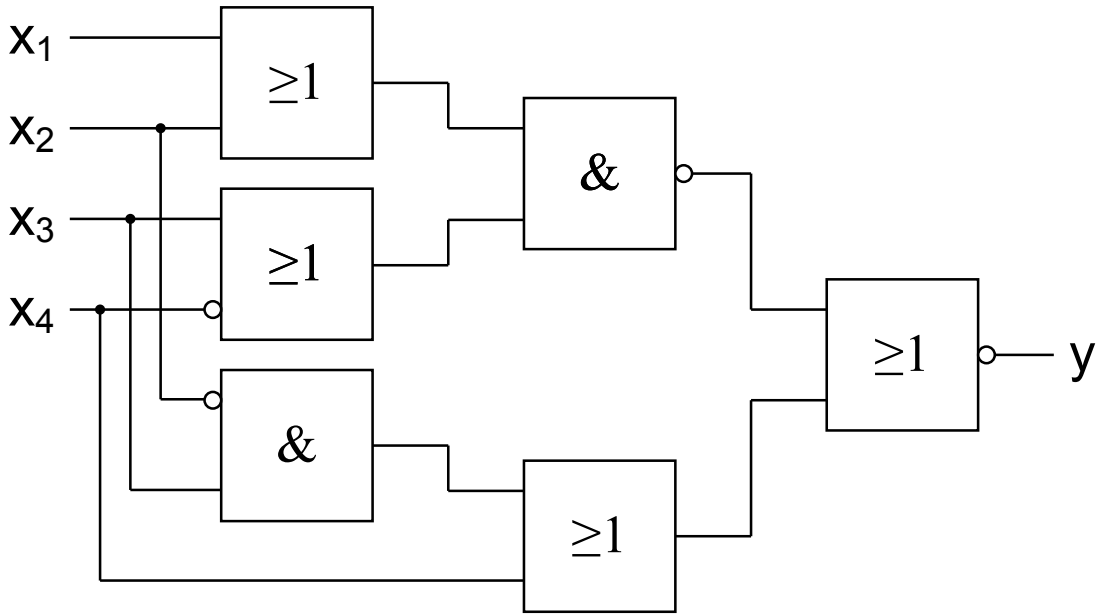
Umsetzung einer Schaltfunktion

$$y = \overline{\overline{((x_1+x_2) * (x_3+x_4'))} + ((x_2'*x_3) + x_4)}$$



Umsetzung einer Schaltfunktion

$$y = \overline{\overline{((x_1+x_2) * (x_3+x_4'))} + ((x_2'*x_3) + x_4)}$$



Kontrollfragen

- Können Sie die formale Definition der mathematischen Struktur „Boole‘sche Algebra“ sinngemäß wiedergeben?
- Wie verhalten sich Boole‘sche Algebra und Schaltalgebra zueinander?
- Gibt es eine Boole‘sche Algebra mit drei Elementen, entsprechend drei Antwortmöglichkeiten bei der Kodierung von Informationen? Versuchen Sie, eine solche Algebra durch Angabe von Verknüpfungen + und * sowie des Komplements ‘ zu konstruieren, so dass die Huntington‘schen Axiome erfüllt sind. Vielleicht funktioniert es; wenn nicht, sollten Sie auf einen Widerspruch stoßen, der die Konstruktion unmöglich macht.
- Wie verhalten sich die Begriffe Schaltfunktion, Boole‘scher Ausdruck und Schaltnetz zueinander?
- Geben Sie die Funktionalität der Grundgatter mit Hilfe von verkürzten Wahrheitstabellen (und damit kompakter als auf Folie 21-23) an.
- Was ist der Unterschied zwischen einem „negierten Ausdruck“ und einem „dualen Ausdruck“?
- Wie viele unterschiedliche Schaltfunktionen mit *n* Variablen gibt es? (Hinweis: wie viele unterschiedliche Einträge sind in der Wahrheitstabelle möglich?)
- Wie viele unterschiedliche Boole‘sche Ausdrücke mit *n* Variablen gibt es?

Kontrollfragen

- Definieren Sie die Begriffe Literal, Monom, Minterm, DNF, KDNF
- Stellen Sie diesen Begriffen ihre dualen Begriffe gegenüber
- Wann ist es im Vergleich günstiger, eine KDNF oder eine KKNF zu verwenden?
- Reduzieren Sie das BDD von Folie 28 so weit wie möglich (in der Vorlesung wurde dies nur für das BDD von Folie 31 gezeigt).
- Reduzieren Sie das BDD von Folie 31 intuitiv (ohne Anwendung der Reduktionsregeln, möglicherweise in einem Schritt), indem Sie Pfade, die zum gleichen Blatt führen, zusammenfassen.
- Welchen Zusammenhang gibt es zwischen BDDs und dem Shannon-Theorem?
- Welche Darstellungsform erhalten Sie, wenn Sie das Shannon-Theorem nacheinander auf sämtliche Variablen einer Schaltfunktion anwenden? (Beispiel: entwickeln Sie die Schaltfunktion $f(x_1, x_2, x_3) = (x_1 * x_2) + x_3$ nach x_1, das Zwischenergebnis nach x_2 und schließlich nach x_3.)
- Erstellen Sie aus einem ROBDD (z.B. Tafelanschrieb zu Folie 31) einen äquivalenten booleschen Ausdruck. Zur Probe können Sie alle Wertekombinationen der Variablen einsetzen und die Ergebnisse mit dem BDD vergleichen.

Kontrollfragen

- Warum sind in Schaltnetzen keine Zyklen erlaubt?
- Können Sie die hier definierten Grundgatter mit ihrem Schaltsymbol (entweder konsistent nach DIN oder nach US-Norm) und ihrer Wahrheitstabelle angeben?
- Welche der vorgestellten Analyseverfahren sind einsetzbar, wenn ein Chip als Black-Box vorliegt (d.h. Sie können Werte an die Eingänge anlegen und an den Ausgängen messen)?
- Was ist erforderlich, um die symbolische Analyse einzusetzen?
- Formulieren Sie eine Grammatik Boolescher Ausdrücke, z.B. als (E)BNF – (Extended) Backus-Naur-Form. Kommen Sie ggf. auf diesen Punkt zurück, wenn (E)BNF in anderen Vorlesungen behandelt wurde.
- Wie können BDDs zur Optimierung von Schaltfunktionen eingesetzt werden? Was wäre dabei das Optimierungsziel?