Kapitel 3: Boolesche Algebra

Inhalt:

3.1 Grundlegende Operationen und Gesetze

Die Boolesche Algebra stellt die Grundlage für den Entwurf von elektronischen Schaltungen bis hin zu Computern dar.

Sie ist nach George Boole (1815 – 1864) benannt, der als erster eine "Algebra der Logik" entwickelt hat.

Diese kennt nur die beiden Zustände "wahr" und "falsch", die in einem Schaltkreis den grundlegenden Zuständen "Strom fließt" und "Strom fließt nicht" entsprechen.

Diese beiden Zustände werden im Folgenden durch die Zahlen 1 und 0 modelliert.

Konjunktion

Die Boolesche Algebra geht von der Menge $\{0, 1\}$ aus. Auf dieser Menge sind folgende drei Operationen definiert.

1. Die Konjunktion $\wedge$ (Und-Verknüpfung) ist eine binäre Verknüpfung, hängt also von zwei Argumenten ab. Sie ist genau dann 1, wenn das erste *und* das zweite Argument 1 ist, und in jedem anderen Fall 0. Der Ausdruck $a \wedge b$ wird "a und b" gelesen.

$\wedge$	0	1
0	0	0
1	0	1

Disjunktion

2. Auch die Disjunktion $\vee$ (Oder-Verknüpfung) ist eine binäre Verknüpfung. Sie ist genau dann 1, wenn das erste *oder* das zweite Argument 1 ist, und sonst 0. Der Ausdruck $a \vee b$ wird "a oder b" gelesen. "Oder" ist dabei als einschließendes Oder zu verstehen, das heißt nicht im Sinne von "entweder oder".

$\vee$	0	1
0	0	1
1	1	1

Negation

3. Die Negation $\neg$ verlangt nur ein Argument. Sie ist 0, wenn das Argument 1 ist, und 1, wenn das Argument 0 ist. Die Negation heißt auch Nicht-Operator, und man liest $\neg a$ als "nicht a".

x	$\neg x$
0	1
1	0

Reihenfolge der Auswertung

Um komplexere boolesche Ausdrücke zu erhalten, können diese drei Operationen mehrfach hintereinander ausgeführt werden.

Dabei ist zu beachten, dass die Operationen unterschiedliche Priorität haben:

$\neg$ kommt vor $\wedge$, und $\wedge$ kommt vor $\vee$.

Möchte man andere Prioritäten setzen, so muss man die entsprechenden Teilausdrücke in Klammern setzen.

Beispiel: Es gilt $\neg 0 \vee 1 \wedge 0 = (\neg 0) \vee (1 \wedge 0) = 1 \vee 0 = 1$.

Rechengesetze

Für die Operationen $\wedge$, $\vee$ und $\neg$ gelten eine Reihe Rechengesetze. Sie sind uns vom Umgang mit den rationalen und reellen Zahlen her vertraut, wenn wir an Addition und Multiplikation denken.

3.1.1 Satz. Für alle $x, y, z \in \{0, 1\}$ gelten die folgenden Gesetze:

(a) Kommutativgesetze: $x \wedge y = y \wedge x$ und $x \vee y = y \vee x$.

(b) Assoziativgesetze: $x \wedge (y \wedge z) = (x \wedge y) \wedge z$, $x \vee (y \vee z) = (x \vee y) \vee z$.

(c) Distributivgesetze: $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$

und $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$.

(d) Existenz neutraler Elemente: $1 \wedge x = x$ und $0 \vee x = x$.

(e) Existenz des Komplements: $x \wedge \neg x = 0$ und $x \vee \neg x = 1$.

Beweis

Exemplarisch beweisen wir das erste Distributivgesetz. Dazu zeigen wir mit einer Wertetabelle, dass sich für alle möglichen Werte von x, y und z auf der linken Seite stets das Gleiche ergibt wie auf der rechten:

x	y	z	$y \wedge z$	$x \vee (y \wedge z)$	$x \vee y$	$x \vee z$	$(x \vee y) \wedge (x \vee z)$
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

❑

Axiomatische Definition

Bemerkung. Oft wird der Begriff Boolesche Algebra weiter gefasst, als wir es hier tun. Man kann die Gesetze aus Satz 3.1.1 auch als Axiome fordern und sagen: „Eine Menge mit den Operationen $\wedge$, $\vee$ und $\neg$ heißt Boolesche Algebra, wenn die folgenden Gesetze gelten ...".

Dann folgt nach Satz 3.1.1, dass die Menge $\{0, 1\}$ zusammen mit der Und-, Oder- und Nicht-Operation eine Boolesche Algebra ist.

Allerdings ist sie dann nicht mehr die einzige. Zum Beispiel bildet dann auch die Menge aller Teilmengen einer Menge eine Boolesche Algebra, wenn man als Operationen die Mengenoperationen Durchschnitt, Vereinigung und Komplement nimmt.

Dualität

Beobachtung: Die Gesetze aus Satz 3.1.1 bestehen jeweils aus zwei Teilen, die auseinander hervorgehen, wenn man $\wedge$ und $\vee$, sowie 1 und 0 vertauscht. Aus dieser Symmetrie folgt, dass wir auch in jeder Folgerung aus diesen Gesetzen diese Vertauschungen durchführen können. Diese Eigenschaft der Booleschen Algebra heißt Dualität.
Ein Satz, der durch Vertauschen von $\wedge$ und $\vee$ und von 1 und 0 aus einem anderen Satz hervorgeht, heißt zu diesem dual.

3.1.2 Korollar (Dualität). Jede Aussage, die aus Satz 3.1.1 folgt, bleibt gültig, wenn die Operationen $\wedge$ und $\vee$ sowie die Elemente 1 und 0 überall gleichzeitig vertauscht werden. ❑

Weitere Gesetze

Eine erste Anwendung findet die Dualität beim Beweis des folgenden Satzes, der weitere Gesetze der Booleschen Algebra beschreibt.

3.1.3 Satz. Für alle $x, y \in \{0, 1\}$ gelten die folgenden Gesetze:

(a) Absorptionsgesetze: $x \wedge (x \vee y) = x$ und $x \vee (x \wedge y) = x$.

(b) Idempotenzgesetze: $x \vee x = x$ und $x \wedge x = x$.

(c) Involutionsgesetz: $\neg(\neg x) = x$.

(d) Gesetze von de Morgan (Augustus de Morgan, 1806 – 1871):

$$\neg(x \wedge y) = \neg x \vee \neg y \quad \text{und} \quad \neg(x \vee y) = \neg x \wedge \neg y.$$

Beweis (I)

Eine Möglichkeit, diese Gesetze zu beweisen, ist sicherlich, wieder alle möglichen Werte für x und y einzusetzen und zu überprüfen, ob die linke und rechte Seite übereinstimmen. Diese Möglichkeit bietet sich für den Nachweis von (c) und (d) an.

Eine andere Möglichkeit ist, die bereits bewiesenen Gesetze aus 3.1.1 anzuwenden. Dies wollen wir am Beispiel von (a) und (b) verdeutlichen. Es gilt

$$x \wedge (x \vee y) = (x \vee 0) \wedge (x \vee y) = x \vee (0 \wedge y) = x \vee 0 = x.$$

Dies ist das erste Absorptionsgesetz.

Beweis (II)

Auf Grund der Dualität können wir in jedem dieser Schritte, also auch im Endergebnis, $\wedge$ und $\vee$ sowie 1 und 0 vertauschen und erhalten daraus das zweite Absorptionsgesetz:

$$x \vee (x \wedge y) = x.$$

Ferner gilt

$$x \vee x = (x \vee x) \wedge 1 = (x \vee x) \wedge (x \vee \neg x) = x \vee (x \wedge \neg x) = x \vee 0 = x;$$

damit haben wir das erste Idempotenzgesetz gezeigt. Das zweite Idempotenzgesetz folgt wiederum aus der Dualität.

Ganz ähnlich kann man die restlichen Gesetze nachweisen. ❑

3.2 Boolesche Funktionen und ihre Normalformen

Eine boolesche Funktion ist eine Abbildung, die jeweils n Bits auf ein einziges Bit abbildet. Formal können wir das wie folgt ausdrücken:
Eine n-stellige boolesche Funktion ist eine Abbildung

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

Das bedeutet, dass jedem n-Tupel $(x_1, x_2, ..., x_n)$ mit $x_i \in \{0, 1\}$ eindeutig eine Zahl $f(x_1, x_2, ..., x_n) \in \{0, 1\}$ zugeordnet wird.

In der Schaltungstechnik können wir uns eine boolesche Funktion als Schaltung vorstellen, die aus mehreren Eingabebits ein einziges Ausgabebit (zum Beispiel die Summe der Eingabebits mod 2) berechnet.

Wie viele boolesche Funktionen gibt es?

3.2.1 Satz. Es gibt $2^{(2^n)}$ verschiedene n-stellige boolesche Funktionen.

Beweis. Die Menge $\{0, 1\}^n$ besteht aus 2^n Elementen. Um eine boolesche Funktion festzulegen, muss man für jedes dieser 2^n Elemente das Bild festlegen. Für jedes Element gibt es dabei genau zwei Möglichkeiten: 0 oder 1. Insgesamt gibt es also

$$\underbrace{2 \cdot 2 \cdot \ldots \cdot 2}_{2^n \text{ Faktoren}} = 2^{(2^n)}$$

Möglichkeiten, die boolesche Funktion festzulegen. ❑

Beispiel: Die vier 1-stelligen booleschen Funktionen sind die Nullfunktion $f(x) := 0$, die Identität $f(x) := x$, die Negation $f(x) := \neg x$ und die Einsfunktion $f(x) := 1$.

Alle 2-stelligen booleschen Funktionen

Wir wollen im Folgenden die 2-stelligen booleschen Funktionen genauer untersuchen. All diese Funktionen sind in folgender Wertetabelle aufgelistet.

x	y	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}	f_{16}
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

Konjunktion und Disjunktion

Einige dieser 2-stelligen booleschen Funktionen sind uns schon bekannt. So erkennen wir etwa in f_2 die Konjunktion (Und-Verknüpfung) wieder, denn an der Wertetabelle können wir ablesen

$$f_2(x, y) = x \wedge y.$$

Die Funktion f_8 ist die Disjunktion (Oder-Verknüpfung):

$$f_8(x, y) = x \vee y.$$

Aber auch andere 2-stellige boolesche Funktionen sind von besonderer Bedeutung. Ihre Bedeutung wird klarer, wenn wir sie als boolesche Ausdrücke schreiben, also als Verknüpfungen von $\wedge$, $\vee$ und $\neg$.

NOR und NAND

Die Funktion f_9 lässt sich als

$$f_9(x, y) = \neg(x \vee y)$$

schreiben. Es handelt sich dabei also um eine negierte Oder-Verknüpfung. Daher wird sie auch als NOR-Verknüpfung (vom englischen “not or”) bezeichnet.

Genauso gibt es auch eine NAND-Verknüpfung (von “not and”). In der Tabelle finden wir sie als

$$f_{15}(x, y) = \neg(x \wedge y).$$

XOR

Die Funktion f_7 ergibt genau dann 1, wenn ihre beiden Argumente unterschiedlich sind, das heißt, wenn *entweder* x *oder* y gleich 1 ist. Daher können wir sie als

$$f_7(x, y) = (x \vee y) \wedge \neg(x \wedge y)$$

schreiben.

Vom englischen “exclusive or” für “ausschließendes Oder” leitet sich ihr Name ab: XOR-Verknüpfung.

Äquivalenzfunktion und Implikation

Durch Negation der XOR-Verknüpfung erhalten wir die Funktion

$$f_{10}(x, y) = (x \wedge y) \vee \neg(x \vee y)$$

Sie ist genau dann gleich 1, wenn ihre beiden Argumente gleich („äquivalent") sind. Daher heißt sie Äquivalenzfunktion.

Die Funktion f_{14} ergibt stets 1, außer wenn $x = 1$ und $y = 0$ ist. Sie heißt Implikation und lässt sich wie folgt als boolescher Ausdruck schreiben:

$$f_{14}(x, y) = \neg x \vee y.$$

Von der Wertetabelle zum booleschen Ausdruck?

Bei einigen der bisher betrachteten booleschen Funktionen war es ganz einfach, von der Wertetabelle auf einen booleschen Ausdruck zu kommen. Bei anderen haben wir eine ganze Menge Intuition gebraucht.

Frage: Wie kann man systematisch von der Wertetabelle einer Funktion auf einen booleschen Ausdruck schließen?

Klar: Ein solcher boolescher Ausdruck kann nicht eindeutig sein, denn boolesche Ausdrücke können mittels der Gesetze aus 3.1 umgeformt werden. So beschreiben beispielsweise die beiden Ausdrücke $x \wedge (y \vee z)$ und $(x \wedge y) \vee (x \wedge z)$ die gleiche Funktion, da sie auf Grund des Distributivgesetzes ineinander umgeformt werden können.

Ziel: Ablesen des Ausdrucks aus der Tabelle!

Ziel: Gesucht ist ein möglichst einfacher Ausdruck, den man möglichst einfach aus der Wertetabelle erhalten kann.

Dieses Ziel erreichen wir in zwei Schritten:

1. Zunächst werden wir die Normalformen kennen lernen. Dabei handelt es sich spezielle Formen von booleschen Ausdrücken, die man aus der Wertetabelle „ablesen" kann.

2. Da diese Normalformen oft eine komplizierte Gestalt haben, werden wir danach untersuchen, wie man boolesche Ausdrücke vereinfachen kann.

Das Prinzip der Normalformen

Ein boolescher Ausdruck besteht im Allgemeinen aus mehreren (eventuell negierten) Variablen, die durch $\wedge$ und $\vee$ verknüpft sind.

Diese beiden Operationen können „getrennt" werden. Dazu gibt es zwei Möglichkeiten:
Entweder werden die Variablen zunächst durch $\wedge$ verknüpft (innere Verknüpfung) und die daraus entstehenden Terme anschließend durch $\vee$ verbunden (äußere Verknüpfung) oder umgekehrt.

Je nachdem, welche Operation dabei als äußere durchgeführt wird, erhält man auf diese Weise eine disjunktive bzw. eine konjunktive Normalform des booleschen Ausdrucks.

Konjunktive Normalform

Eine Vollkonjunktion ist ein boolescher Ausdruck, in dem alle Variablen genau einmal vorkommen und durch $\wedge$ (konjunktiv) verbunden sind. Dabei dürfen die Variablen auch negiert auftreten.

Ein Ausdruck liegt in der disjunktiven Normalform vor, wenn er aus Vollkonjunktionen besteht, die durch $\vee$ (disjunktiv) verknüpft sind.

Beispiel: Der boolesche Ausdruck

$$(x \wedge \neg y \wedge \neg z) \vee (\neg x \wedge y \wedge \neg z) \vee (x \wedge \neg y \wedge z)$$

ist aus drei Vollkonjunktionen aufgebaut, die durch $\vee$ verknüpft sind. Er liegt also in der disjunktiven Normalform vor.

Von der Tabelle zur disjunktiven Normalform

betrachten wir die 3-stellige boolesche Funktion f, die durch folgende Wertetabelle gegeben ist.

Zeile	x	y	z	f(x, y, z)
1	0	0	0	0
2	0	0	1	0
3	0	1	0	1
4	0	1	1	1
5	1	0	0	0
6	1	0	1	0
7	1	1	0	0
8	1	1	1	1

Aufstellen der einzelnen Vollkonjunktionen

Um die disjunktive Normalform aufzustellen, müssen wir Vollkonjunktionen finden, die, wenn man sie mit $\vee$ verknüpft, die Funktion f darstellen. Dazu gehen wir schrittweise vor.

1. Schritt: Wir suchen die Zeilen, die den Funktionswert 1 liefern. Hier sind dies die Zeilen 3, 4 und 8.

2. Schritt: Für jede dieser Zeilen stellen wir die Vollkonjunktion auf, die für die Variablenwerte dieser Zeilen den Wert 1 liefert. Dazu verknüpfen wir alle Variablen durch $\wedge$, wobei genau diejenigen negiert werden, deren Wert in der entsprechenden Zeile gleich 0 ist. Wir erhalten für

Zeile 3: $\neg x \wedge y \wedge \neg z$, Zeile 4: $\neg x \wedge y \wedge z$, Zeile 8: $x \wedge y \wedge z$.

Aufstellen der disjunktiven Normalform

3. Schritt: Diese Vollkonjunktionen werden durch $\vee$ verknüpft. Dadurch ist der resultierende Ausdruck genau dann gleich 1, wenn eine dieser Vollkonjunktionen gleich 1 ist, das heißt für die Variablenwerte der Zeilen 3, 4 und 5. Der resultierende Ausdruck hat daher die gleiche Wertetabelle wie die Funktion f, stellt also die gleiche Funktion dar. Damit haben wir die disjunktive Normalform von f gefunden. Sie lautet

$$f(x, y, z) = (\neg x \wedge y \wedge \neg z) \vee (\neg x \wedge y \wedge z) \vee (x \wedge y \wedge z).$$

Beobachtung: Die disjunktive Normalform enthält genau so viele Vollkonjunktionen, wie in der Wertetabelle der Funktionswert 1 vorkommt. Daher bietet es sich an, die disjunktive Normalform aufzustellen, wenn der Funktionswert 1 relativ selten vorkommt.

Konjunktive Normalform

Treten in der Wertetabelle viele Einsen auf, so ist es günstiger, die konjunktive Normalform zu verwenden.

Eine Volldisjunktion ist ein boolescher Ausdruck, in dem alle Variablen genau einmal vorkommen und $\vee$ (disjunktiv) verbunden sind. Dabei dürfen die Variablen auch negiert auftreten.

Ein Ausdruck liegt in der konjunktiven Normalform vor, wenn er aus Volldisjunktionen besteht, die durch $\wedge$ (konjunktiv) verknüpft sind.

Beispiel: Ein Ausdruck in konjunktiver Normalform ist

$$(\neg x \vee y \vee \neg z) \wedge (\neg x \vee \neg y \vee z) \wedge (x \vee \neg y \vee z)$$

Von der Tabelle zur konjunktiven Normalform

Das Verfahren, mit dem man von der Wertetabelle einer booleschen Funktion zu ihrem Ausdruck in konjunktiver Normalform gelangt, funktioniert ähnlich wie das Verfahren für die disjunktive Normalform.

Genau genommen ist es „dual" zum vorherigen Verfahren.

Das heißt, die beiden Verfahren gehen auseinander hervor, wenn man $\wedge$ durch $\vee$ und 1 durch 0 ersetzt.

Beispiel

wollen wir die konjunktive Normalform der 3-stelligen Funktion aufstellen, die durch folgende Wertetabelle gegeben ist.

Zeile	x	y	z	f(x, y, z)
1	0	0	0	1
2	0	0	1	0
3	0	1	0	0
4	0	1	1	1
5	1	0	0	1
6	1	0	1	1
7	1	1	0	0
8	1	1	1	1

Aufstellen der einzelnen Volldisjunktionen

Wieder gehen wir schrittweise vor.

1. Schritt: Wir suchen die Zeilen, die den Funktionswert 0 liefern. Hier sind dies die Zeilen 2, 3 und 7.

2. Schritt: Für jede dieser Zeilen stellen wir die Volldisjunktion auf, die für die Variablenwerte dieser Zeilen den Wert 0 liefert. Dazu verknüpfen wir alle Variablen durch $\vee$, wobei genau diejenigen negiert werden, deren Wert in der entsprechenden Zeile gleich 1 ist. Auf diese Weise erhalten wir für

Zeile 2: $x \vee y \vee \neg z$, Zeile 3: $x \vee \neg y \vee z$, Zeile 7: $\neg x \vee \neg y \vee z$.

Aufstellen der konjunktiven Normalform

3. Schritt: Diese Volldisjunktionen werden durch $\wedge$ verknüpft. Durch die Verknüpfung mit $\wedge$ ist der resultierende Ausdruck genau dann gleich 0, wenn eine dieser Volldisjunktionen gleich 0 ist, das heißt für die Variablenwerte der Zeilen 2, 3 und 7. Der resultierende Ausdruck hat daher die gleiche Wertetabelle wie die Funktion f, stellt also die gleiche Funktion dar. Damit haben wir die konjunktive Normalform von f gefunden. Sie lautet

$$f(x, y, z) = (x \vee y \vee \neg z) \wedge (x \vee \neg y \vee z) \wedge (\neg x \vee \neg y \vee z).$$

Bemerkung: Auch zu einem gegebenen Ausdruck kann man eine Normalform finden, indem man zunächst die Wertetabelle aufstellt.

3.3 Vereinfachen von Funktionen

Prinzipiell kann man zur Vereinfachung eines booleschen Ausdrucks alle Gesetze aus Abschnitt 3.1 anwenden.

Wie kann man dabei systematisch vorgehen?

Beim Verfahren von Karnaugh und Veitch handelt es sich um ein graphisches Verfahren, das sich für Funktionen mit bis zu vier Variablen einfach durchführen lässt.

Bei mehr als vier Variablen werden die benötigten Diagramme sehr unübersichtlich. Alternativ kann man auf das Verfahren von Quine und McCluskey (ohne graphische Darstellung) zurückgreifen.

Das Verfahren von Karnaugh und Veitch

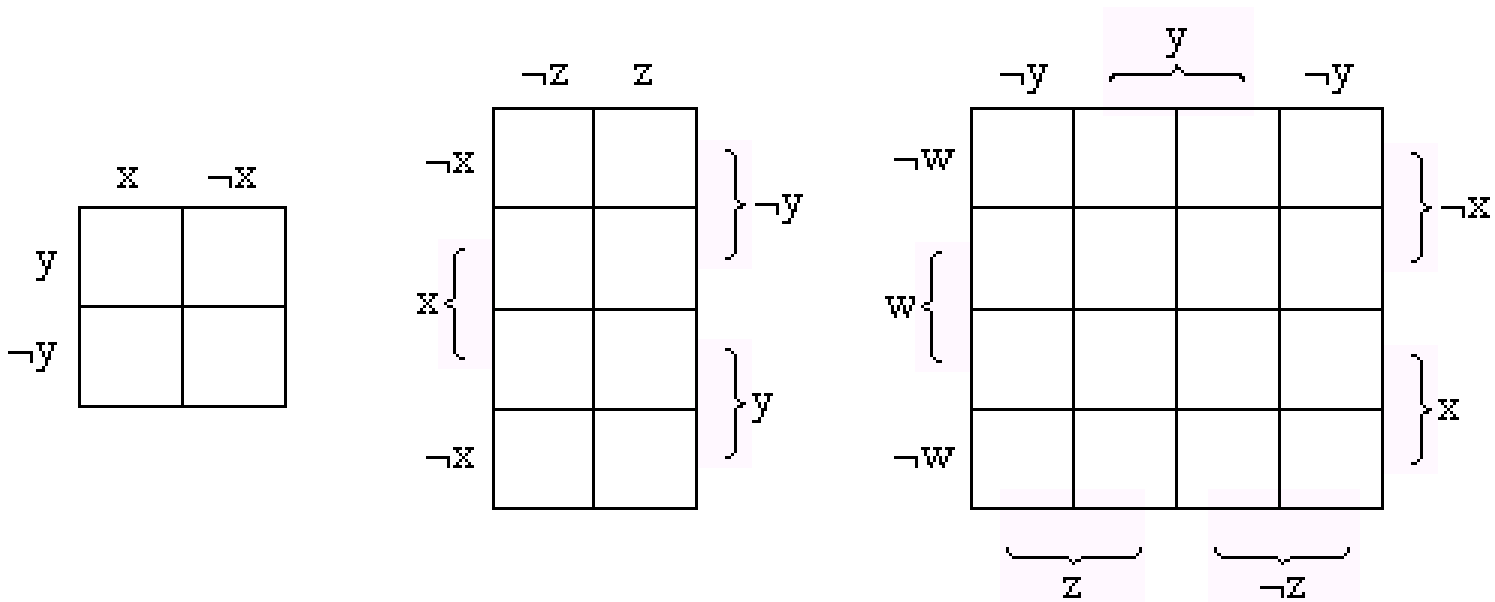
Das Verfahren von Karnaugh und Veitch geht von der disjunktiven Normalform aus.

Grundidee: Den Ausdruck systematisch so umzuformen, dass Terme der Form $x \vee \neg x$ entstehen. Nach 3.1.1 (e) haben diese Terme stets den Wert 1 und können in einer Konjunktion weggelassen werden.

Dies erreicht man wie folgt: Der in disjunktiver Normalform vorliegende Ausdruck wird in einem Karnaugh-Veitch-Diagramm (kurz: KV-Diagramm) dargestellt. Dies ist ein rechteckiges Schema, in dem jedes Feld genau einer möglichen Vollkonjunktion entspricht. Je nach Anzahl der Variablen sieht dieses Diagramm unterschiedlich aus.

KV-Diagramme

Die KV-Diagramme für 2-, 3- und 4-stellige Funktionen sind in folgender Abbildung dargestellt.



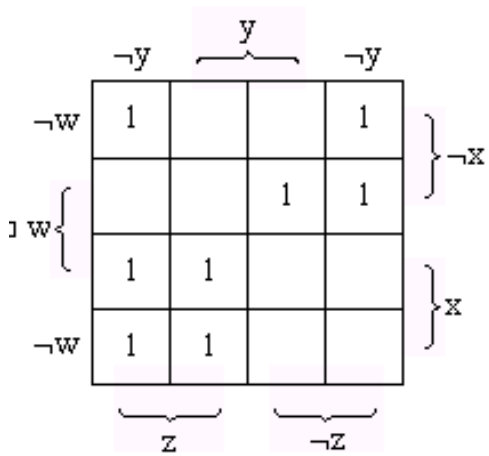
Eintragen eines Ausdrucks

Um einen kompletten Ausdruck, der in disjunktiver Normalform vorliegt, einzutragen, schreiben wir für jede Vollkonjunktion, die in dem Ausdruck vorkommt, eine 1 in das entsprechende Feld des KV-Diagramms.

Zum Beispiel wird die 4-stellige boolesche Funktion

$$f(w, x, y, z) = (w \wedge x \wedge \neg y \wedge z) \vee (w \wedge x \wedge y \wedge z)$$
$$\vee (\neg w \wedge x \wedge \neg y \wedge z) \vee (\neg w \wedge x \wedge y \wedge z)$$
$$\vee (\neg w \wedge \neg x \wedge \neg y \wedge z) \vee (\neg w \wedge \neg x \wedge \neg y \wedge \neg z)$$
$$\vee (w \wedge \neg x \wedge y \wedge \neg z) \vee (w \wedge \neg x \wedge \neg y \wedge \neg z)$$

durch nebenstehendes KV-Diagramm dargestellt.



Benachbarte Felder

Um mit einem KV-Diagramm einen Ausdruck zu vereinfachen, nutzen wir eine besondere Eigenschaft dieser Diagramme aus: Benachbarte Felder unterscheiden sich genau um eine Variable! Das bedeutet, benachbarte Felder repräsentieren fast den gleichen Ausdruck; die beiden Ausdrücke unterscheiden sich lediglich dadurch, dass genau eine Variable einmal negiert und einmal nicht negiert auftritt.

Beispiel: Die beiden Ausdrücke, die zu den ersten beiden Feldern der untersten Zeile von Folie 36 gehören, lauten $\neg w \wedge x \wedge \neg y \wedge z$ und $\neg w \wedge x \wedge y \wedge z$. Sie unterscheiden sich nur durch die Variable y, die einmal negiert und einmal nicht negiert vorkommt.

Benachbarte Einsen

Wenn in zwei benachbarten Feldern Einsen eingetragen sind, enthält die dargestellte Funktion die beiden entsprechenden Konjunktionen.

Beispiel: Die beiden Einsen in der untersten Zeile von Folie 36 zeigen, dass die Funktion f folgende Gestalt hat:

$$f(w, x, y, z) = \ldots \vee (\neg w \wedge x \wedge \neg y \wedge z) \vee (\neg w \wedge x \wedge y \wedge z) \vee \ldots$$

Da die beiden Ausdrücke sich nur in einer Variablen (hier in y) unterscheiden, können wir die restlichen Variablen nach dem Distributivgesetz ausklammern:

$$(\neg w \wedge x \wedge \neg y \wedge z) \vee (\neg w \wedge x \wedge y \wedge z) = (\neg w \wedge x \wedge z) \wedge (\neg y \vee y).$$

Das allgemeine Schema

Nach 3.1.1 (e) gilt $\neg y \vee y = 1$. Nach 3.1.1 (d) können wir diese durch $\wedge$ verknüpfte 1 weglassen, und es ergibt sich der vereinfachte Ausdruck

$$(\neg w \wedge x \wedge \neg y \wedge z) \vee (\neg w \wedge x \wedge y \wedge z) = \neg w \wedge x \wedge z.$$

In der Praxis müssen wir diese Schritte nicht einzeln durchführen sondern können nach folgendem Schema vorgehen:

Wir suchen nach benachbarten Einsen im KV-Diagramm. Die zugehörigen beiden Terme können dann zusammengefasst werden, indem diejenige Variable gestrichen wird, die einmal negiert und einmal nicht negiert vorkommt.

Randfelder

Dabei ist zu beachten, dass auch gegenüberliegende Randfelder als benachbart gelten sollen.

Beispiel: In diesem Sinne sind auf Folie 36 etwa die beiden Felder links und rechts oben benachbart; auch sie unterscheiden sich in genau einer Variablen, in diesem Fall in z. Die zugehörige Vereinfachung lautet

$$(\neg w \wedge \neg x \wedge \neg y \wedge z) \vee (\neg w \wedge \neg x \wedge \neg y \wedge \neg z) = \neg w \wedge \neg x \wedge \neg y.$$

Viererblöcke

Es kann vorkommen, dass man mehr als zwei benachbarte Einsen zusammenfassen kann.

Beispiel: Das KV-Diagramm auf Folie 36 zeigt links unten einen Vierer-block von Einsen. In diesem Fall können wir zunächst $\neg y \vee y = 1$ und dann $\neg w \vee w = 1$ ausklammern und wegstreichen:

$$(w \wedge x \wedge \neg y \wedge z) \vee (w \wedge x \wedge y \wedge z) \vee (\neg w \wedge x \wedge \neg y \wedge z) \vee (\neg w \wedge x \wedge y \wedge z)$$
$$= (w \wedge x \wedge z) \wedge (\neg y \vee y) \vee (\neg w \wedge x \wedge z) \wedge (\neg y \vee y)$$
$$= (w \wedge x \wedge z) \vee (\neg w \wedge x \wedge z) = (x \wedge z) \wedge (\neg w \vee w) = x \wedge z.$$

Bei einem Viererblock können also zwei Variablen gestrichen werden, nämlich die beiden, die sowohl negiert als auch nicht negiert auftreten.

Ergebnis der Vereinfachung

Bemerkungen:

1. Bei einem Achterblock können drei Variablen gestrichen werden.
2. Ein Einserfeld kann auch für mehrere Blöcke verwendet werden.

Insgesamt können wir also unsere auf Folie 36 dargestellte Funktion f wie folgt vereinfachen:

$$f(w, x, y, z) = (\neg w \wedge \neg x \wedge \neg y) \wedge (x \wedge z) \wedge (w \wedge \neg x \wedge \neg z).$$

Vorteil des KV-Verfahrens

Vorteil des Verfahrens von Karnaugh und Veitch: Keinerlei Umformungen müssen per Hand durchgeführt werden. Sämtliche Vereinfachungen kann man durch bloßes Zusammenfassen von Einserblöcken am KV-Diagramm ablesen.

Oft ist es möglich, verschiedene Einteilungen in Einserblöcke zu finden. Dann erfordert es ein wenig Geschick, die einfachste Form des Ausdrucks herauszufinden.

3.4 Logische Schaltungen

Eine wichtige Anwendung der Booleschen Algebra ist der Entwurf von logischen Schaltungen.

Eine solche logische Schaltung ist nichts weiter als eine physikalische Realisierung einer booleschen Funktion. Letztendlich ist jeder Computer aus logischen Schaltungen aufgebaut.

Die beiden Zustände 0 und 1 der Booleschen Algebra werden durch unterschiedliche elektrische Spannungen realisiert. Meist entspricht der Zustand 0 der Spannung 0 (oder einer minimalen Spannung U_{min}) und der Zustand 1 einer maximalen Spannung U_{max}. Dabei sind gewisse Toleranzbereiche um diese Spannungen erlaubt.

Gatter

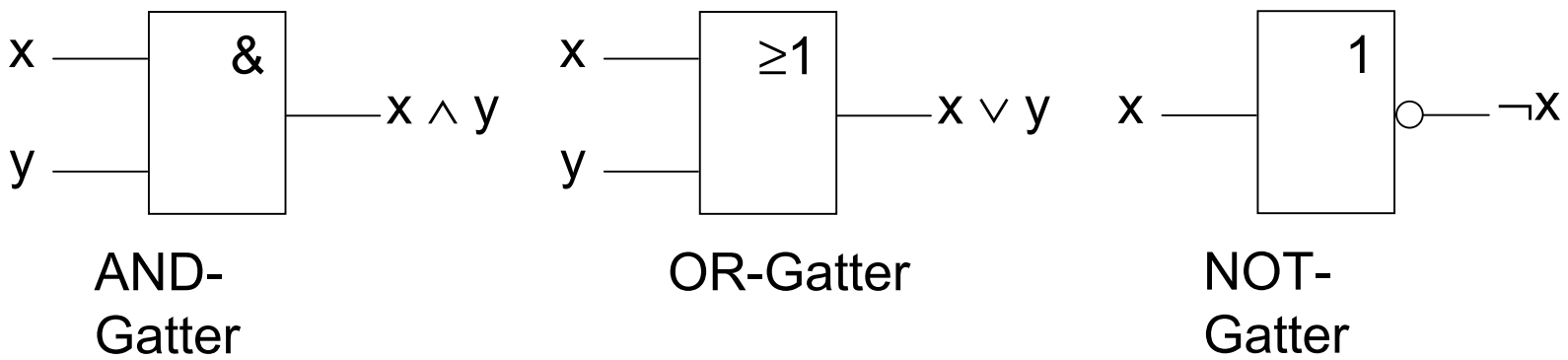
Die grundlegenden booleschen Operationen $\wedge$, $\vee$ und $\neg$ werden durch elektronische Bauteile umgesetzt, die man Gatter nennt.

Solche Gatter kann man prinzipiell mit einfachen Schaltern und Relais verwirklichen, heute werden allerdings aus Platz- und Performance-gründen Halbleiterbauelemente verwendet.

In Schaltplänen werden Gatter durch ihre jeweiligen Schaltsymbole dargestellt.

Die drei Grundgatter

Die Schaltsymbole der drei Grundgatter AND, OR und NOT, die die booleschen Grundoperationen $\wedge$, $\vee$ bzw. $\neg$ realisieren, sind in folgender Abbildung dargestellt.



AND-Gatter

OR-Gatter

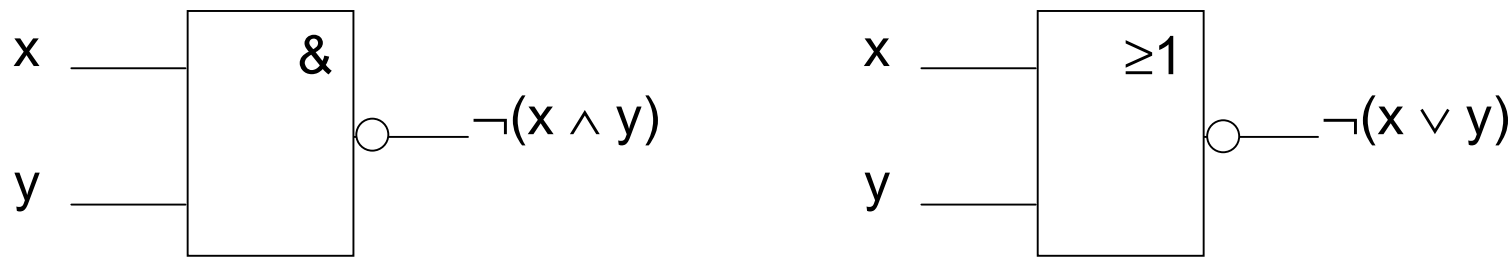
NOT-Gatter

NAND- und NOR-Gatter

Diese drei Grundgatter können geeignet hintereinandergeschaltet werden.

Zur Vereinfachung werden dabei vor- oder nachgeschaltete NOT-Gatter am Eingang bzw. am Ausgang einfach als Kreis symbolisiert.

Auf diese Weise ergeben sich die beiden Gatter zur Realisierung der NAND- und der NOR-Funktion (siehe Abschnitt 3.2) wie folgt:



NAND- und NOR-Technik

Besondere Bedeutung der NAND- und NOR-Gatter: Mit jedem von ihnen kann man alle drei Grundoperationen $\wedge$, $\vee$ und $\neg$ aufbauen!

Praktische Anwendung: Es genügt eine einzige Sorte von Bauteilen, nämlich NAND- oder NOR-Gatter, um jede beliebige boolesche Funktion zu verwirklichen.

3.4.1 Satz (NAND- und NOR-Technik). Die drei booleschen Grundoperationen Konjunktion, Disjunktion und Negation können als Hintereinanderausführung von ausschließlich NAND-Funktionen oder ausschließlich NOR-Funktionen geschrieben werden.

Beweis (I)

Der Übersichtlichkeit wegen schreiben wir $NAND(x, y) := \neg(x \wedge y)$ und $NOR(x, y) := \neg(x \vee y)$.

In NAND-Technik können wir die Operationen $\wedge$, $\vee$ und $\neg$ wie folgt ausdrücken:

$$x \wedge y = (x \wedge y) \vee 0 = \neg(\neg(x \wedge y) \wedge 1) = NAND(NAND(x, y), 1),$$

$$x \vee y = (x \wedge 1) \vee (y \wedge 1) = \neg(\neg(x \wedge 1) \wedge \neg(y \wedge 1))$$
$$= NAND(NAND(x, 1), NAND(y, 1)),$$

$$\neg x = \neg(x \wedge 1) = NAND(x, 1).$$

Beweis (II)

In NOR-Technik können wir schreiben:

$$x \wedge y = (x \vee 0) \wedge (y \vee 0) = \neg(\neg(x \vee 0) \vee \neg(y \vee 0)) = \text{NOR}(\text{NOR}(x, 0), \text{NOR}(y, 0)),$$

$$x \vee y = (x \vee y) \wedge 1 = \neg(\neg(x \vee y) \vee 0) = \text{NOR}(\text{NOR}(x, y), 0),$$

$$\neg x = \neg(x \vee 0) = \text{NOR}(x, 0).$$

Damit haben wir alles bewiesen. Im Beweis kamen im Wesentlichen die Gesetze von de Morgan (Satz 3.1.3) zur Anwendung. ❑

Realisierung von booleschen Funktionen

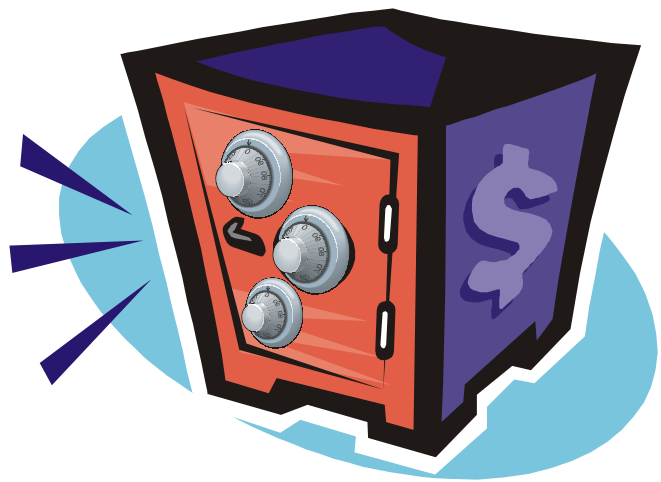
Beliebige boolesche Funktionen lassen sich durch Hintereinanderschaltung der drei Grundgatter (bzw. von NAND- oder von NOR-Gattern) wie folgt realisieren:

1. Im Allgemeinen ist zunächst die Wertetabelle der zu realisierenden Funktion aufzustellen.
2. Aus der Wertetabelle kann man die (evtl. komplizierte) disjunktive Normalform der Funktion ablesen.
3. Da man in der Praxis die Funktion mit so wenig Bauteilen wie möglich realisieren möchte, ist es sinnvoll, den booleschen Ausdruck zu vereinfachen. Dies geschieht am besten mit einem KV-Diagramm.
4. Der vereinfachte Ausdruck wird mit einer Gatterschaltung realisiert.

1. Beispiel: 2-aus-3-Schaltung

Wir wollen eine Schaltung mit drei Eingängen konstruieren, an deren Ausgang genau dann der Zustand 1 auftritt, wenn an mindestens zwei Eingängen 1 anliegt.

Ein mögliches Anwendungsbeispiel einer solchen Schaltung ist eine Tresortür, die sich nur öffnet, wenn mindestens zwei von drei Schlössern geöffnet werden.



Wertetabelle und disjunktive Normalform

Zunächst stellen wir die Wertetabelle der gesuchten Funktion f auf:

x	y	z	f(x, y, z)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Aus der Tabelle können wir die disjunktive Normalform ablesen:

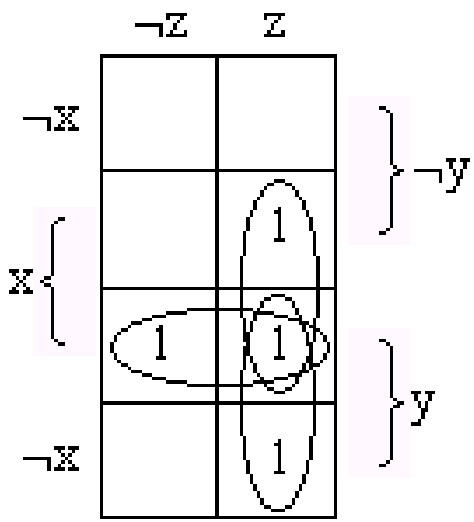
$$f(x, y, z) = (\neg x \wedge y \wedge z) \vee (x \wedge \neg y \wedge z) \vee (x \wedge y \wedge \neg z) \vee (x \wedge y \wedge z).$$

Vereinfachen mittels KV-Diagramm

Zur Vereinfachung tragen wir diese Funktion

$$f(x, y, z) = (\neg x \wedge y \wedge z) \vee (x \wedge \neg y \wedge z)$$
$$\vee (x \wedge y \wedge \neg z) \vee (x \wedge y \wedge z)$$

in ein nebenstehendes KV-Diagramm ein.



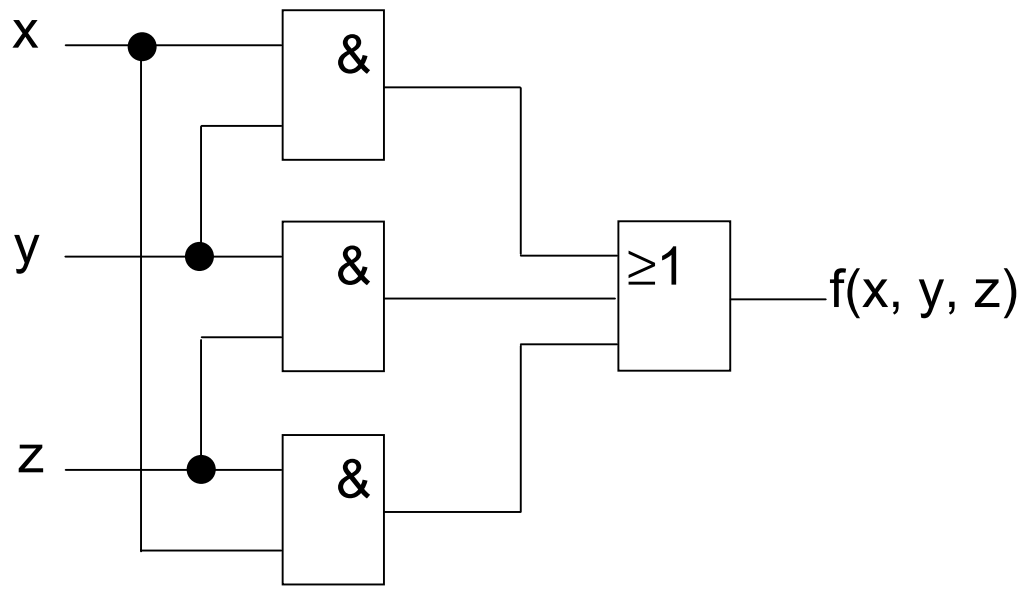
Es können drei Zweierblöcke gebildet werden, so dass der vereinfachte Ausdruck die folgende Form hat:

$$f(x, y, z) = (y \wedge z) \vee (x \wedge z) \vee (x \wedge y)$$

Die fertige Gatterschaltung

Die zu $f(x, y, z) = (y \wedge z) \vee (x \wedge z) \vee (x \wedge$

ist in folgender Abbildung dargestellt:



2. Beispiel: Halbaddierer

Logische Schaltungen können auch mehr als einen Ausgang haben. Ist dies der Fall, so muss für jeden Ausgang eine eigene boolesche Funktion aufgestellt werden. Wir wollen uns auch dies an einem Beispiel betrachten.

Wir wollen die einfachste Form einer Rechenschaltung realisieren. Sie soll zwei einstellige Binärzahlen addieren. Dabei können sich zweistellige Binärzahlen ergeben, denn falls beide Bits gleich 1 sind, entsteht ein Übertrag in die nächsthöhere Binärstelle:

$$0 + 0 = 0,\ \ 0 + 1 = 1,\ \ 1 + 0 = 1,\ \ 1 + 1 = 10.$$

Für jede der beiden Binärstellen benötigt die Schaltung einen Ausgang.

Wertetabelle und disjunktive Normalform

Wir bezeichnen die beiden Ausgänge mit s (für Summe) und ü (für Übertrag). Die Wertetabelle hat folgende Gestalt.

x	y	ü	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

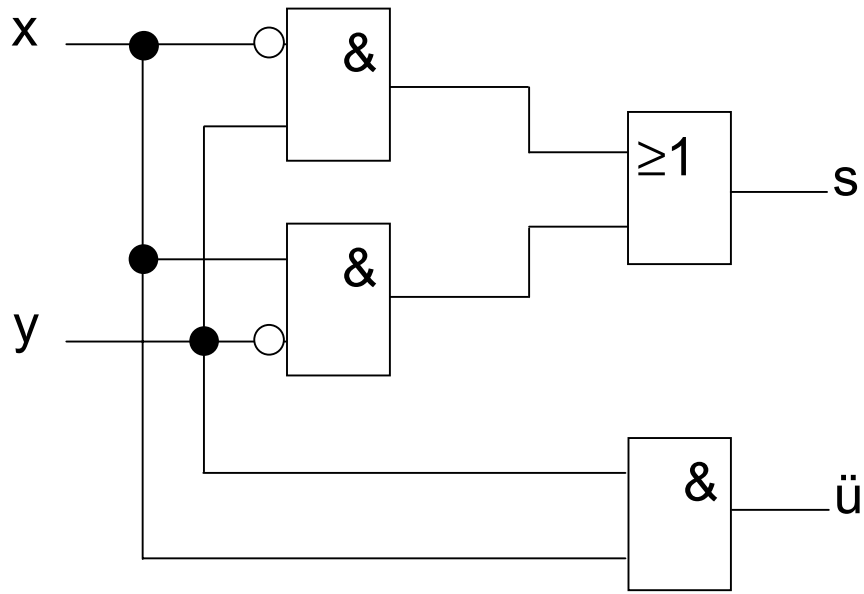
Für beide Ausgänge lesen wir die disjunktive Normalform ab:

$$ü = x \wedge y,$$

$$s = (\neg x \wedge y) \vee (x \wedge \neg y).$$

Schaltung des Halbaddierers

Diese beiden Ausdrücke können mit KV-Diagrammen nicht weiter vereinfacht werden. Daher können wir direkt die Schaltung angeben:



Volladdierer und Addierwerke

Um mehrstellige Binärzahlen zu addieren, reichen Halbaddierer nicht mehr aus. Denn zur Summe zweier Bits muss dann im Allgemeinen noch der Übertrag aus der vorherigen Stelle addiert werden. Insgesamt müssen also an jeder Stelle drei Bits addiert werden.

Die Schaltung, die drei Bits addiert, heißt Volladdierer. Aus zwei Halbaddierern und einem OR-Gatter kann man einen Volladdierer zusammensetzen (daher kommt die Bezeichnung „Halbaddierer").

Durch Zusammenschalten von $n-1$ Volladdierern und einem Halbaddierer kann man zwei n-stellige Binärzahlen addieren. Derartige Addierwerke bilden die Grundlage der heutigen Computertechnik.