

Digitaltechnik:

Skriptum zur Vorlesung

Grundlagen der Informatik Schulstufe 1

DI Dr. Sabine Strohmayer

HTBL Krems – Informationstechnologie

SJ 2011/12

Inhalt

1. Einführung	4
2. Zahlensysteme	6
Dezimalsystem.....	6
Dualzahlen	6
Hexadezimalsystem	8
3. Rechnen mit Dualzahlen	10
Umrechnung von Dual nach Dezimal bei positiven Ganzzahlen	10
Umrechnungsarten von Dezimal nach Dual bei positiven Ganzzahlen	11
Ganze positive Dualzahlen addieren	12
Ganze positive Dualzahlen subtrahieren	13
Subtraktion durch Borgen	14
Subtraktion von positiven Dualzahlen mittels Addition des Zweierkomplements.....	15
Gleichung (der Subtrahend ist größer als der Minuend)	16
Positive Dualzahlen multiplizieren.....	17
Positive Dualzahlen dividieren	18
4. Logische Verknüpfungen.....	19
5. Schaltalgebra (Boolsche Algebra)	28
Rechenregeln und Theoreme	28
Kommutativgesetz und Assoziativgesetz.....	29
Distributivgesetz.....	30
1. DeMorgansches Gesetz:.....	32
2. DeMorgansches Gesetz:.....	32
6. Schaltalgebra - Schaltungsbeispiel	40
7. KV-Diagramme	41
8. Code.....	47

1. Einführung

Die Digitaltechnik arbeitet im Gegensatz zur Analogtechnik mit diskreten anstatt kontinuierlichen Signalen.

Zudem haben die Signale meist nur einen geringen Wertevorrat, in aller Regel von zwei Werten. Diese Werte sind meist **1 (Strom)** und **0 (kein Strom)** oder **H (HIGH)** und **L (LOW)**, welche die booleschen Konstanten „Wahr“ und „Falsch“ repräsentieren.

Wenn ein High-Pegel mit 1 und ein Low-Pegel mit 0 dargestellt wird, spricht man von **positiver Logik**, bei umgekehrtem Sachverhalt, also LOW-Pegel = 1 und HIGH-Pegel = 0 von **negativer Logik**. Zusätzlich müssen in realen Schaltungen noch weitere mögliche Zustände beachtet werden. Hierzu gehören etwa der unbestimmte und der hochohmige Zustand.

Digitale Schaltungen bestehen hauptsächlich aus **Logikelementen**, wie AND, NAND, OR, NOR, NOT, XOR, XNOR und anderen, mit denen digitale Ja/Nein-Informationen miteinander verknüpft werden, z. B. im Rahmen von Zählern oder Flipflops. Komplexere Anwendungen sind Prozessoren.

Theoretisch reicht eine einzige Art (*NAND* oder *NOR*) von Gattern, dann als „Basis“ bezeichnet, um alle anderen logischen Funktionen zusammenzusetzen. Bei der Digitaltechnik wird meist unter Verwendung der **Schaltalgebra** das **Dualsystem** (entsprechend obiger Ja/Nein-Unterscheidung) zugrunde gelegt. So lässt sich für jedes Logikelement eine Schaltfunktion erstellen, die ihre Funktionsweise beschreibt. In der Praxis verwendet man meist nur NAND-Gatter, mit denen man die Funktionen der anderen Gatter nachbilden kann.

Digitale Schaltungen können zusätzlich zu logischen Funktionen auch zeitabhängige Bestandteile enthalten und ferner takt- oder zustandsgesteuert (synchron/asynchron) arbeiten. Enthält eine digitale Schaltung lediglich Logikelemente ohne Rückkopplung von Ausgängen auf Eingänge, so spricht man von einem Schaltnetz. Werden zusätzlich Speicher verwendet, oder mindestens ein Ausgang auf einen Eingang zurückgekoppelt, so handelt es sich um ein Schaltwerk oder auch einen Automaten. Ein Mikrocontroller oder Prozessor besteht hauptsächlich aus diesen Logikelementen und wird über einen Datenbus mit Speichern und anderen digitalen Baugruppen erweitert. Eine zeitlich gestaffelte Ausführung von Logikverknüpfungen ist möglich. Diese können festverdrahtet oder programmiert sein.

Bit und Byte

Ein Signal, dessen Zustand ausschließlich durch zwei Werte beschrieben wird, ist ein **binäres Signal** und wird als **BIT = BINARY DIGIT** bezeichnet. Ein **BIT** steht als Einheit für ein binäres Signal und stellt damit die kleinste informationstechnische Einheit mit zwei klaren Signalzuständen, definiert "0" oder "1", dar.

4 Bit ergeben ein Halbbyte bzw. ein Nybble. 2 Nybbles/Halbbytes ergeben ein Byte.

8 Bit werden zu einem **Byte** zusammengefasst. Ein **Byte** kann in Binärdarstellung die Werte 0 bis 255 oder die Werte -127 bis +127 annehmen. Zwei zusammengehörende Byte (2Byte = 16 Bit) sind ein digitales Wort (**WORD**). 1 Wort = 2 Byte = 16 Bit. Zwei Wörter die zusammengehören ergeben ein digitales Doppelwort (**DWORD**). 1 Doppelwort = 2 Wörter = 4 Byte = 32 Bit.

2. Zahlensysteme

Dezimalsystem

- Ziffern: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- Basis: 10
- Stellenwerte: Potenzen der Basis 10

Beispiel: Die Zahl 289 besteht aus 9×1 (Einer, 10^0), 8×10 (Zehner, 10^1) und 2×100 (Hunderter, 10^2).

$$289 = 2 \cdot 10^2 + 8 \cdot 10^1 + 9 \cdot 10^0$$

Das Dezimalsystem beruht auf der Basis 10. Die Basis bestimmt auch die Anzahl der Ziffern im Zahlensystem (0 bis 9).

Die Stellenwerte werden durch die Potenzen zur Basis gebildet.

Dualzahlen

Das ist für uns das wichtigste Zahlensystem, da die Funktionsweise jedes Computers und jeder digitalen Schaltung auf dem Dualsystem basiert. Die Basis bestimmt auch die Anzahl der Ziffern im Zahlensystem (0 und 1).

- Ziffern: 0, 1
- Basis: 2
- Stellenwerte: Potenzen der Basis 2

Beispiel: Die Zahl $6 = 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 4 + 2 + 0 =$ in binärer Darstellung **110**

Zur besseren Übersicht teilt man bei größeren Dualzahlen diese in Vierergruppen auf.

Beispiel: 0100 0111 0110 1101

Übersicht der Dezimalzahlen von 1 bis 15 und der zugehörigen Dualzahlen

Dezimalzahl	Dualzahl (mit 4 Bit)
-------------	----------------------

0	0000
---	------

1	0001
---	------

2	0010
---	------

3	0011
---	------

4	0100
---	------

5	0101
---	------

6	0110
---	------

7	0111
---	------

8	1000
---	------

9	1001
---	------

10	1010
----	------

11	1011
----	------

12	1100
----	------

13	1101
----	------

14	1110
----	------

15	1111
----	------

Hexadezimalsystem

Das Hexadezimalsystem beruht auf der Basis 16. Die Basis bestimmt die Anzahl des Zeichenvorrates.

Im Hexadezimalsystem sind es die Ziffern 0 bis 9 und die Buchstaben A bis F.

- Ziffern und Zeichen: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- Basis: 16
- Stellenwerte: Potenzen der Basis 16

Die Buchstaben haben folgende Bedeutung: A=10, B=11, C=12, D=13, E=14, F=15

Das Hexadezimalsystem wird benutzt um mit möglichst wenig Ziffern und Zeichen große Zahlen darzustellen.

Beispiel: Umwandeln der Dezimalzahl 512 in eine Hexadezimalzahl.

Dazu dividiert man die Zahl 512 durch die Basis 16, man erhält die Zahl 32. Der Rest ist **0**.

$$512/16 = 32 \text{ (Rest 0)}$$

Diese **0** stellt die erste Hexadezimalzahl **links vom Komma** dar. Dann wird die 32 durch 16 geteilt, man erhält die 2 und als Rest wieder **0**.

$$32/16 = 2 \text{ (Rest 0)}$$

Als nächstes dividiert man die 2 durch die Basis 16 und erhält als Rest **2**.

$$2/16 = \dots \text{ (Rest 2)}$$

Der Rest 2 entsteht weil die Division von 2 durch 16 kein ganzzahliges Ergebnis ergibt. Nun fasst man die Ergebnisse zusammen und erhält die Hexadezimalzahl **200**.

Die Hexadezimalzahl **200** hat nun folgende Stellenwerte:

beginnend von rechts nach links $0 \times 16^0 = 0 \times 1 = 0$ (erste Stelle links vom Komma), $0 \times 16^1 = 0 \times 16 = 0$ (zweite Stelle links vom Komma), $2 \times 16^2 = 2 \times 256 = 512$.

Die hexadezimale Darstellung von Zahlen in der Steuerungstechnik ist eine weitverbreitete Kurzschreibweise für die als Grundlage dienenden Dualzahlen. Ein Byte sind 8 Bit und werden z.B. mit 1111 1111 dargestellt.

Die gleiche Zahl hexadezimal dargestellt ist FF. Durch die hexadezimale Zahlendarstellung wird nicht der Zahlenumfang des Dualsystems verändert, man erreicht dadurch eine strukturierte Lesart da immer 4 Bit zu einer Einheit zusammengefasst werden.

Die hexadezimale Zahlendarstellung dient also auch dem Zweck größere Dualzahlen

einfacher darzustellen. Die Dualzahl **1101 0110 1000 0101 0001 0111** wird in 4-Bit Einheiten zerlegt und dann als **D68517** dargestellt. Aus diesem Beispiel wird die Vereinfachung ersichtlich und es ergibt sich eine weniger fehleranfällige Darstellung der Zahl als wie beim Dualsystem.

Das hexadezimale Zahlensystem wird in der Steuerungstechnik bei technischen Prozessen bei Zahleneinstellern und Ziffernanzeigern und programmintern bei Maskierungen in Verbindung mit ODER- bzw. UND-Wortbefehlen zum Ein- und Ausblenden von Binärstellen in Wort-Operanden verwendet.

Gegenüberstellung Dezimal-, Dual- und Hexadezimalsystem

Dezimalsystem			Dualsystem								Hexadezimalsystem		
10^2	10^1	10^0	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	16^2	16^1	16^0
		0	0	0	0	0	0	0	0	0	0	0	0
		1	0	0	0	0	0	0	0	1	0	0	1
		2	0	0	0	0	0	0	1	0	0	0	2
		3	0	0	0	0	0	0	1	1	0	0	3
		4	0	0	0	0	0	1	0	0	0	0	4
		5	0	0	0	0	0	1	0	1	0	0	5
		6	0	0	0	0	0	1	1	0	0	0	6
		7	0	0	0	0	0	1	1	1	0	0	7
		8	0	0	0	0	1	0	0	0	0	0	8
		9	0	0	0	0	1	0	0	1	0	0	9
	1	0	0	0	0	0	1	0	1	0	0	0	A
	1	1	0	0	0	0	1	0	1	1	0	0	B
	1	2	0	0	0	0	1	1	0	0	0	0	C
	1	3	0	0	0	0	1	1	0	1	0	0	D
	1	4	0	0	0	0	1	1	1	0	0	0	E
	1	5	0	0	0	0	1	1	1	1	0	0	F
	1	6	0	0	0	1	0	0	0	0	0	1	0
	1	7	0	0	0	1	0	0	0	1	0	1	1
	.	.											
	5	2	0	0	1	1	0	1	0	0	0	3	4
	.	.											
	9	5	0	1	0	1	1	1	1	1	0	5	F
	.	.											
2	1	4	1	1	0	1	0	1	1	0	0	D	6

3. Rechnen mit Dualzahlen

Umrechnung von Dual nach Dezimal bei positiven Ganzzahlen

Beispiel für die Umrechnung der Dualzahl 1010 in die Dezimalzahl 10

Die Ziffern werden mit ihren Stellenwertigkeiten ausmultipliziert und die Ergebnisse werden addiert.

Potenzen dual	2^3	2^2	2^1	2^0
Dualzahl	1	0	1	0
Potenzen dezimal	$2^3 = 2 \times 2 \times 2$	$2^2 = 2 \times 2$	$2^1 = 2$	$2^0 = 1$
zu addierende Werte	8	0	2	0
Ergebnis Dezimalzahl	10			

Umrechnungsarten von Dezimal nach Dual bei positiven Ganzzahlen

Bei der Umrechnung von Dezimalzahlen in Dualzahlen gibt es zwei Verfahrensweisen:

a)

Nach dem Restwertalgorithmus wird die Dezimalzahl durch die Basis 2 dividiert und der Quotient und der Rest, 0 oder 1, werden notiert. Der Rest 1 ergibt sich immer dann wenn die positive dezimale Ganzzahl nicht durch die Basis 2 teilbar ist, also bei allen ungeraden positiven Dezimalzahlen. Der Quotient wird erneut durch die Basis dividiert und wieder wird ein Quotient und der Rest gebildet. Dieses Verfahren setzt man so lange fort bis der Quotient 0 wird.

Verfahrensweise am Beispiel der **10**:

$10 : 2 = 5$, Rest 0 -> 1. Stelle links vom Komma

$5 : 2 = 2$, Rest 1 -> 2. Stelle links vom Komma

$2 : 2 = 1$, Rest 0 -> 3. Stelle links vom Komma

$1 : 2 = 0$, Rest 1 -> 4. Stelle links vom Komma

Jetzt schreibt man die Dualzahl auf: **1010**

b)

Bei dem Verfahren durch die Darstellung in Zweierpotenzen wird von der Zahl die größtmögliche Zweierpotenz subtrahiert und dann die Ziffer 1 an die entsprechende Stelle der Dualzahl notiert. Nun subtrahiert man vom Rest die nächst kleinere Zweierpotenz, die zu einem positiven Ergebnis führt. An der entsprechenden Stelle der Dualzahl wird eine 1 notiert.

Verfahrensweise am Beispiel der **10**:

Die größte mögliche Zweierpotenz bei der Dezimalzahl 10 ist die $2^3 = 8$. Also muß man an der vierten Stelle links vom Komma, an der Stelle der 2^3 , eine 1 schreiben. $10 - 8 = 2$, das nächste positive Ergebnis. $2^1 = 2$. Demzufolge steht also an der zweiten Stelle links vom Komma eine 1. Die 10 ist nun komplett in Zweierpotenzen zerlegt und man kann die Dualzahl notieren, **1010**

Ganze positive Dualzahlen addieren

Dualzahlen werden nach folgenden Regeln addiert:

- Stellenweise von rechts, von der kleinsten Stelle, nach links zum größten Stellenwert
- Bei einer Überschreitung des Ziffernvorrates (z.B. $1+1$) wird ein Übertrag bei der nächsthöheren, nächsten Stelle links gebildet.

Additionsregeln:

- $0+0=0$
- $0+1=1$
- $1+0=1$
- $1+1=10$ (Übertrag von 1 an der nächsten Stelle)
- $1+1+1=11$ (Übertrag von 1 an der nächsten Stelle)
- $1+1+1+1=100$ (Übertrag von 1 an der übernächsten Stelle)

Werden zwei oder drei Einsen addiert führt dies zu einem Übertrag von "1", der bei der nächsthöheren Stelle zu berücksichtigen ist oder als nächsthöhere Stelle ganz links angeschrieben wird. Werden vier Einsen addiert führt dies zu einem Übertrag an der übernächsten Stelle bzw. der Übertrag wird als "10" ganz links angeschrieben.

Beispiel: $1001 + 1111$

Man addiert zuerst die Stellen ganz rechts, $1 + 1 = 10$. Beim **Ergebnis** steht also an der kleinsten, **ersten** (ganz rechts) **Stelle** eine **0**. Der Übertrag von 1 wird nun beim addieren der zweiten Stelle der beiden Dualzahlen berücksichtigt, $0 + 1 + 1(\text{Übertrag}) = 10$. Im **Ergebnis** steht also an der **zweiten Stelle** links vom Komma eine **0** und der Übertrag ist bei der Addition der dritten Stelle zu berücksichtigen. Addieren der dritten Stelle, $0 + 1 + 1(\text{Übertrag}) = 10$. Im **Ergebnis** steht an der **dritten Stelle** wieder eine **0**. Addieren der vierten Stelle, $1 + 1 + 1$.

$1 + 1$ ergibt 10. Dazu wird noch die dritte 1 vom Übertrag addiert, $10 + 1$. Man rechnet aber in diesem Fall nur mit der Null des Übertrages, hat also im **Ergebnis** an **vierter Stelle** eine **1** stehen, da laut Regel $0 + 1 = 1$. Nun muß der Übertrag noch ganz links angeschrieben werden und man erhält die Summe der beiden Dualzahlen.

$$1001 + 1111 = 1\ 1000$$

Jetzt kann man zur Überprüfung des Ergebnisses die Dualzahlen in Dezimalzahlen umwandeln.

$$1001 = 9, 1111 = 15, \text{ daraus folgt } 9 + 15 = 24.$$

$11000 = 24$. Das Ergebnis stimmt also.

Ganze positive Dualzahlen subtrahieren

Dualzahlen werden nach folgenden Regeln subtrahiert:

- Stellenweise von rechts, von der kleinsten Stelle, nach links zum größten Stellenwert
- Ermitteln der Differenz zwischen dem Subtrahenten und dem Minuenden. Ist der Minuend kleiner als der Subtrahend, so beträgt die Differenz 1 und der Übertrag 1 (für den nächsthöheren Stellenwert).

Daraus ergeben sich folgende Subtraktionsregeln:

- $0 - 0 = 0$
- $1 - 0 = 1$
- $1 - 1 = 0$
- $0 - 1 = 1$ (mit Übertrag 1)
- $10 - 1 = 1$

Das Ergebnis bei $0 - 1$ führt zu einem Übertrag der bei der nächsthöheren Stelle zu beachten ist.

Beispiel:

1111	(Minuend)
-1001	(Subtrahend)
=0110	(Differenz)

Man subtrahiert zuerst die Stellen ganz rechts, $1 - 1 = 0$. An der **ersten Stelle** ganz rechts im **Ergebnis** steht also eine **0**. Nun die zweiten Stellen, $1 - 0 = 1$. An der **zweiten Stelle** im **Ergebnis** steht also eine **1**. Nun die dritten Stellen, $1 - 0 = 1$. An der **dritten Stelle** im **Ergebnis** steht also auch eine **1**. Nun die vierte Stelle, $1 - 1 = 0$. An der **vierten Stelle** im **Ergebnis** steht eine **0**.

$$1111 - 1001 = 0110$$

Die erste Null bei diesem Ergebnis kann auch weggelassen werden. -> 110

Zur Überprüfung der Subtraktion werden die Dualzahlen in Dezimalzahlen gewandelt.

$$1111 = 15, 1001 = 9$$

$$15 - 9 = 6.$$

$$6 = 0110$$

Subtraktion durch Borgen

Hat man einen dualen Subtrahenden der größer ist als der duale Minuend, dann ist das Ergebnis eine negative Differenz. In diesem Fall 'borgt' man sich eine Einheit von der nächsthöheren Stelle aus. Diese geborgte Einheit hat die Wertigkeit der Basis des Dualsystems, also 10.

Beispiel:

$$\begin{array}{r} 0111 \\ - 1111 \\ \hline = 101000 \end{array}$$

Hier liegt der Fall vor das das Ergebnis negativ ist. Das bedeutet das die Ziffer die am weitesten links steht, das höchstwertigste Bit (MSB - Most Significant Bit) ein Vorzeichenbit ist. Eine 0 bedeutet das die Dualzahl positiv ist, eine 1 bedeutet das die Dualzahl negativ ist. Diese Subtraktion in Dezimalzahlen notiert sähe so aus:

$$7 - 15 = - 8$$

Die Dualzahl 101000 als vorzeichenbehaftete Zahl ergibt nach der Umwandlung in das Dezimalsystem - 8.

Subtraktion von positiven Dualzahlen mittels Addition des Zweierkomplements

Das Einer-Komplement einer Dualzahl wird gebildet durch die Negation der Dualzahl, die 0 wird durch eine 1 ersetzt und die 1 durch eine 0.

Beispiel:

0110 (Dualzahl)

1001 (Einer-Komplement der Dualzahl 0110)

Das Zweier-Komplement einer Dualzahl wird gebildet durch die Negation der Dualzahl (Einer-Komplement-Bildung) und Addition von 1. Beispiel: Die Dualzahl 1001 soll von der Dualzahl 1111 subtrahiert werden (positives Ergebnis, der Minuend ist größer als der Subtrahend).

Gleichung (der Subtrahend ist kleiner als der Minuend):

1111 (Minuend)
-1001 (Subtrahend)
=xxxx (Ergebnis)

bilden des Zweier-Komplements des Subtrahenden:

1001 (Subtrahend, gegebenenfalls den Subtrahenden mit Nullen erweitern)
0110 (Einer-Komplement der Dualzahl 1001₂)
+1 (addieren von 1 zur Zweier-Komplementbildung)
=0111 (Zweier-Komplement der Dualzahl 1001)

Somit lautet die Gleichung dann:

1111
+0111
=10110

Ergebnis: 110

Die Komplementaddition ergibt das Ergebnis der Subtraktion. Die 1 ganz links, der Übertrag, wird in diesem Fall gestrichen. Die führenden Nullen kann man weglassen und man erhält das Ergebnis der Subtraktion -> 110.

Subtrahiert man zwei positive n-stellige Dualzahlen bei denen der Subtrahend kleiner ist als der Minuend, ergibt sich ein Übertrag von n+1. Nach der Streichung dieses Übertrages erhält man das Ergebnis als positive Dualzahl.

Gleichung (der Subtrahend ist größer als der Minuend)

101011 (Minuend, entspricht der Dezimalzahl 43)
-111000 (Subtrahend, entspricht der Dezimalzahl 56)
=xxxxxx (Ergebnis)

bilden des Zweierkomplements des Subtrahenden:

111000 (Subtrahend, 56)
000111 (Einer-Komplement der Dualzahl 111000)
+ 1 (addieren von 1 zur Zweier-Komplementbildung)
=001000 (Zweier-Komplement der Dualzahl 000111)

Somit lautet die Gleichung dann:

0101011 (vorzeichenbehafteter Minuend)
+0001000 (Komplementaddition der Dualzahl 111000, vorzeichenbehaftet)
= 110011 (vorzeichenbehaftetes Zwischenergebnis)

Ergebnis: 110011 (vorzeichenbehaftete Dualzahl, MSB = 1, negative Dualzahl)

Für das richtige Ergebnis muß die negative Dualzahl rekomententiert werden.

110011 (Zwischenergebnis)
001100 (Einer-Komplement von 110011)
+ 1 (addieren von 1 zur Rekomententierung)
= 1101 (Ergebnis = -13)

Die Subtraktion in Dezimalzahlen umgewandelt lautet $43 - 56 = -13$

Das Komplement einer Zahl kann als negativer Wert dieser Zahl angesehen werden. Mittels der Komplementbildung können positive Dualzahlen in negative Dualzahlen umgewandelt werden.

Ist eine Dualzahl als vorzeichenbehaftet definiert, dann gilt das positive Dualzahlen durch eine Null gekennzeichnet sind und negative Dualzahlen durch eine Eins.

+11(dezimal) entspricht 01011
dazu das Einer-Komplement: 10100
+ 1 zur Bildung des Zweier-Komplements: 10101
10101 ist die vorzeichenbehaftete duale Darstellung für die negative Dezimalzahl -11

Natürlich kann die negative Dualzahl 10101 durch Komplementbildung wieder in die positive Dualzahl 01011 umgewandelt werden.

Positive Dualzahlen multiplizieren

Multiplikationsregeln:

1. $0 \cdot 0 = 0$
2. $0 \cdot 1 = 0$
3. $1 \cdot 0 = 0$
4. $1 \cdot 1 = 1$

Genau so wie bei der Multiplikation von Dezimalzahlen werden Dualzahlen durch Addition des stellenverschobenen Multiplikanten miteinander multipliziert.

Beispiel für die Multiplikation von $10 \cdot 100$

$$\begin{array}{r} 10 \cdot 100 \\ + \quad 00 \\ + \quad 00 \\ + \quad 10 \\ = \quad 1000 \end{array}$$

$$1000 = 8$$

Man multipliziert die duale Zahl 100 ziffernweise mit der dualen Zahl 10. Der erste Schritt wäre dann die 0 (ganz rechts, erste Stelle links vom Komma, von der 100) multipliziert mit 10 ergibt nach den Multiplikationsregeln 0. Das Ergebnis wird unter die 0 (ganz rechts, erste Stelle links vom Komma, von der 100) von 100 geschrieben. Das Ergebnis ist eigentlich 00, aber es reicht wenn man eine 0 schreibt um die Übersicht bei größeren Dualzahlen zu behalten. Dann multipliziert man die mittlere 0 der 100 mit der 10 und das Ergebnis, 0, wird unter die mittlere 0 der 100 geschrieben. Dasselbe macht man mit der 1 der 100.

Dieses Ergebnis wird ebenfalls dann unter die 1 der 100 geschrieben. Nun addiert man die Ergebnisse nach den Regeln für die Addition von Dualzahlen und notiert die Summe. Bei der Multiplikation von Dualzahlen muß man immer beachten, das jedes Ergebnis der Teilmultiplikationen immer eine Stelle nach links rückt (Shifting). Multiplikation von Dualzahlen ist eine Kombination von Shifting und Addition.

Positive Dualzahlen dividieren

Beispiel für die Division der Dualzahlen 1 1110 und 1010. Der Divisor wird durch den Dividenden dividiert und man erhält den Quotienten.

Dividend \div Divisor = Quotient

$$1\ 1110 \div 1010 = 11$$

Bei der Division wird geprüft, wie oft der Divisor im Dividenden enthalten ist. Als Ergebnis erhält man den Quotienten und meistens einen Rest. Die einfachste Art der Division fußt auf der fortgesetzten Subtraktion des Divisors vom Dividenden wobei sich der Quotient stellenweise ergibt. Der Divisor wird so oft vom Dividenden stellenrichtig abgezogen, bis der Rest kleiner als der Divisor ist, und damit die Differenz negativ wird. Die Anzahl der Subtraktionen entspricht dem Wert der Quotientenstelle, solange die Differenz noch positiv ist.

Man braucht also nur zu prüfen, ob der Divisor vom Betrag her kleiner oder größer als der Dividend ist.

- Die Quotientenstelle erhält eine "0", wenn der Divisor größer als der Dividend ist.
- Die Quotientenstelle erhält eine "1", wenn der Divisor kleiner als der Dividend ist. In diesem Fall wird der Divisor vom Dividenden subtrahiert und die nächste Stelle wird an den betreffenden Dividenden angehängt.

$$\begin{array}{r} 11110 \div 1010 = 11 \\ - \quad 1010 \\ = \quad 01010 \\ - \quad 1010 \\ = \quad 0000 \end{array}$$

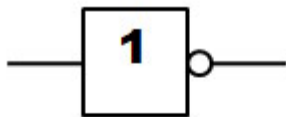
Die Division bei den Dualzahlen basiert auf der gleichen Vorgehensweise wie die Division bei den Dezimalzahlen. Man schreibt den Divisor unter den Dividenden und führt Subtraktionen durch.

4. Logische Verknüpfungen

Eine logische Verknüpfung beschreibt das Verhalten des Ausgangszustandes in Abhängigkeit der Eingangszustände.

Die einfachste Verknüpfung ist die NICHT-Verknüpfung (NICHT-Gatter). Hier wird der Eingang einfach invertiert, d.h. der Ausgang ist das Gegenteil des Einganges. Ist der Eingang 1, so ist der Ausgang 0 und umgekehrt.

DIN-Schaltzeichen, Wahrheitstabelle und Funktionsgleichung eines NICHT-Gatters



A	Z
0	1
1	0

 $\longrightarrow Z = \bar{A}$

In der Wahrheitstabelle werden alle möglichen Eingangszustände für den Eingang A (0 und 1) eingetragen. Damit ergeben sich die Ausgangszustände für den Ausgang Z.

Dies liefert uns die Funktionsgleichung $Z = A'$ (A' ist eine andere Schreibweise für A-nicht und gleichbedeutend wie A mit dem Querstrich).

Genormte Schreibweise:

Zeichen für und:

\wedge

Zeichen für oder:

\vee

A negiert (A-Nicht):

\bar{A}

B und A negiert (B und A-NICHT):

$B \wedge \bar{A}$

(A und B) negiert (A und B)-Nicht:

$\overline{A \wedge B}$

(A oder B) negiert (A oder B)-Nicht:

$\overline{A \vee B}$

DIN-Schaltzeichen, Wahrheitstabelle und Funktionsgleichung eines UND-Gatters



	A	B	Z
0	0	0	0
1	0	1	0
2	1	0	0
3	1	1	1

 $\longrightarrow Z = A \wedge B$

Die UND-Funktion liefert uns nur dann am Ausgang eine 1, wenn beide Eingänge auf 1 sind. Ist einer von beiden Eingängen auf 0, kann auch der Ausgang nicht mehr 1 sein.

In der Wahrheitstabelle werden auch hier alle möglichen Eingangszustände für den Eingang A und B (0 und 1) eingetragen. Damit ergeben sich die Ausgangszustände für den Ausgang Z.

Dies liefert uns die Funktionsgleichung $Z = A * B$ (andere Schreibweise für $Z = A \text{ und } B$)

DIN-Schaltzeichen, Wahrheitstabelle und Funktionsgleichung eines ODER-Gatters



	A	B	Z
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	1

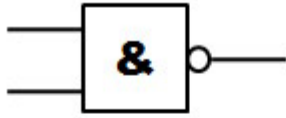
 → $Z = A \vee B$

Die ODER-Funktion liefert uns nur dann am Ausgang eine 1, wenn einer der beiden Eingänge auf 1 ist. Sind beide Eingänge auf 0, kann auch der Ausgang nicht mehr 1 sein.

In der Wahrheitstabelle werden auch hier alle möglichen Eingangszustände für den Eingang A und B (0 und 1) eingetragen. Damit ergeben sich die Ausgangszustände für den Ausgang Z.

Dies liefert uns die Funktionsgleichung $Z = A + B$ (andere Schreibweise für $Z = A \text{ oder } B$)

DIN-Schaltzeichen, Wahrheitstabelle und Funktionsgleichung eines NAND-Gatters



	A	B	Z
0	0	0	1
1	0	1	1
2	1	0	1
3	1	1	0

 $\longrightarrow Z = \overline{A \wedge B}$

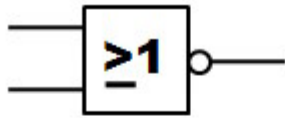
Die NAND-Funktion liefert uns nur dann am Ausgang keine 1, wenn beide Eingänge auf 1 ist. Sind beide Eingänge auf 0 oder auch nur ein Eingang auf 1, ist der Ausgang 1.

Die NAND-Funktion ist demnach eine invertierte UND-Funktion, wie auch am Schaltzeichen erkennbar.

In der Wahrheitstabelle werden auch hier alle möglichen Eingangszustände für den Eingang A und B (0 und 1) eingetragen. Damit ergeben sich die Ausgangszustände für den Ausgang Z.

Dies liefert uns die Funktionsgleichung $Z = (A * B)'$ (andere Schreibweise für $Z = A$ und B nicht)

DIN-Schaltzeichen, Wahrheitstabelle und Funktionsgleichung eines NOR-Gatters



	A	B	Z
0	0	0	1
1	0	1	0
2	1	0	0
3	1	1	0

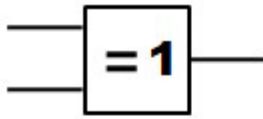
 $\longrightarrow Z = \overline{A \vee B}$

Die NOR-Funktion (NICHT-ODER) ist eine am Ausgang negierte ODER-Funktion und liefert uns nur dann am Ausgang eine 1, wenn beide Eingänge auf 0 sind. Ist einer der beiden Eingänge auf 1, kann auch der Ausgang nicht mehr 1 sein.

In der Wahrheitstabelle werden auch hier alle möglichen Eingangszustände für den Eingang A und B (0 und 1) eingetragen. Damit ergeben sich die Ausgangszustände für den Ausgang Z.

Dies liefert uns die Funktionsgleichung $Z = (A + B)'$ (andere Schreibweise für $Z = A$ oder B nicht)

DIN-Schaltzeichen, Wahrheitstabelle und Funktionsgleichung eines XOR-Gatters



	A	B	Z
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	0

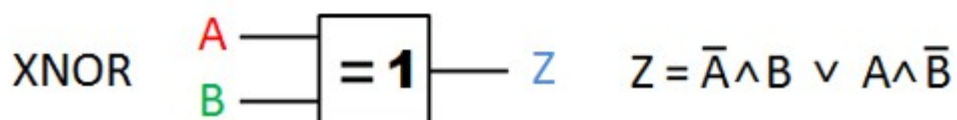
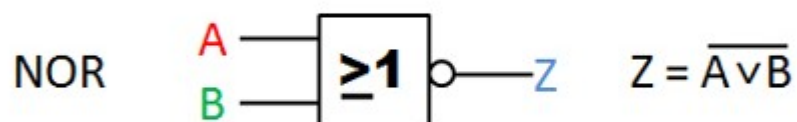
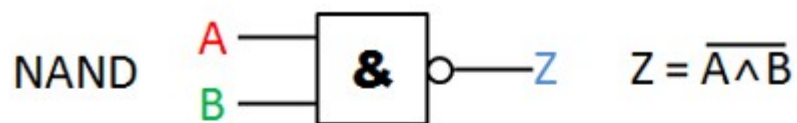
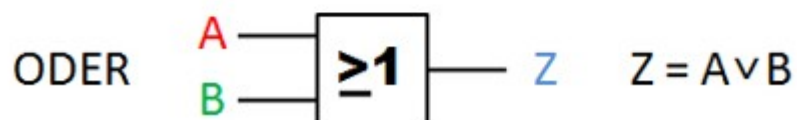
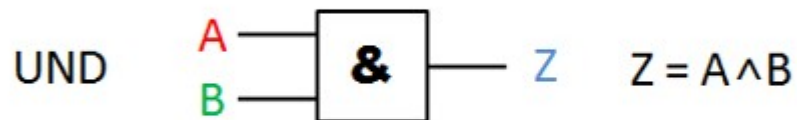
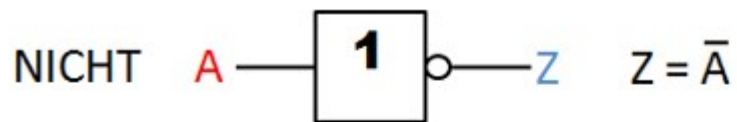
$$\longrightarrow Z = \bar{A} \wedge B \vee A \wedge \bar{B}$$

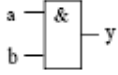
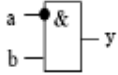
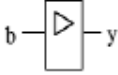
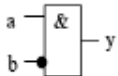

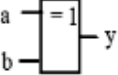

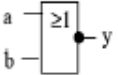



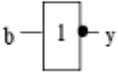


Die XOR-Funktion (EXCLUSIV-ODER) liefert uns nur dann am Ausgang eine 1, wenn beide Eingänge unterschiedliche Zustände also 0 oder 1 bzw. 1 oder 0 haben. Sind beide Eingänge auf 0 oder 1 ist der Ausgang 0.

In der Wahrheitstabelle werden auch hier alle möglichen Eingangszustände für den Eingang A und B (0 und 1) eingetragen. Damit ergeben sich die Ausgangszustände für den Ausgang Z.

Dies liefert uns die Funktionsgleichung $Z = (A' * B) + (A * B')$ (andere Schreibweise für $Z = (A \text{ nicht und } B) \text{ oder } (A \text{ und } B \text{ nicht})$)

Übersichtstabelle der wichtigsten logischen Verknüpfungen:



Name	Logiksymbol	Schaltfunktion	Kontaktäquivalent	Bemerkung
NULL		$y = 0$	$1 \text{ --- } y$	Konstante Null
UND		$y = a \wedge b$	$1 \text{ --- } \begin{array}{c} \diagup \\ a \end{array} \text{ --- } \begin{array}{c} \diagdown \\ b \end{array} \text{ --- } y$	a und b
Inhibition		$y = \bar{a} \wedge b$	$1 \text{ --- } \begin{array}{c} \text{---} \\ \bar{a} \end{array} \text{ --- } \begin{array}{c} \diagdown \\ b \end{array} \text{ --- } y$	b aber nicht a
Transfer (Identität)		$y = b$	$1 \text{ --- } \begin{array}{c} \diagup \\ b \end{array} \text{ --- } y$	b
Inhibition		$y = a \wedge \bar{b}$	$1 \text{ --- } \begin{array}{c} \diagup \\ a \end{array} \text{ --- } \begin{array}{c} \text{---} \\ \bar{b} \end{array} \text{ --- } y$	a aber nicht b
Transfer (Identität)		$y = a$	$1 \text{ --- } \begin{array}{c} \diagdown \\ a \end{array} \text{ --- } y$	a
Antivalenz (Exklusiv-ODER)		$y = \bar{a}b \vee a\bar{b}$	$1 \text{ --- } \begin{array}{c} \diagup \\ \bar{a} \end{array} \text{ --- } \begin{array}{c} \diagdown \\ b \end{array} \text{ --- } y$	a oder b, aber nicht beide
ODER		$y = a \vee b$	$1 \text{ --- } \begin{array}{c} \diagup \\ a \end{array} \text{ --- } \begin{array}{c} \diagdown \\ b \end{array} \text{ --- } y$	a oder b
NOR		$y = \overline{a \vee b}$	$1 \text{ --- } \begin{array}{c} \text{---} \\ \bar{a} \end{array} \text{ --- } \begin{array}{c} \text{---} \\ \bar{b} \end{array} \text{ --- } y$	Nicht - ODER
Äquivalenz Exklusiv-NOR		$y = ab \vee \bar{a}\bar{b}$	$1 \text{ --- } \begin{array}{c} \diagup \\ a \end{array} \text{ --- } \begin{array}{c} \diagdown \\ b \end{array} \text{ --- } y$	genau wenn a, dann b (a gleich b)
Negation, Komplement		$y = \bar{a}$	$1 \text{ --- } \begin{array}{c} \text{---} \\ a \end{array} \text{ --- } y$	Nicht a
Implikation		$y = \bar{a} \vee b$	$1 \text{ --- } \begin{array}{c} \text{---} \\ \bar{a} \end{array} \text{ --- } \begin{array}{c} \diagdown \\ b \end{array} \text{ --- } y$	wenn a, dann b
Negation		$y = \bar{b}$	$1 \text{ --- } \begin{array}{c} \text{---} \\ \bar{b} \end{array} \text{ --- } y$	Nicht b
Implikation		$y = a \vee \bar{b}$	$1 \text{ --- } \begin{array}{c} \diagup \\ a \end{array} \text{ --- } \begin{array}{c} \text{---} \\ \bar{b} \end{array} \text{ --- } y$	wenn b, dann a
NAND		$y = \overline{a \wedge b}$	$1 \text{ --- } \begin{array}{c} \text{---} \\ \bar{a} \end{array} \text{ --- } \begin{array}{c} \text{---} \\ \bar{b} \end{array} \text{ --- } y$	Nicht - UND
EINS		$y = 1$	$1 \text{ --- } y$	Konstante Eins

5. Schaltalgebra (Boolsche Algebra)

Wir nehmen an, es soll auf Grund einer verbalen Vorgabe eine Schaltung zu entwerfen sein, die ein vorgegebenes Verhalten zeigen soll.

Der erste Schritt wird es nun sein, eine Wahrheitstabelle aufzustellen und daraus die Formel für jeden der Ausgänge abzuleiten. In vielen Fällen wird der Versuch, die Schaltung zu diesem Zeitpunkt zu zeichnen ein ziemlich kompliziertes Vorgehen sein. Bei komplexen Schaltungen wird dies auch durch probieren in vertretbarer Zeit nicht zu lösen sein.

Hier setzt nun die **Schaltalgebra** an, mit deren Hilfe man die Formeln rechnerisch auf die **kleinstmögliche Schaltung minimieren** kann.

Benannt nach dem englischen Mathematiker Boole wird die boolsche Algebra in der Digital- und Rechnertechnik verwendet.

Rechenregeln und Theoreme

Die Schaltalgebra kennt VARIABLE und KONSTANTEN.

Die KONSTANTEN sind 0 und 1 entsprechend der logischen Zustände 0 und 1.

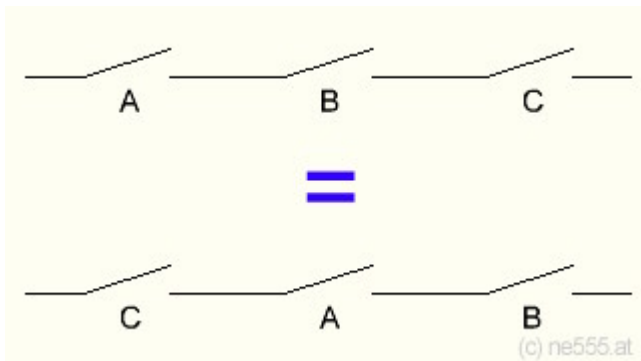
Die Eingänge eines Gatters sind z.B. Variablen, da Sie den Wert 0 und 1 annehmen können. Eine Konstante ist z.B. eine Steckbrücke, die immer den fixen Wert 0 oder 1 liefert.

Die Rechenregeln für die Verknüpfung einer Variablen mit einer Konstanten oder einer Variablen mit sich selbst oder ihrer Negation nennt man Theoreme.

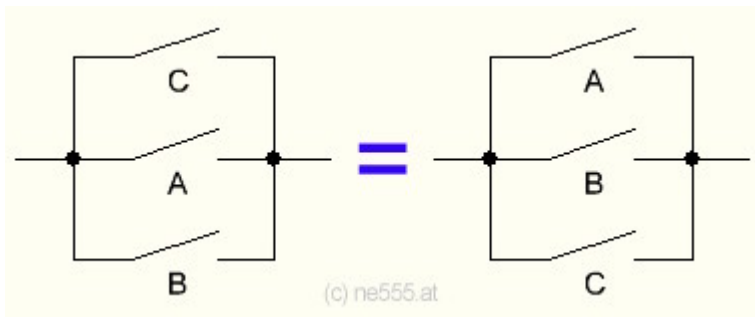
Kommutativgesetz und Assoziativgesetz

KOMMUTATIVGESETZ:

Kommutativ bedeutet Vertauschung. Somit zeigt uns das Kommutativgesetz die Vertauschbarkeit der Variablen bei der UND- und der ODER-Verknüpfung.



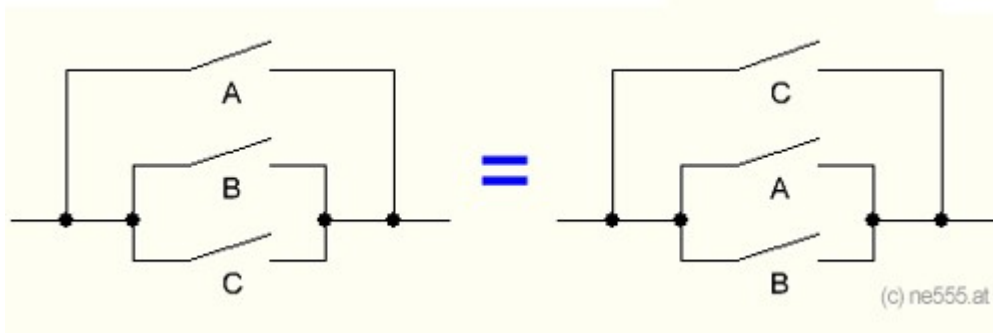
$$Z = A \text{ und } B \text{ und } C = C \text{ und } A \text{ und } B$$



$$Z = C \text{ oder } A \text{ oder } B = A \text{ oder } B \text{ oder } C$$

Die Reihenfolge der Variablen einer UND- bzw. einer ODER-Verknüpfung ist beliebig und hat keinen Einfluß auf das Ergebnis.

ASSOZIATIVGESETZ:



$$Z = A \text{ oder } (B \text{ oder } C) = C \text{ oder } (A \text{ oder } B)$$

Die Reihenfolge der Zuordnung der Variablen bei der UND- bzw. ODER-Verknüpfung hat keinen Einfluß auf das Ergebnis.

Distributivgesetz

Das Distributivgesetz entspricht dem Ausklammern bzw. Herausheben eines Faktors in der normalen Algebra.

Man unterscheidet das **konjunktive** und das **disjunktive** Distributivgesetz.

Das konjunktive Distributivgesetz lautet:

$$Z = A * (B + C) = (A * B) + (A * C)$$

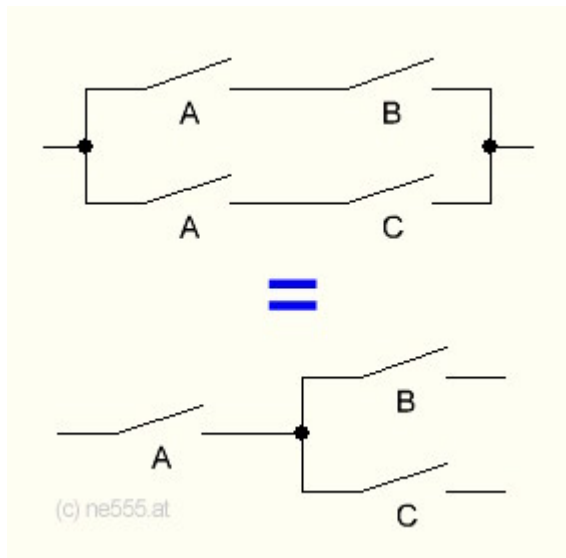
Wir verwenden, obwohl veraltet, die einfachere Schreibweise:

* entspricht dem UND
+ entspricht dem ODER

Das disjunktive Distributivgesetz lautet:

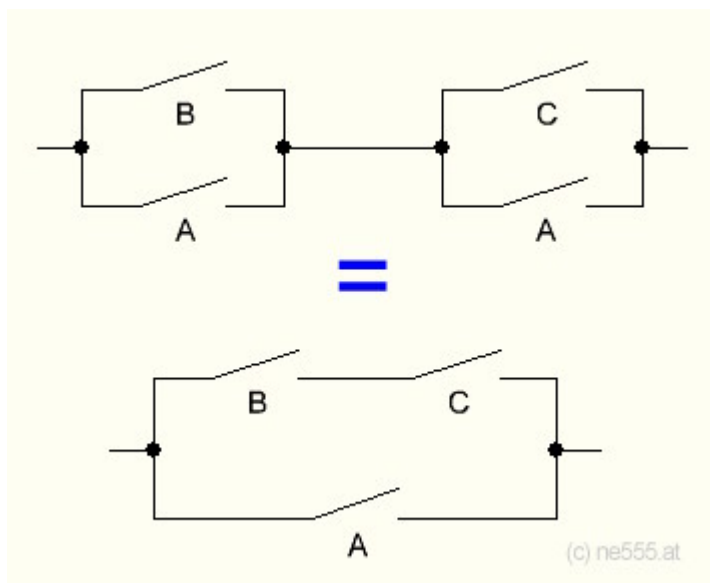
$$Z = A + (B * C) = (A + B) * (A + C)$$

Das ganze mit Schaltern dargestellt sieht dann so aus:



$$Z = (A * B) + (A * C) = A * (B + C)$$

bzw:



$$Z = (A + B) * (A + C) = A + (B * C)$$

Der englische Mathematiker Augustus DeMorgan hat diese nach ihm benannten Gesetze gefunden.

Die Gesetze werden verwendet, um Ausdrücke zu vereinfachen bzw. zur Umrechnung von NAND auf NOR oder NOR auf NAND.

1. DeMorgansches Gesetz:

Das 1. DeMorgansche Gesetz beschreibt die Umwandlung von NAND auf NOR.

$$Z = \overline{A \wedge B} = \bar{A} \vee \bar{B}$$

Eine UND-Verknüpfung kann durch Auftrennen des Negationsstriches in eine ODER-Verknüpfung umgewandelt werden

2. DeMorgansches Gesetz:

Das 2. DeMorgansche Gesetz beschreibt die Umwandlung von NOR auf NAND.

$$Z = \overline{A \vee B} = \bar{A} \wedge \bar{B}$$

Eine ODER-Verknüpfung kann durch Auftrennen des Negationsstriches in eine UND-Verknüpfung umgewandelt werden

Beide Gesetze gelten nicht nur für zwei, sondern für beliebig viele Variablen.

Eine Verknüpfung mehrerer Variablen kann leicht zu Mehrdeutigkeiten führen.

Die Gleichung:

$$Z = A \wedge B \vee C$$

kann auf zwei Arten interpretiert werden:

a.)

$$Z1 = (A \wedge B) \vee C$$

b.)

$$Z2 = A \wedge (B \vee C)$$

Die Ergebnisse Z1 und Z2 sind vollkommen verschieden. Um solche Mehrdeutigkeiten zu vermeiden, wurde eine Bindungsregel eingeführt.

Wie die "Punkt vor Strich-Rechnung" der Algebra gilt hier:

UND vor ODER

Eine UND-Verknüpfung bindet stärker als eine ODER-Verknüpfung.

Befindet sich also eine UND- und eine ODER-Verknüpfung ohne Klammern in einer Gleichung, so gilt die UND-Verknüpfung als geklammert. Sollten zwei oder mehr Variablen mit einem Negationsstrich verbunden sein, gilt dies auch als Klammerung.

Um solche Mehrdeutigkeiten zu vermeiden, schreibt man auch:

$$Z = AB \vee C$$

Die UND-Verknüpfung zwischen A und B wurde weggelassen. Wenn zwischen zwei Variablen keine Verknüpfung steht, ist es immer eine UND-Verknüpfung.

Laut DeMorgan kann jede UND-Verknüpfung mittels ODER- und NICHT-Gattern gebildet werden.

Beispiel:

1. DeMorgansches Gesetz

$$Z = \overline{A \wedge B} = \bar{A} \vee \bar{B}$$

beides negieren

$$Z = \overline{\overline{A \wedge B}} = \overline{\bar{A} \vee \bar{B}}$$

da sich eine doppelte Negation aufhebt ergibt sich ein NOR mit negierten Eingängen

$$Z = A \wedge B = \overline{\bar{A} \vee \bar{B}}$$

Daraus folgt, dass ein UND-Gatter nicht erforderlich ist.

Jede Verknüpfungsschaltung kann somit aus ODER- und NICHT-Gattern aufgebaut werden.

Da sogar NICHT-Gatter aus NOR-Gattern bestehen können, nennt man das NOR-Gatter auch Universalgatter mit dem jede gewünschte Verknüpfungsschaltung aufgebaut werden kann.

Aus dem 2. DeMorganschen Gesetz lässt sich ein ODER auch mittels UND und NICHT bilden.

Beispiel:

2. DeMorgansches Gesetz

$$Z = \overline{A \vee B} = \bar{A} \wedge \bar{B}$$

beides negieren

$$Z = \overline{\overline{A \vee B}} = \overline{\bar{A} \wedge \bar{B}}$$

da sich eine doppelte Negation aufhebt ergibt sich ein NAND mit negierten Eingängen

$$Z = A \vee B = \overline{\bar{A} \wedge \bar{B}}$$

Daraus folgt, dass ein ODER-Gatter nicht erforderlich ist.

Jede Verknüpfungsschaltung kann somit aus NAND-Gattern aufgebaut werden.

Daher bezeichnet man das NAND-Gatter wie auch das NOR-Gatter als Universalgatter mit dem jede gewünschte Verknüpfungsschaltung aufgebaut werden kann.

Aufgabe:


Die folgende Schaltung soll nur aus NAND-Gattern realisiert werden.

Die Schaltungsgleichung lautet:

$$Z = (\bar{A} \wedge \bar{B} \wedge C) \vee (A \wedge \bar{B} \wedge \bar{C})$$

der 1. Schritt ist nun, die ganze rechte Seite der Gleichung doppelt zu negieren.

2. der innere Negationsstrich wird aufgetrennt und aus der ODER-Verknüpfung wird eine UND-Verknüpfung.

$$Z = \overline{\overline{(\bar{A} \wedge \bar{B} \wedge C) \vee (A \wedge \bar{B} \wedge \bar{C})}}$$

$$Z = \overline{\overline{(\bar{A} \wedge \bar{B} \wedge C)} \wedge \overline{\overline{(A \wedge \bar{B} \wedge \bar{C})}}}$$

Voilà, wir haben unsere Gleichung aus lauter NAND-Gatter realisiert.

Ein weiteres Beispiel:

Die folgende Gleichung soll vereinfacht werden.

$$Z = B \vee (\bar{A} \wedge B \wedge C) \vee \bar{B}$$

Hier fällt auf, dass wir am Anfang und am Ende der Gleichung eine ODER-Verknüpfung mit B und B-Nicht haben.

$$Z = \underbrace{B \vee \bar{B}}_1 \vee (\bar{A} \wedge B \wedge C)$$

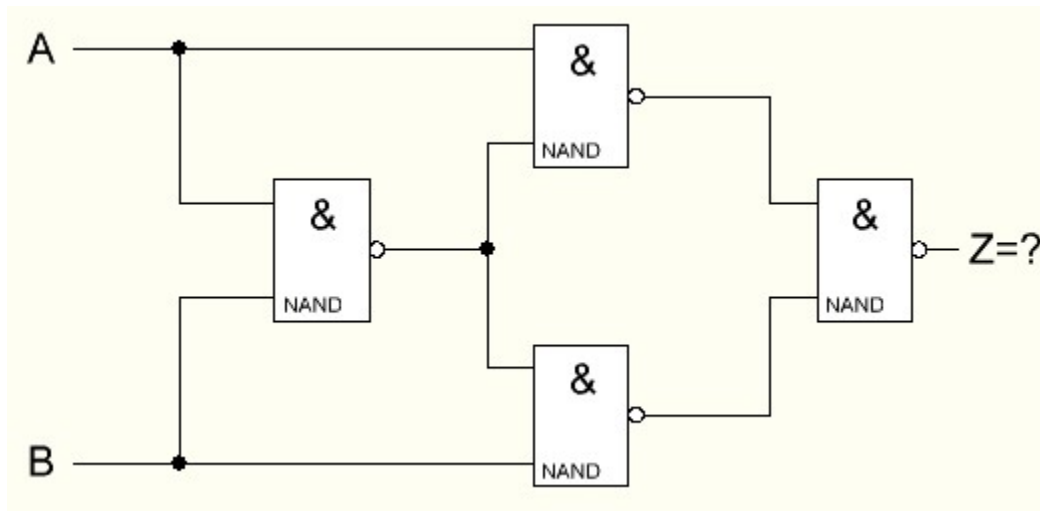
Das ergibt immer 1. (Vergleichbar mit einer Parallelschaltung von zwei Schaltern B. Ein Schalter ist immer geschlossen und einer offen.)

$$Z = 1 \vee (\bar{A} \wedge B \wedge C)$$

Das ergibt auch 1. Egal wie A, B oder C steht, durch die ODER-Verknüpfung mit der Konstanten 1 ist das Ergebnis immer 1.

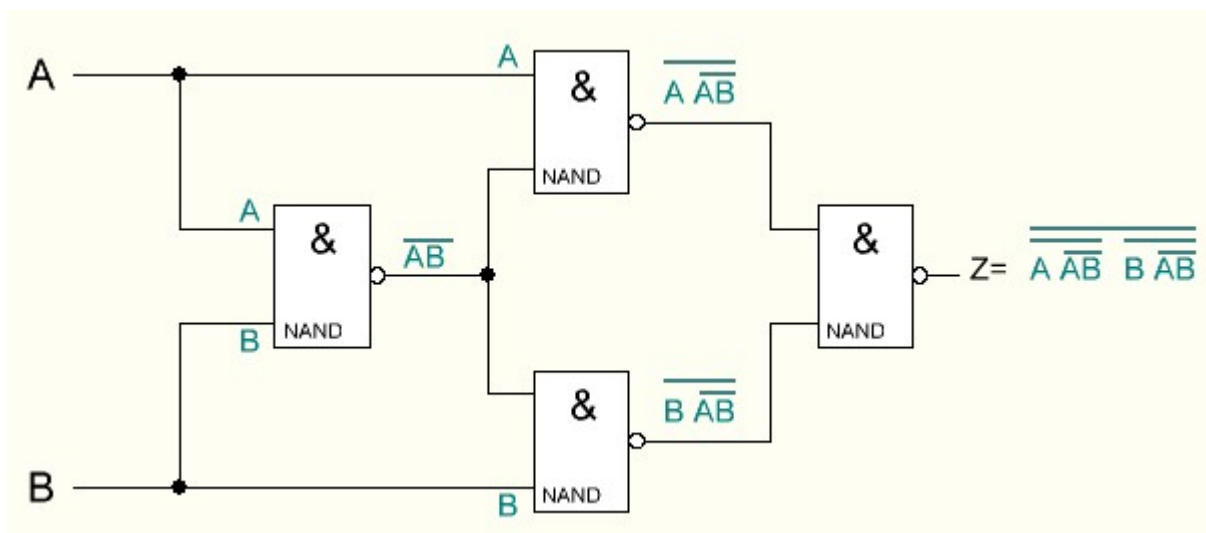
$$Z = 1$$

Anhand der gegebenen Schaltung soll die Funktionsgleichung erstellt werden.



Dies ist am einfachsten, wenn man die jeweiligen Ausgangsgleichungen mit dem nächsten Eingang verbindet. So kommt man schrittweise auf das Endergebnis.

Die UND-Verknüpfung wurde hier bereits weggelassen. Also $AB = A$ und B usw.



Diese Gleichung kann man aber noch weiter vereinfachen. Wir wenden das 2. DeMorgansche Gesetz an.

Die doppelte Negierung hebt sich auf und wir trennen den Negationsstrich auf.

$$Z = \overline{\overline{A \wedge \overline{A} \wedge B}} \wedge \overline{\overline{B \wedge \overline{A} \wedge B}}$$

$$Z = \overline{\overline{A \wedge \overline{A} \wedge B}} \vee \overline{\overline{B \wedge \overline{A} \wedge B}}$$

Aus unserem UND wird ein ODER. Die beiden doppelten Negationen heben sich auf und wir erhalten

$$Z = (A \wedge \overline{A} \wedge \overline{B}) \vee (B \wedge \overline{A} \wedge \overline{B})$$

$$Z = (A \wedge \overline{A} \vee \overline{B}) \vee (B \wedge \overline{A} \vee \overline{B})$$

Wir trennen weiter auf und heben die gemeinsamen Faktoren heraus. Es entsteht

$$Z = \underbrace{A \overline{A} \vee A \overline{B}}_0 \vee \underbrace{B \overline{A} \vee B \overline{B}}_0$$

somit bleibt

$$Z = 0 \vee A \overline{B} \vee B \overline{A} \vee 0$$

die 0 noch weg und es bleibt

$$Z = A \overline{B} \vee B \overline{A}$$

Das ist ein XOR

6. Schaltalgebra - Schaltungsbeispiel

Zwei Zahlen (A und B) sollen miteinander verglichen werden.

Gesucht:

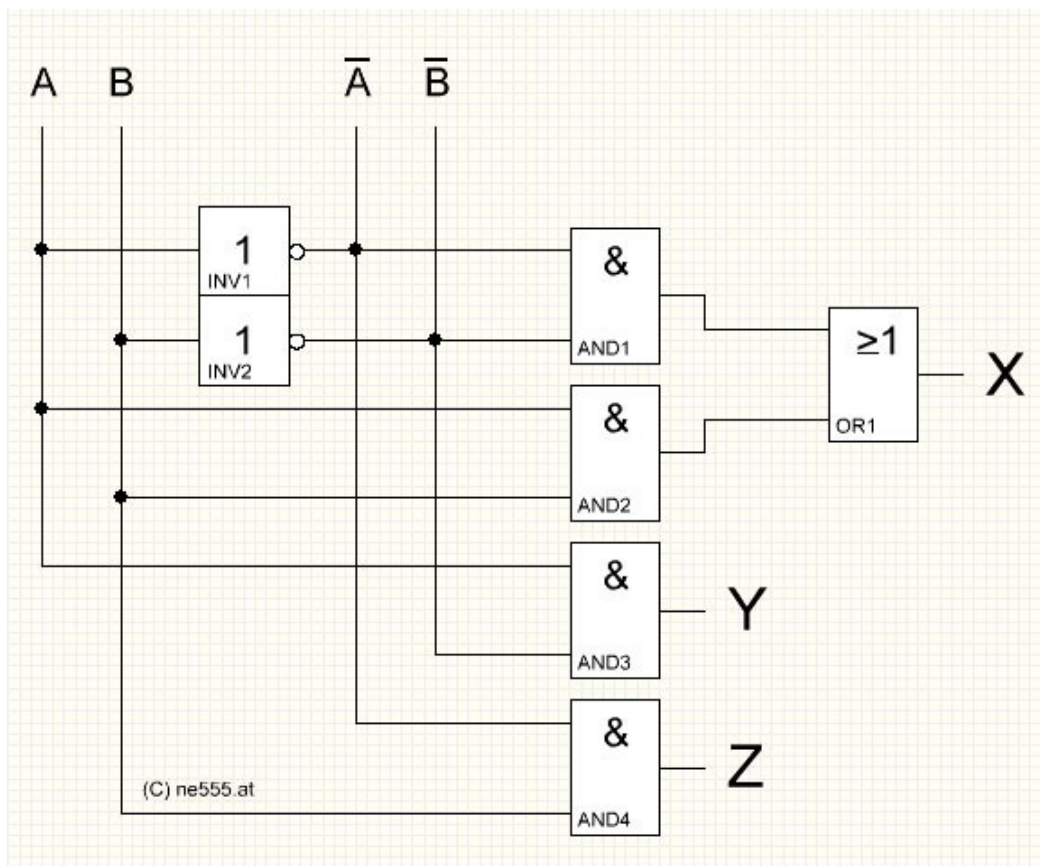
- 1.) Wahrheitstabelle und Gleichungen
- 2.) Schaltung

1.) Wir erstellen unsere Wahrheitstabelle: * = ODER, ' steht für NICHT

A	B	X	Y	Z
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

$X = (A'B') * (AB)$
 $Z = (A'B)$
 $Y = (AB')$

2.) Schaltung



7. KV-Diagramme

Das **Karnaugh-Veitch-Diagramm**, kurz **KV-Diagramm**, dient der übersichtlichen Darstellung und Vereinfachung Boolescher Funktionen – Umwandlung der disjunktiven Normalform in einen minimalen logischen Ausdruck.

Mittels eines KV-Diagramms lässt sich jede beliebige disjunktive Normalform (DNF) in einen minimalen logischen Ausdruck umwandeln. Der Vorteil gegenüber anderen Verfahren ist, dass der erzeugte Term (meist) minimal ist. Sollte der Term noch nicht minimal sein, ist eine weitere Vereinfachung durch Anwenden des Distributivgesetzes (Ausklammern) möglich. Das Umwandeln beginnt mit dem Erstellen einer Wahrheitstafel, aus der dann die DNF abgeleitet wird, die dann wiederum direkt in ein KV-Diagramm umgewandelt wird.

Da sich benachbarte Felder jeweils in einer Variable nur um ein Bit unterscheiden, ist folgende Regel anwendbar: A oder $\neg A = 1$. Auf dieser Regel basiert die Reduzierung der Gruppen.

Hier ein Beispiel:

Zuerst wird die Wahrheitstabelle einer Schaltung und daraus die Gleichung erstellt. Da wir hier nur zwei Eingänge haben, ergibt sich die Größe des KV-Diagramms mit 2^n also $2^2 = 4$ Felder. (n steht für die Anzahl der Eingangsvariablen)

Nun werden die Werte entsprechend ihrer Bedingungen in die Felder (dort wo sie sich überschneiden) eingetragen.

Nach der Minterm-Methode werden die Zustände für 1 ($Z=1$) und nach der Maxterm-Methode die Zustände für 0 ($Z=0$) eingetragen.

Nun werden alle 1 oder alle 0 zusammengefasst.

Wir wenden die Minterm-Methode an und fassen die 1 zusammen. Es können immer nur 2, 4, 8 usw. benachbarte Felder horizontal oder vertikal zusammengefasst werden.

	B	A	Z	
0	0	0	1	$\longrightarrow Z = \bar{A} \wedge \bar{B} = \overline{A \wedge B}$
1	0	1	1	$\longrightarrow Z = A \wedge \bar{B}$
2	1	0	1	$\longrightarrow Z = \bar{A} \wedge B$
3	1	1	0	

Die Gleichung lautet:

$$Z = \bar{A} \wedge \bar{B} \vee Z = A \wedge \bar{B} \vee Z = \bar{A} \wedge B$$

	\bar{A}	A
\bar{B}	1	1
B	1	0

	\bar{A}	A	
\bar{B}	1	1	$\longrightarrow \bar{B}$
B	1	0	$\longrightarrow \bar{A}$

$\longrightarrow Z = \bar{A} \vee \bar{B}$

	\bar{A}	A	
\bar{B}	1	1	
B	1	0	$\longrightarrow \bar{Z} = A \wedge B \longrightarrow Z = \overline{A \wedge B}$

Nach der Maxterm-Methode:

Da wir 3 Eingänge haben, vergrößert sich unser KV-Diagramm auf 2^3 also 8 Felder.

Wir erstellen auch hier wieder die Wahrheitstabelle und übertragen die Gleichungen in das KV-Diagramm.

Durch Zusammenfassen der 2-er Kombinationen erhalten wir unsere Gleichung.

	C	B	A	Z	
0	0	0	0	0	
1	0	0	1	0	
2	0	1	0	0	
3	0	1	1	1	$\longrightarrow Z = A \wedge B \wedge \bar{C}$
4	1	0	0	0	
5	1	0	1	1	$\longrightarrow Z = A \wedge \bar{B} \wedge C$
6	1	1	0	1	$\longrightarrow Z = \bar{A} \wedge B \wedge C$
7	1	1	1	1	$\longrightarrow Z = A \wedge B \wedge C$

Die Funktionsgleichung lautet:

$$Z = A \wedge B \wedge \bar{C} \vee Z = A \wedge \bar{B} \wedge C \vee Z = \bar{A} \wedge B \wedge C \vee Z = A \wedge B \wedge C$$

	\bar{A}	A	A	\bar{A}	
\bar{B}	0	0	1	0	$A \wedge B$
B	0	1	1	1	$A \wedge C$
	\bar{C}	\bar{C}	C	C	$B \wedge C$

$$Z = A \wedge B \vee A \wedge C \vee B \wedge C$$

Da wir 4 Eingänge haben, vergrößert sich unser KV-Diagramm auf 2^4 also 16 Felder.

Wir erstellen auch hier wieder die Wahrheitstabelle und übertragen die Werte in das KV-Diagramm.

Da wir einen 6er-Block nicht zusammenfassen können, teilen wir ihn in zwei 4er-Blöcke auf.

	D	C	B	A	Z
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	1
10	1	0	1	0	1
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	0
14	1	1	1	0	0
15	1	1	1	1	1

	\bar{A}	A	A	\bar{A}	
\bar{B}	1	1	0	0	\bar{D}
B	1	1	1	0	\bar{D}
B	1	1	1	0	D
\bar{B}	1	1	0	0	D
	\bar{C}	\bar{C}	C	C	

Als Ergebnis für $Z=1$ erhalten wir die Gleichung

$$\bar{C} \vee (B \wedge \bar{C}) \vee (B \wedge A)$$

Möglich sind auch folgende Zusammenfassungen:

Auch Eckfelder sind benachbarte Felder und können zusammengefasst werden.

	\bar{A}	A	A	\bar{A}	
\bar{B}	0	1	5	4	\bar{D}
B	2	3	7	6	\bar{D}
B	10	11	15	14	D
\bar{B}	8	9	13	12	D
	\bar{C}	\bar{C}	C	C	

Das Ergebnis ist:

$$\boxed{B \wedge \bar{C}} \vee \boxed{B \wedge A} \vee \boxed{\bar{B} \wedge D} \vee \boxed{A \wedge C} \vee \boxed{\bar{B} \wedge C}$$

oder auch:

	\overline{A}	A	A	\overline{A}	
\overline{B}	1	1	0	0	\overline{D}
	0	1	5	4	
B	1	1	1	1	\overline{D}
	2	3	7	6	
B	1	1	1	1	D
	10	11	15	14	
\overline{B}	1	1	0	0	D
	8	9	13	12	
	\overline{C}	\overline{C}	C	C	

Das Ergebnis ist:

$$\overline{C} \vee (B \wedge \overline{A}) \vee (B \wedge A)$$

8. Code

Was ist ein Code?

Ein Code ist eine eindeutige Zuordnung zwischen zwei Mengen. Der BCD-Code ist z.B. eine Zuordnung zwischen Dezimal- und Binärzahlen. Jeder Dezimalzahl entspricht dabei eine Binärzahl.

Z_{in} ----->  -----> Z_{out}

Es gibt viele verschiedene Codes, die für bestimmte Anwendungen mehr oder weniger gut geeignet sind. Beispielsweise EAN-Code auf jeder Verpackung oder der amerikanische Standard Code ASCII.

Zunächst wollen wir uns mit einem für den Elektroniker sehr wichtigen Code - dem **BCD-Code** beschäftigen.

BCD-CODE (**B**inary **C**oded **D**ecimals)

Beim BCD-Code werden die ersten 10 Kombinationen (also von 0 bis 9) aus dem Dualcode übernommen. Die Kombinationen A...F sind die sogenannten Pseudotetraden (unechte Tetraden).

Jede Dezimalziffer wird durch 4 binäre Stellen (Tetraden) dargestellt.

MSB...Most Sygnificant Bit

LSB...Least Sygnificant Bit

	MSB		LSB	
Dez	D	C	B	A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

„echte Tetraden“

„unechte Tetraden“

Bsp: $239_{\text{dez}} \rightarrow \text{BCD}$

2 = 0010

3 = 0011

9 = 1001

$239_{\text{dez}} = 0010\ 0011\ 1001$ im BCD-Code

Addition BCD-Code

3+6

3 -> 0011

6 -> 0110

9 -> 1001 (0+1 = 1 1+1=0 +1 Übertrag)

9+3

9 -> 1001

3 -> 0011

12 -> 1100 = Pseudotetrade

+ 6 addieren (da 6 Pseudotetraden)

12 1100

+6 0110

0001 0010

0001 = 1, 0010 = 2 = 12

78+69

78 -> 0111 1000

69 -> 0110 1001

1110 0001

+6 0110 0110 (+6, da mit 1110 eine Pseudotetrade)

0001 0100 0111

1 4 7

Quelle:

<http://www.ne555.at/elektronik-grundlagen/digitaltechnik.html>