



EUROPA-FACHBUCHREIHE
für informationstechnische und
kommunikationstechnische Berufe

IKT-Fachkunde

Bearbeitet von Lehrern und Ingenieuren an beruflichen Schulen

6. Auflage

Informationsverarbeitung

(Auszug)

VERLAG EUROPA-LEHRMITTEL · Nourney, Vollmer GmbH & Co. KG
Düsselberger Straße 23 · 42781 Haan-Gruiten

Europa-Nr.: 36519

6 Informationsverarbeitung und Elektrotechnik

6.1 Informationsverarbeitung in IT-Systemen	187
6.1.1 Bedeutung und Darstellung der Information.....	187
6.1.2 Zahlensysteme.....	188
6.1.3 Binärcodes.....	189
6.1.4 Logische Funktionen.....	191
6.1.5 Boole'sche Algebra	196
6.1.6 Entwicklung logischer Schaltungen.....	198
6.1.7 Digitalschaltungen mit speicherndem Verhalten.....	202
6.1.8 Tristate-Schaltelemente.....	206
6.1.9 Multiplexer, Demultiplexer	207

6 Informationsverarbeitung und Elektrotechnik

6.1 Informationsverarbeitung in IT-Systemen

6.1.1 Bedeutung und Darstellung der Information

Die Verarbeitung und die Übertragung von elektrischen Signalen erfolgt in zunehmendem Maße in digitaler Form. Hierfür werden analoge Größen, z. B. Musik, Sprache, Bilder oder gemessene physikalische Größen wie die Temperatur, digitalisiert. Dadurch können z. B. Bilder oder Musikstücke leichter nachbearbeitet werden, Messwerte schnell und grafisch übersichtlich ausgewertet, Abläufe in der Technik oder der Natur simuliert und alle Informationen auf kleinstem Raum gespeichert werden.

Ein einzelner analoger Musikton ist eine zeitkontinuierliche Sinusspannung (**Bild 1**). Zur digitalen Übertragung wird diese zyklisch (in gleichen Zeitabständen) abgetastet und in Binärsignale (lat. bini = je zwei) umgewandelt. Eine einzelne Stelle eines binären Signals nennt man Binärzeichen (von binary digit) oder einfach Bit, Mehrzahl Bits. Diese besitzen die Einheit 1 bit oder 1 Sh (Shannon).

Ein Bit hat den Signalwert 0 oder 1.

Besitzt ein Binärsignal 8 Bits, hat es die Einheit 8 bit oder 1 B = 1 Byte (**Tabelle 1**).

Die Speicherkapazitäten von Speicherbauelementen, z.B. RAM, Flash-EEPROM, werden in Kibit = Kilobit oder in KiB = Kilobyte angegeben. Die Abkürzung Kilo beträgt dabei nicht $10^3 = 1000$ wie in der Physik, sondern $2^{10} = 1024$. Zur Unterscheidung wird das Kilo nicht mit dem Kleinbuchstaben k sondern mit Ki abgekürzt.

Massenspeicher, z. B. Festplatten, haben Speicherkapazitäten von GiB = Gigabyte oder TiB = TeraByte. 1 Gi (giga) hat hier den Wert $1\text{Ki} \cdot 1\text{Ki} \cdot 1\text{Ki} = 2^{10} \cdot 2^{10} \cdot 2^{10} = 2^{30}$.

Beispiel 1: Speicherkapazität berechnen

Wie viel bit Speicherkapazität hat der Arbeitsspeicher eines PC mit 4 GiB?

Lösung:

$$4\text{GiB} = 4 \cdot 2^{30} \cdot 8\text{bit} \approx 34,36\text{ Milliarden bit}$$

Den Signalwerten 0 und 1 sind Spannungen zugeordnet. Der Wert 0 erhält z.B. die Spannung 0 V und wird mit L (von low = niedrig) bezeichnet (**Tabelle 2**). Der Wert 1 oder H (von high = hoch) kann je nach Gerätetechnologie unterschiedliche Spannungen führen, z.B. 3 V.

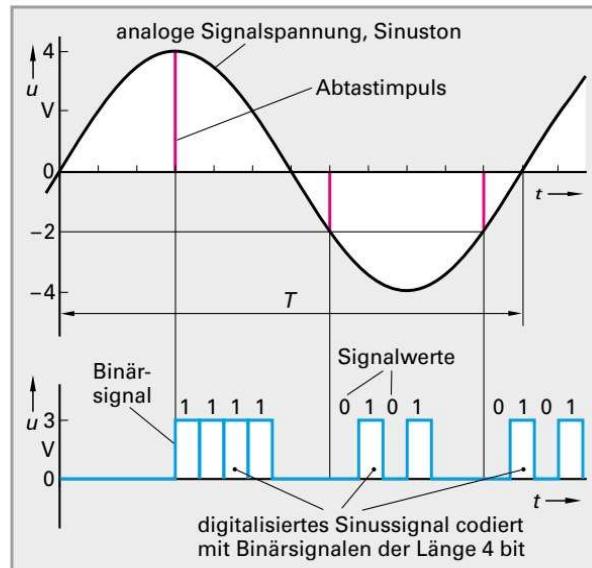


Bild 1: Digitalisierung eines Musiktons

Tabelle 1: Einheiten von binären Signalen

Einheit	IEC-Norm	Umformungen
1 bit	1 bit	$2^0 \text{ bit} = 1 \text{ bit}$
1 Kbit	1 Kibit	$2^{10} \text{ bit} = 1024 \text{ bit}$
1 Mbit	1 Mibit	$1\text{Ki} \cdot 1\text{Ki} \text{ bit} = 2^{20} \text{ bit} = 1048576 \text{ bit}$
1 Gbit	1 Gibit	$1\text{Ki} \cdot 1\text{Mi} \text{ bit} = 2^{30} \text{ bit} = 1073741824 \text{ bit}$
1 kbit		$10^3 \text{ bit} = 1000 \text{ bit}$
1 B	1 B	$2^3 \text{ bit} = 8 \text{ bit}$
1 KB	1 KiB	$2^{10} \text{ B} = 1024 \text{ B} = 1024 \cdot 2^3 \text{ bit} = 1024 \cdot 8 \text{ bit} = 8192 \text{ bit}$
1 MB	1 MiB	$1\text{Ki} \cdot 1\text{KiB} = 2^{20} \text{ B} = 1048576 \text{ B}$
1 GB	1 GiB	$2^{20} \cdot 2^3 \text{ bit} = 2^{23} \text{ bit} = 8388608 \text{ bit}$
1 TB	1 TiB	$1\text{Ki} \cdot 1\text{MiB} = 2^{30} \text{ B} = 1073741824 \text{ B}$
1 PB	1 PiB	$2^{30} \cdot 2^3 \text{ bit} = 2^{33} \text{ bit} = 8589934592 \text{ bit}$
1 EB	1 EiB	$1\text{Ki} \cdot 1\text{GiB} = 2^{40} \text{ B} = 1099511627776 \text{ B}$
$2^{40} \cdot 2^3 \text{ bit} = 2^{43} \text{ bit} = 8796093022208 \text{ bit} = 8,8 \text{ Billionen bit}$		
$1\text{Ki} \cdot 1\text{TiB} = 2^{50} \text{ B} = 1,13 \text{ Billiarden B}$		
$2^{50} \cdot 2^3 \text{ bit} = 2^{53} \text{ bit} = 9 \text{ Billiarden bit}$		
$1\text{Ki} \cdot 1\text{PiB} = 2^{60} \text{ B} = 1,15 \text{ Trillionen B}$		
$2^{60} \cdot 2^3 \text{ bit} = 2^{63} \text{ bit} = 9,22 \text{ Trillionen bit}$		
1 Bit $\leq 1 \text{ Sh}$, Sh von Shannon		

Tabelle 2: Spannungspegel bei Binärsignalen

Anwendung	Binärwert	Bezeichnung	Spannung
Prozessoren, logische Bauelemente	0 1	low high	0V ... 0,8V 1,2V ... 5V
Datenleiter der V.24-Schnittstelle	0 1	low high	3V ... 15V -3V ... -15V
Steuerleiter der V.24 Schnittstelle	0 1	low high	-3V ... -15V 3V ... 15V
Speicherprogrammierbare Steuerungen	0 1	low high	0 V 24 V

6.1.2 Zahlensysteme

Aus Speicherbauelementen werden mittels binär codierter Adressen Speicherinhalte in Form von Datenworten ein- oder ausgelesen (**Bild 1**). Dazu sind den Adressleitern, z. B. A0 bis A7, des Adressbusses Stellenwertigkeiten des dualen Zahlensystems zugeordnet. Der niederwertigste Adresseingang A0 erhält die Stellenwertigkeit $2^0 = 1$, A1 erhält $2^1 = 2$, usw. Die Stellenwertigkeit verdoppelt sich somit mit jedem folgenden Adresseingang (**Tabelle 1**). Der höchste Adresseingang, hier A7, erhält die Wertigkeit $2^7 = 128$.

Beispiel 1: Dual-Dezimal-Umwandlung

Welcher Dezimalzahl entspricht die in Bild 1 eingestellte Speicheradresse?

Lösung:

$$\begin{array}{ccccccccc}
 & A_7 & A_6 & A_5 & A_4 & A_3 & A_2 & A_1 & A_0 \\
 \text{Stellen-} & \left\{ \begin{array}{ccccccccc} 2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\ 128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \end{array} \right. \\
 \text{wert} \\
 \text{Adresse} & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\
 \\
 = & 128 + 0 + 0 + 0 + 0 + 8 + 4 + 0 + 1 = 141
 \end{array}$$

Da die Adressen von größeren Speichern aus mehr als 8 Binärzeichen, also 8 bit, bestehen, entstehen bei der Darstellung im Dualsystem sehr lange Zahlen.

Wegen der besseren Lesbarkeit werden diese, z. B. bei der Erstellung von Programmen, hexadezimal codiert (Tabelle 1). Die Hexadezimalziffern können 16 unterschiedliche Werte annehmen. Deshalb werden Werte größer als 9 mit den ersten 6 Großbuchstaben des Alphabets dargestellt. Da der Übertrag der Hexadezimalzahlen in die zweite Stelle des Hexadezimalsystems gleichzeitig mit dem Übertrag der Dualzahlen in die fünfte Stelle des Dualsystems auftritt (Tabelle 1), können immer 4 Binärstellen in eine Hexadezimalziffer beginnend mit den 4 niedrigwertigsten Bits umgewandelt werden.

Beispiel 2: Dual-Hexadezimal-Umwandlung

Geben Sie die Speicheradresse aus Bild 1 in hexadezimaler Form an.

Lösung:

$$\begin{array}{r} 1000 \\ \times 8 \\ \hline 1101 \end{array} = 8D$$

Die Anzahl aller wählbaren Adressen bestimmt die Kapazität eines Speichers. Hinter jeder Adresse ist ein Datenwort einer bestimmten Größe, z. B. 8 bit (Bild 1), abgelegt. Beträgt die Anzahl der Adressleiter eines Speichers n , so beträgt die Gesamtzahl aller wählbaren Adressen 2^n .

! Anschlussbezeichnungen werden mit Großbuchstaben aber deren Signale mit Kleinbuchstaben gekennzeichnet.

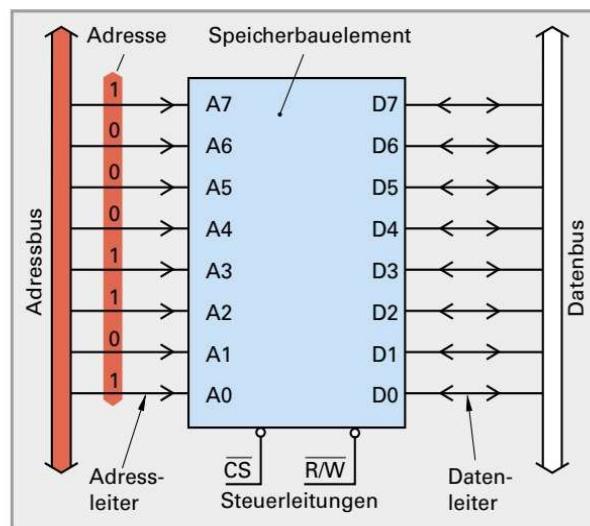


Bild 1: Signalbelegung eines Speicherbauelementes

Tabelle 1: Zahlensysteme

System	dezimal			dual						hexadezimal		
Wertigkeit	10 ²	10 ¹	10 ⁰	2 ⁶	2 ⁵	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰	16 ¹	16 ⁰
Zahl im jeweiligen Zahlen-system	0								0		0	
	1								1		1	
	2							1	0		2	
	3							1	1		3	
	4						1	0	0		4	
	5						1	0	1		5	
	6						1	1	0		6	
	7						1	1	1		7	
	8					1	0	0	0		8	
	9					1	0	0	1		9	
	1 0					1	0	1	0		A	
	1 1					1	0	1	1		B	
	1 2					1	1	0	0		C	
	1 3					1	1	0	1		D	
	1 4					1	1	1	0		E	
	1 5					1	1	1	1		F	
	1 6				1	0	0	0	0	1	0	
	1 7				1	0	0	0	1	1	1	
	1 8				1	0	0	1	0	1	2	
	1 9				1	0	0	1	1	1	3	
	2 0				1	0	1	0	0	1	4	
	3 1				1	1	1	1	1	1	F	
	3 2			1	0	0	0	0	0	2	0	
	3 3			1	0	0	0	0	1	2	1	
	4 8			1	1	0	0	0	0	3	0	
	6 4		1	0	0	0	0	0	0	4	0	
	1 2 6		1	1	1	1	1	1	0	7	E	
	1 2 7		1	1	1	1	1	1	1	7	F	

Beispiel 1: Speicherkapazität berechnen

Berechnen Sie die Speicherkapazität eines Speichers mit 8 Adressleitern und 8 Datenleitern.

Lösung:

$$2^8 \cdot 8 \text{ bit} = 2^8 \text{ B} = 256 \text{ Byte}$$

In Rechenwerken werden binäre Ziffern verarbeitet, z. B. addiert. Bei der Addition zweier Einsen erhält man einen Übertrag in die nächst höhere Stelle, da Ziffern größer 1 nicht möglich sind (**Bild 1**).

Beispiel 2: Dualzahlen addieren

Addieren Sie die Dualzahlen 1001 und 1011.

Lösung:

$$\begin{array}{r} 1001 \\ + 1011 \\ \hline \text{Übertrag} & 111 \\ & \hline 10100 & \triangleq 20 \end{array}$$

6.1.3 Binärcodes

Zum Rechnen in Rechenwerken, bei der Datenübertragung und beim Erfassen von Messwerten werden Dezimalzahlen nicht als mehrstellige Dualzahl eingesetzt, sondern jede einzelne Ziffer wird binär verschlüsselt.

BCD-Codes

BCD-Codes (von binary coded decimal = binär codiertes Zehnersystem) besitzen mindestens 4 bit zur Verschlüsselung von Dezimalziffern (**Tabelle 1**). Ein 4-Bit-Wort nennt man auch Tetrade (von griech. tetra = vier). Da man zur Verschlüsselung von 10 Ziffern aber nur zehn Tetraden benötigt, unterscheidet man echte Tetraden von Pseudotetraden. Letztere werden für die Codierung nicht

Hamming-Code. Ist ein fehlerkorrigierender Code zum gesicherten Übertragen oder Speichern von Daten.

$$\begin{array}{r} 0 \\ + 0 \\ \hline 0 \end{array} \quad \begin{array}{r} 1 \\ + 0 \\ \hline 1 \end{array} \quad \begin{array}{r} 0 \\ + 1 \\ \hline 1 \end{array} \quad \begin{array}{r} 1 \\ + 1 \\ \hline 10 \end{array}$$

Bild 1: Addieren von Dualziffern

verwendet. Wird eine dreistellige Dezimalzahl mit Tetraden codiert, ist sie 12 bit groß.

Oft werden Dezimalziffern mit Binärworten von mehr als 4 bit codiert, z. B. im Biquinär-Code oder im 2-aus-5-Code (Tabelle 1). Solche Codes nennt man redundant (weitschweifig, lat. redundans = Überfluss habend). Redundante Codes sind leichter auf Fehler überprüfbar. Bei der Datenübertragung prüft der Empfänger z. B. beim 1-aus-10 Code (Tabelle 1), ob das empfangene Binärwort aus einem Wert 1 und neun Werten 0 besteht. Ist dies nicht der Fall, wird der Datensender vom Empfänger aufgefordert, das Binärwort zu wiederholen. Der Empfänger kann somit den Fehler in einer Binärstelle sicher erkennen.

Viele BCD-Codes besitzen eine Stellenwertigkeit. Der 2-aus-5-Code hat für die Dezimalziffern 1 bis 9 die Stellenbewertung 7-4-2-1-0, für die Dezimalziffer 0 hat er jedoch keine Stellenbewertung.

Ein weiterer redundanter Code ist der Hamming¹-Code (Tabelle 1). Bei ihm können bis zu zwei Bitfehler in einem Codewort erkannt werden, da sich zwei beliebige Codewörter in mindestens drei Binärstellen unterscheiden. Der dargestellte Hamming-Code hat somit die Hammingdistanz 3.

Tabelle 1: Beispiele von BCD-Codes

Dezimalziffer	8-4-2-1-Code	Aiken ² -Code	Biquinär-Code	2-aus-5-Code	1-aus-10-Code	Hamming-Code
0	0 0 0 0	0 0 0 0	0 0 0 0 1 0 1	1 1 0 0 0	0 0 0 0 0 0 0 0 1	0 0 0 0 0 0 0
1	0 0 0 1	0 0 0 1	0 0 0 1 0 0 1	0 0 0 1 1	0 0 0 0 0 0 0 1 0	0 0 0 0 1 1 1
2	0 0 1 0	0 0 1 0	0 0 1 0 0 0 1	0 0 1 0 1	0 0 0 0 0 0 0 1 0 0	0 0 0 1 0 0 1
3	0 0 1 1	0 0 1 1	0 1 0 0 0 0 1	0 0 1 1 0	0 0 0 0 0 0 1 0 0 0	0 0 1 1 1 1 0
4	0 1 0 0	0 1 0 0	1 0 0 0 0 0 1	0 1 0 0 1	0 0 0 0 0 1 0 0 0 0	0 1 0 1 0 1 0
5	0 1 0 1	1 0 1 1	0 0 0 0 1 1 0	0 1 0 1 0	0 0 0 0 1 0 0 0 0 0	0 1 0 1 1 0 1
6	0 1 1 0	1 1 0 0	0 0 0 1 0 1 0	0 1 1 0 0	0 0 0 1 0 0 0 0 0 0	0 1 1 0 0 1 1
7	0 1 1 1	1 1 0 1	0 0 1 0 0 1 0	1 0 0 0 1	0 0 1 0 0 0 0 0 0 0	0 1 1 0 1 0 0
8	1 0 0 0	1 1 1 0	0 1 0 0 0 1 0	1 0 0 1 0	0 1 0 0 0 0 0 0 0 0	1 0 0 1 0 1 1
9	1 0 0 1	1 1 1 1	1 0 0 0 0 1 0	1 0 1 0 0	1 0 0 0 0 0 0 0 0 0	1 0 0 1 1 0 0
Stellenwert	8 4 2 1	2 4 2 1	4 3 2 1 0 5 0	7 4 2 1 0 für die Ziffern 1 bis 9	9 8 7 6 5 4 3 2 1 0	keine (Hamming- distanz: 3)
Anwendung	Standard-Code	Taschenrechner	Rechenbrett (Abakus)	Postleitzahlen, Waren	Tastenfelder, Anzeigen	Datenübertragung

¹ Richard W. Hamming, am. Mathematiker, 1915-1998

² Howard H. Aiken, am. Mathematiker, 1900-1973

Gray-Code

Der Gray¹-Code ist ein einschrittiger Code, d. h. von einer codierten Dezimalziffer zur nächsten ändert sich nur in einer einzigen Binärstelle der Wert. Deshalb wird der Gray-Code zur Längenmessung und Winkelmessung verwendet. Dabei lesen Fotodioden die einzelnen Codewörter ein (**Bild 1**).

2D-Codes

2D-Codes bestehen aus quadratischen Flächen, deren Mustern Informationen enthalten (**Bild 2**).

Data-Matrix-Code: Er wird für die elektronische Briefmarke, den 2D-Pharma-Code, Markierungen in der Medizintechnik und Luftfahrt verwendet.

Maxi-Code: Mit ihm werden Pakete identifiziert und deren Versand verfolgt. Er enthält z.B. das Gewicht, die Serviceart und die Adressangaben.

QR-Code: Mit ihm werden industriell gefertigte Teile gekennzeichnet. Diesen können dann produktionsbezogene Unterlagen, z.B. Konstruktionspläne aus einer Datenbank, zugeordnet werden.

Strichcodes (Barcodes)

Zum maschinellen Erfassen von Waren, z.B. an der Kasse eines Kaufhauses oder bei der Inventur in einem Warenlager, wird der EAN-Code (EAN von Europäische Artikel Nummerierung) eingesetzt. Jede Ziffer des Codes wird mit 7 bit verschlüsselt, wobei zur Verschlüsselung ein Codewort aus einem von drei unterschiedlichen Zeichensätzen gewählt werden kann (**Tabelle 1**).

Der EAN-Code wird mit einem Lesestift oder einem Lesescanner abgetastet (gelesen).

Der EAN-Code beginnt und endet mit einem Randzeichen. Dazwischen befinden sich zwölf codierte Dezimalziffern und in deren Mitte ein Trennzeichen (**Bild 3**). Die linken 6 Ziffern werden für europäische Artikel in der Zeichensatzfolge ABAABB codiert, während die rechten 6 Ziffern immer mit dem Zeichensatz C codiert werden.

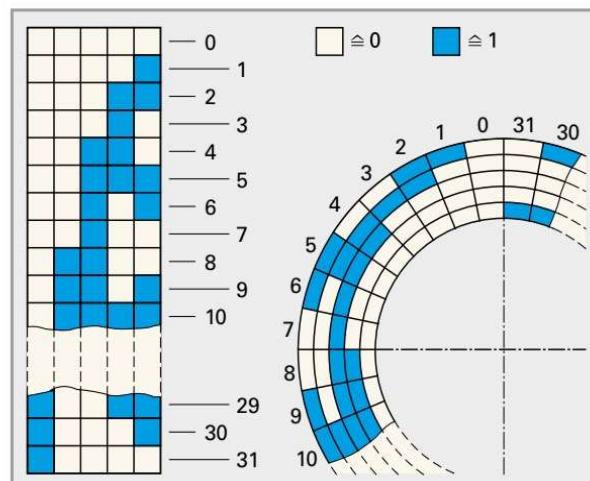


Bild 1: Codelineal und Codescheibe mit Gray-Code

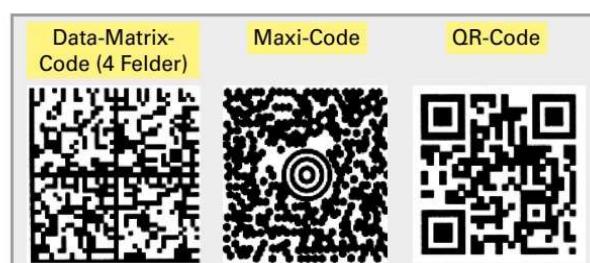


Bild 2: 2D-Codes

Tabelle 1: Zeichensätze des EAN-Codes

Zeichen	Zeichensatz A	Zeichensatz B	Zeichensatz C
0	0001101	0100111	1110010
1	0011001	0110011	1100110
2	0010011	0011011	1101100
3	0111101	0100001	1000010
4	0100011	0011101	1011100
5	0110001	0111001	1001110
6	0101111	0000101	1010000
7	0111011	0010001	1000100
8	0110111	0001001	1001000
9	0001011	0010111	1110100

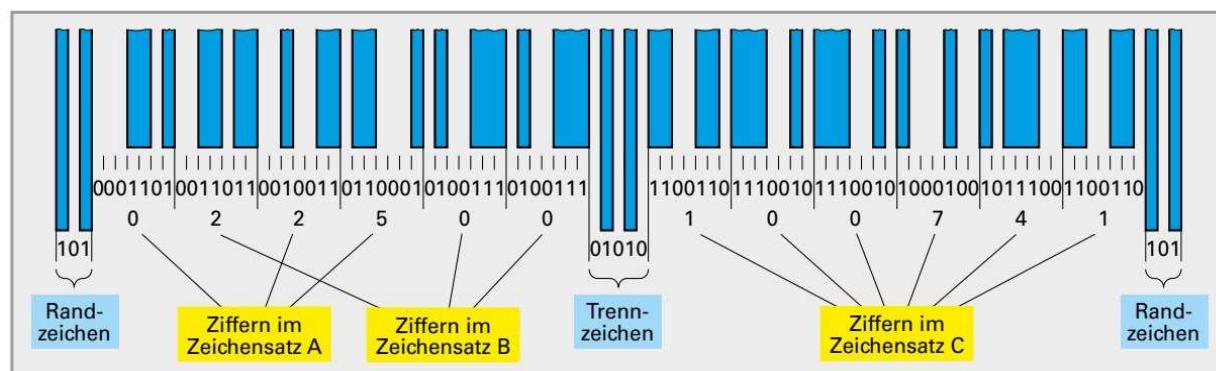


Bild 3: Decodierung eines EAN-Codes

¹ Frank Gray, amerikanischer Elektroingenieur, 1887 bis 1969

Zur automatisierten Verteilung wird auf Briefe und Postkarten eine Folge fluoreszierender Striche aufgedruckt. Ein Strich entspricht dem Binärwert 0 und ein fehlender Strich dem Wert 1. Die Strichfolge wird von rechts nach links gelesen. Bei der Codierung werden zwei verschiedene Codes verwendet, ein Code mit 80 Stellen und ein Code mit 36 Stellen.

Der 36-stellige Code (**Bild 1**) enthält eine Prüfziffer und die fünf Ziffern der Postleitzahl, welche alle im 2-aus-5-Code verschlüsselt sind. Vor allen sechs Ziffern steht der binäre Wert 0 (Strich). Die Prüfziffer erhält man, indem man die Quersumme der Dezimalziffern der Postleitzahl bildet, das Ergebnis durch 10 teilt und den Teilerrest auf 10 ergänzt.

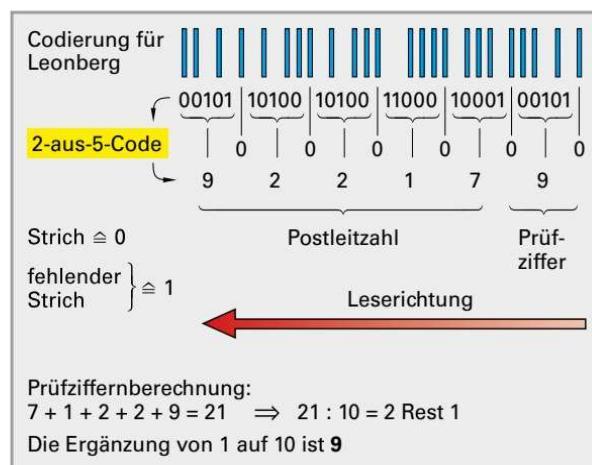


Bild 1: 36-stelliger Postleitzahlencode

6.1.4 Logische Funktionen

UND-Funktion

Ein PC besitzt z. B. neben dem Einschalter ein Schloss, um unbefugtes Einschalten zu verhindern (**Bild 2**). Das PC-Netzteil wird nur mit Strom versorgt, wenn der Einschalter und der Schlüsselschalter betätigt werden. Die UND-Funktion (UND-Verknüpfung) kann somit aus zwei oder mehreren in Reihe geschalteten Schließen dargestellt werden (**Bild 3**). Das Schaltzeichen (Symbol) dieser Verknüpfung hat die Eingangssignale a und b und das Ausgangssignal z .

Das Ausgangssignal z der UND-Verknüpfung hat nur dann den Wert 1, wenn alle Eingangssignale den Wert 1 führen.

Die Funktionsgleichung mit zwei Eingangssignalen lautet $z = a \wedge b$ (sprich: und). Das Zeichen \wedge wird oft weggelassen, wenn Verwechslungen ausgeschlossen sind, d. h. $a \wedge b = ab$.

Die Wertetabelle der UND-Funktion (**Bild 3**) hat bei n Eingangssignalen 2^n verschiedene Wertekombinationen der Eingangssignale, also bei zwei Signalen $2^2 = 4$. Diese werden untereinander in Zeilen geschrieben. Die Werte der Eingangskombinationen steigen im Dualcode 00, 01, 10 und 11. Entsprechend werden die Zeilen nummeriert. Das Ausgangssignal z erhält den Wert entsprechend der Schalterstellungen der Schließer A und B, wobei ein unbetätigter Schließer den Wert 0 führt und ein betätigter Schließer den Wert 1.

Das KV-Diagramm (Karnaugh¹-Veitch²-Diagramm) ist eine gleichwertige Darstellung für die Wertetabelle. Die vier Zeilen der Tabelle werden durch vier Quadrate ersetzt. Bei den zwei rechten Quadranten.

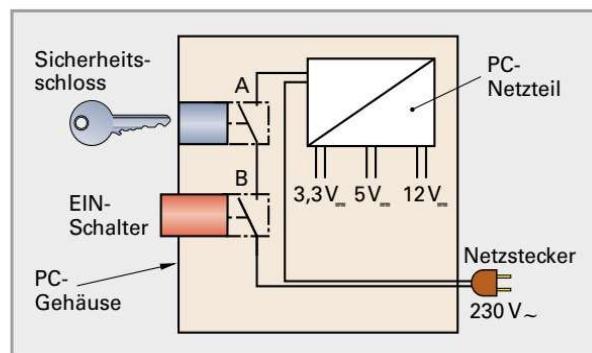


Bild 2: UND-Funktion

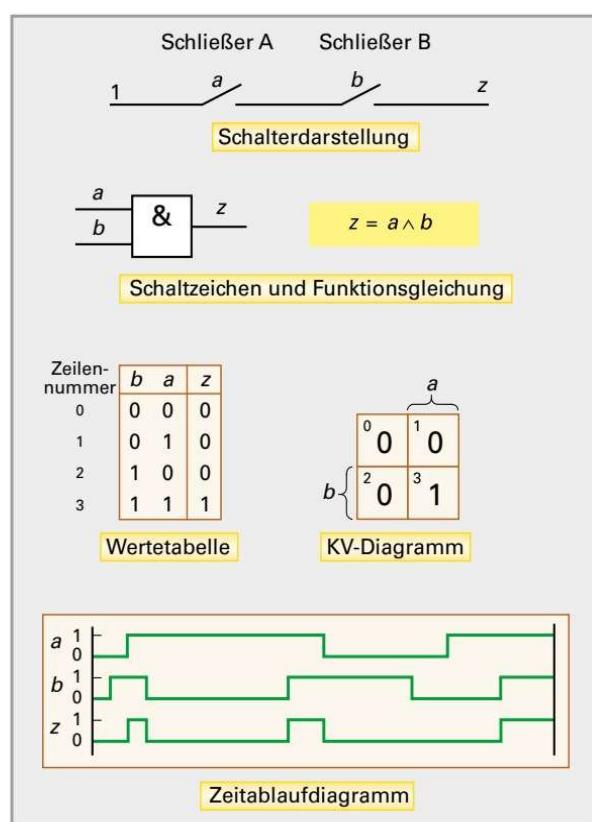


Bild 3: Darstellungsformen der UND-Funktion bei 2 Eingängen

¹ Maurice Karnaugh, amerikanischer Elektroingenieur, geb. 1924

² Edward Veitch, amerikanischer Mathematiker, 1924 bis 2013

führt das Signal a den Wert 1, bei den beiden unteren Quadranten führt das Signal b den Wert 1. Somit stehen die Quadrate 0,1,2 und 3 für die Zeilen 0,1,2 und 3 der Wertetabelle (Bild 3, vorhergehende Seite). Die Werte für das Ausgangssignal z werden aus der Wertetabelle in das entsprechende Quadrat des KV-Diagramms übertragen.

Logische Verknüpfungen sind in logischen integrierten Schaltkreisen (IC = integrated circuit) verwirklicht. Ein IC enthält ein oder mehrere logische Elemente, z. B. 2 UND-Elemente mit je 4 Eingängen (**Bild 1**). Ist, wie abgebildet, die Kerbe des ICs oben, beginnt die Nummerierung der Anschlüsse (Pins) links oben im Gegenuhrzeigersinn. Nicht verwendete Pins werden mit NC (not connected = nicht verbunden) bezeichnet. Bei 14-poligen ICs im DIL-Gehäuse (Dual-In-Line = zweireihig) erfolgt der Spannungsanschluss, z. B. 3,3 V, meist über die Pins 14 und 7. Die ICs gibt es in verschiedenen Ausführungen, z. B. aus bipolaren Transistoren (TTL-Technik) oder aus Feldeffekttransistoren (C-MOS-Technik). Die einzelnen Typen unterscheiden sich vor allem in der Schaltgeschwindigkeit, z. B. 2 ns, und im Leistungsverbrauch je Element, z. B. 5 mW.

ODER-Funktion

Ein Garagentor (**Bild 2**) kann entweder durch einen Handsender oder durch einen Handschalter in der Garage geöffnet werden. Der Motor Z öffnet das Tor, wenn mindestens einer der Schalter betätigt wurde. Die ODER-Funktion kann somit aus zwei oder mehreren parallelen Schließern dargestellt werden. Bei 3 der 4 möglichen Schalterkombinationen erhält der Ausgang den Wert 1 (**Bild 3**).

Das Ausgangssignal z der ODER-Verknüpfung hat dann den Wert 1, wenn mindestens ein Eingangssignal den Wert 1 führt.

Die Funktionsgleichung mit 2 Eingangssignalen lautet $z = a \vee b$ (v sprich: oder, v von lat. vel = oder). Im Gegensatz zum UND-Operator \wedge darf der ODER-Operator \vee in Gleichungen nicht weggelassen werden.

Beispiel 1: Diagrammgrößen

Eine ODER-Verknüpfung hat 3 Eingangssignale. Wie viele Zeilen hat die Wertetabelle und wie viele Quadrate das KV-Diagramm?

Lösung:

$2^3 = 8$ Zeilen, bzw. 8 Quadrate

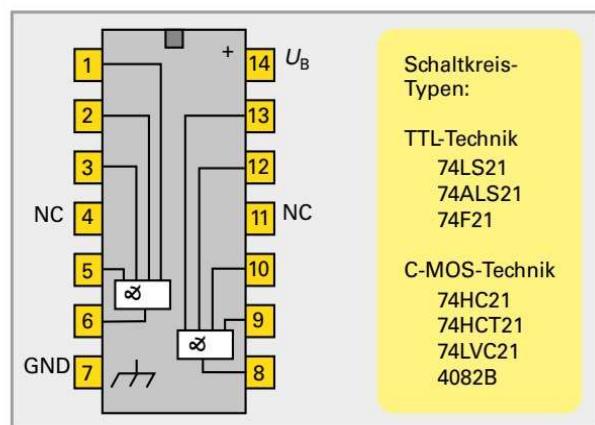


Bild 1: IC mit 2 UND-Elementen

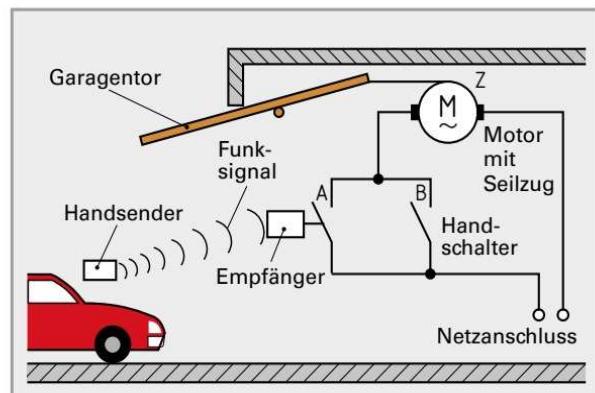


Bild 2: ODER-Funktion

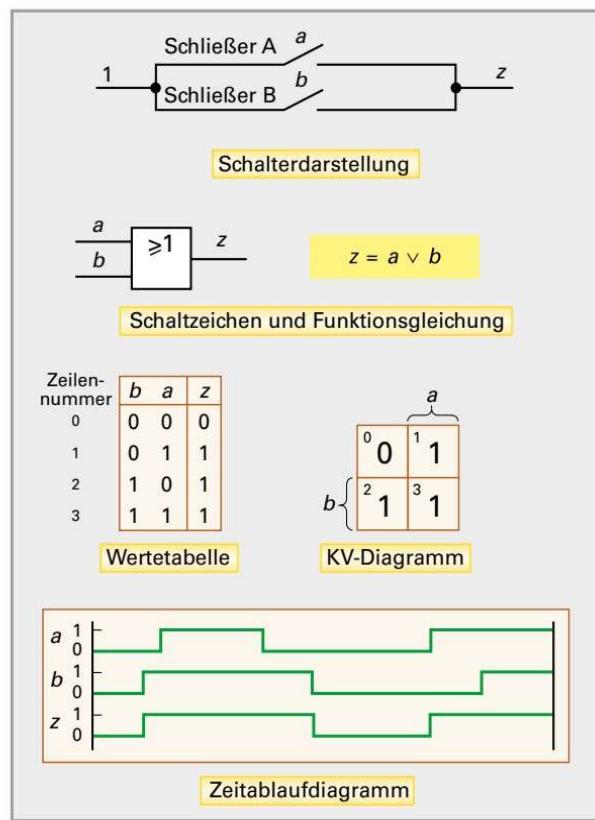


Bild 3: Darstellungsformen der ODER-Funktion bei 2 Eingängen

NICHT-Funktion

Die Innenleuchte eines Kühlschranks (**Bild 1**) erleuchtet, wenn die Kühlschranktür geschlossen und der Öffner \bar{A} betätigt wird. Das Signal \bar{a} (sprich: a nicht, oder: nicht a) ist die Invertierung des Signals a . Beim unbetätigten Öffner (**Bild 2**) ist $\bar{a} = 1$ und damit $a = 0$. Das Signal z hat dann den Wert 1. Dies entspricht der ersten Zeile der Wertetabelle. Die zweite Zeile der Wertetabelle stellt den betätigten Öffner dar. Das Zeitablaufdiagramm zeigt, dass die Signale a und z stets verschiedene Werte haben. Sie sind komplementär.

Die Signale z und a sind bei der NICHT-Funktion immer komplementär.

Dies zeigt auch das Zeitablaufdiagramm (Bild 2).

NAND-Funktion

Auf einem Schrottplatz wird ein defektes Auto mit einem Kran zur Schrottpresse befördert. Das Auto hängt an dem Hubmagneten am Ende des Kranarmes (**Bild 3**). Der Kranführer muss mit jeder Hand einen der beiden Öffner betätigen, damit der Hubmagnet stromlos wird und das Auto frei gibt. Der zweite Öffner dient als Schutzfunktion, damit nicht bei zufälligem Betätigen eines Öffners Personen am Schrottplatz gefährdet werden.

Die Wertetabelle (**Bild 4**) zeigt, dass sich das Ausgangssignal z für alle 4 Schalterkombinationen genau komplementär zur UND-Verknüpfung verhält. Deshalb ist das Schaltzeichen zusammengesetzt aus der UND-Funktion und dem Kreiszeichen der NICHT-Funktion. Bei der Funktionsgleichung ist das Ergebnis der UND-Verknüpfung der Signale a und b durch einen langen Negierungsbalken invertiert.

Das Ausgangssignal z der NAND-Verknüpfung hat dann den Wert 1, wenn nicht alle Eingangssignale gleichzeitig den Wert 1 führen.

Da die NAND-Verknüpfung eine Parallelschaltung aus zwei Öffnern ist (**Bild 3**), kann sie auch als ODER-Verknüpfung von zwei invertierten Signalen dargestellt werden, also durch $\bar{a} \vee \bar{b}$ (**Bild 4**). Das entspricht einem ODER-Schaltzeichen, bei welchem die Eingänge durch Kreiszeichen invertiert sind (**Bild 4**).

Ein UND mit negiertem Ausgang verhält sich gleich wie ein ODER mit negierten Eingängen.

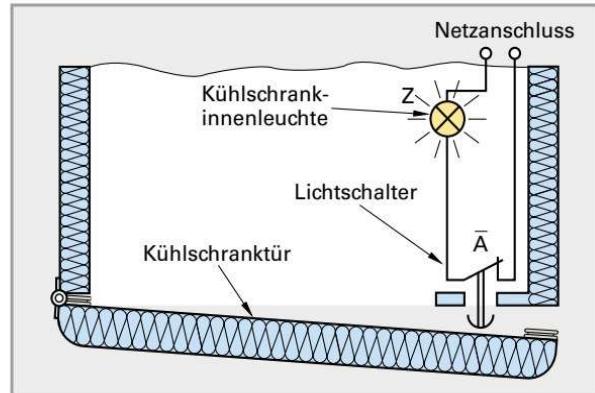


Bild 1: NICHT-Funktion

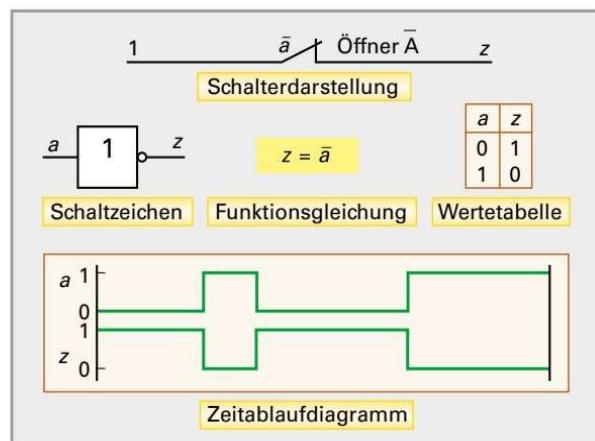


Bild 2: Darstellungsformen der NICHT-Funktion

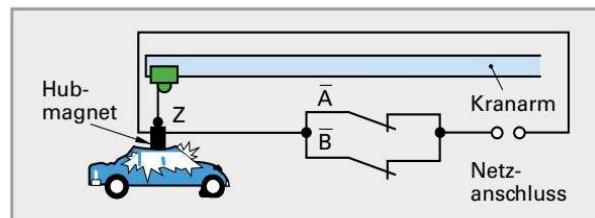


Bild 3: NAND-Funktion

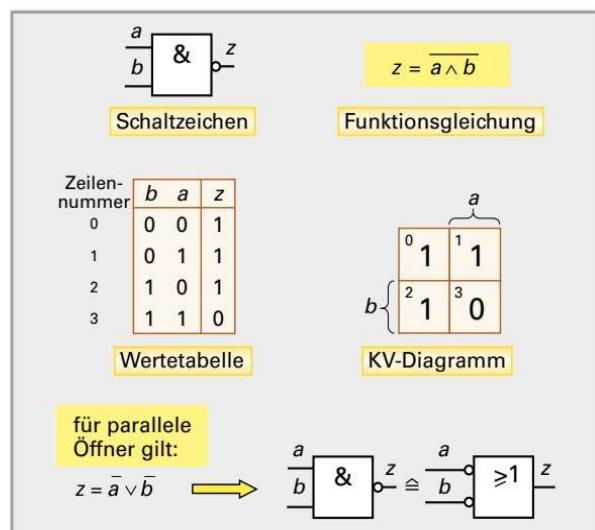


Bild 4: Darstellungsformen der NAND-Funktion

NOR-Funktion

Ein Elektromotor wird in einer Ölwanne mit Öl gekühlt (**Bild 1**). Ein Sicherheitssystem schützt den Motor vor Überhitzung, indem es ihn vom Netz trennt, wenn entweder der Bimetallschalter durch zu hohe Temperatur betätigt wird oder wenn ein Schwimmschalter bei Ölverlust betätigt wird.

Die Wertetabelle (**Bild 2**) zeigt, dass der Motor nur dann Strom erhält, wenn beide Öffner unbetätigt sind. Das Ausgangssignal z verhält sich bei allen Schalterkombinationen komplementär zur ODER-Funktion. Deshalb setzt sich das Schaltzeichen aus dem der ODER-Funktion und aus dem der NICHT-Funktion zusammen.

Das Ausgangssignal z der NOR-Verknüpfung hat dann den Wert 1, wenn keines der Eingangssignale den Wert 1 führt.

Da die NOR-Schaltung eine Reihenschaltung aus zwei Öffnern ist (**Bild 1**), kann sie auch als UND-Verknüpfung von zwei invertierten Signalen dargestellt werden, also $\bar{a} \wedge \bar{b}$ (**Bild 2**). Das entspricht einem UND-Schaltzeichen, bei welchem die Eingänge durch Kreiszeichen invertiert sind (**Bild 2**).

Ein ODER mit negiertem Ausgang verhält sich gleich wie ein UND mit negierten Eingängen.

XOR-Funktion (Exklusiv-ODER)

In einem Treppenhaus kann die Beleuchtung wahlweise im Erdgeschoss oder im Obergeschoss eingeschaltet bzw. ausgeschaltet werden (**Bild 3**). Die Lampe leuchtet nur, wenn einer der beiden Wechselschalter in Stellung 1 gebracht wird. Werden beide Schalter betätigt, bleibt das Licht aus.

Das Ausgangssignal z der XOR-Verknüpfung hat dann den Wert 1, wenn nur eines von zwei Eingangssignalen den Wert 1 führt.

Die Funktionsgleichung kann aus den Zeilen 1 und 2 der Wertetabelle entnommen werden (**Bild 4**). Das Signal z hat dann den Wert 1, wenn $a = 1$ und $b = 0$, also $a \wedge \bar{b}$ ist oder wenn $a = 0$ und $b = 1$, also $\bar{a} \wedge b$, ist. Daraus ergibt sich die Gleichung $z = a \wedge \bar{b} \vee \bar{a} \wedge b$ oder auch $z = ab \vee \bar{a}b$. Auf eine Klammer kann verzichtet werden, da der UND-Operator Vorrang vor dem ODER-Operator hat.

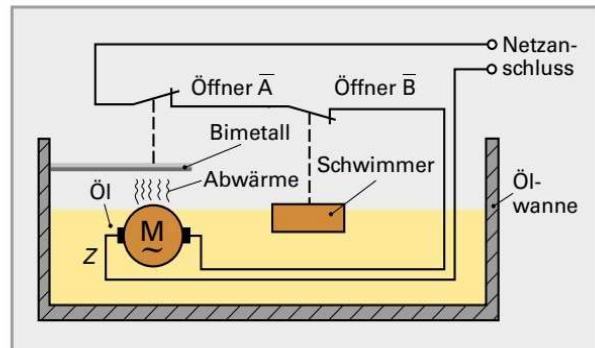


Bild 1: NOR-Funktion

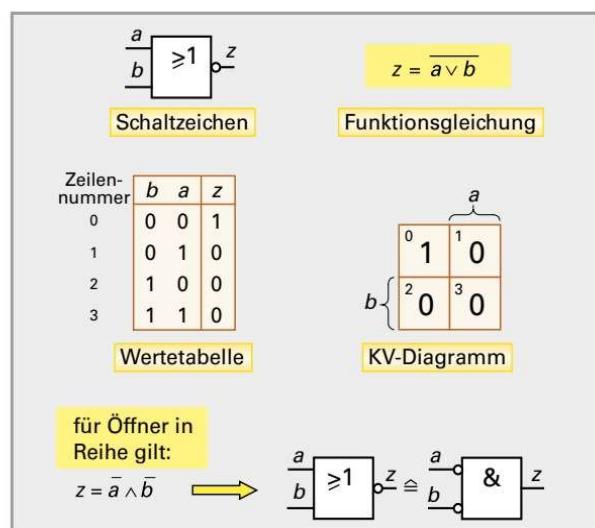


Bild 2: Darstellungsformen der NOR-Funktion

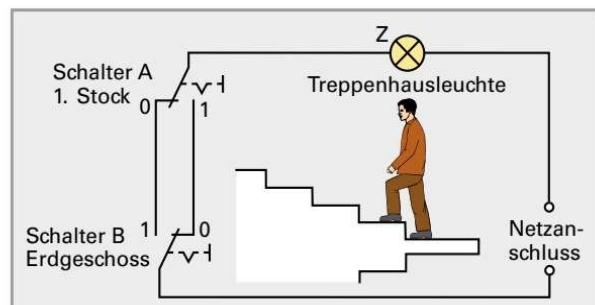


Bild 3: Exklusiv-ODER-Funktion

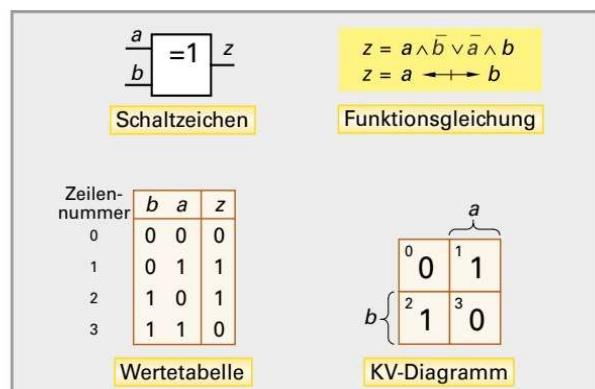


Bild 4: Darstellungsformen der XOR-Funktion

Es gilt immer: UND (\wedge) vor ODER (\vee).

Da bei der XOR-Funktion das Ausgangssignal z nur dann den Wert 1 führt, wenn die beiden Eingangssignale komplementäre Werte haben, nennt man die Funktion auch Antivalenz-Funktion (von lat. antivalens = entgegengesetzt wertig).

XNOR-Funktion (Exclusiv-NICHT-ODER)

Auf einer PC-Karte kann mit einem Jumper (Stecker, von to jump = springen) die Adresse 0 oder 1 eingestellt werden (Bild 1). Mit dem Adresswahlschalter A soll auf die Karte zugegriffen werden. Die Freigabe $z = 1$ erfolgt nur, wenn die beiden Adressen übereinstimmen.

Das Ausgangssignal z der XNOR-Verknüpfung hat dann den Wert 1, wenn alle Eingangssignale den gleichen Wert führen.

Aus der Wertetabelle (Bild 2) entnimmt man, dass für $\bar{a} \wedge \bar{b}$ oder auch für $a \wedge b$ das Signal $z = 1$ ist. Damit gilt $z = \bar{a} \wedge \bar{b} \vee a \wedge b$. Das Ausgangssignal z ist bei allen 4 Zeilen komplementär zur XOR-Funktion, deshalb erhält man z auch aus einem XOR-Element und einem nachgeschalteten NICHT-Element (Bild 2). Da nur bei Gleichheit der Eingangssignalwerte der Ausgang den Wert 1 führt, nennt man die XNOR-Funktion auch Äquivalenz-Funktion (von lat. aequivalens = gleichwertig).

Normen logischer Funktionen

Die im Buch dargestellten Schaltzeichen entsprechen der IEC-Norm (International Electrotechnical Commission = internationale elektrotechnische Kommission). Obwohl die Norm international gültig ist, wird sie bei Datenblättern von ICs für logische und programmierbare logische Elemente von den IC-Herstellern nicht verwendet. Da die IC-Hersteller meist amerikanische Firmen sind, wird die amerikanische Norm ASA (Typ B) verwendet (Tabelle 1).

Für die Programmierung von logischen Elementen fehlt auf der PC-Tastatur der UND-Operator \wedge . Deshalb benutzt man für logische Verknüpfungen oft die Rechenzeichen $*$, $+$ und das Zeichen $/$ (slash).

Die Zeichen $*$ oder $\&$ entsprechen dem UND-Operator \wedge .

Die Zeichen $+$ oder $\#$ entsprechen dem ODER-Operator \vee .

Die Zeichen $/$ oder $!$ entsprechen dem NICHT-Operator.

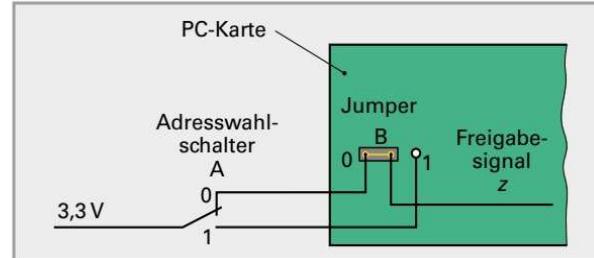


Bild 1: XNOR-Funktion

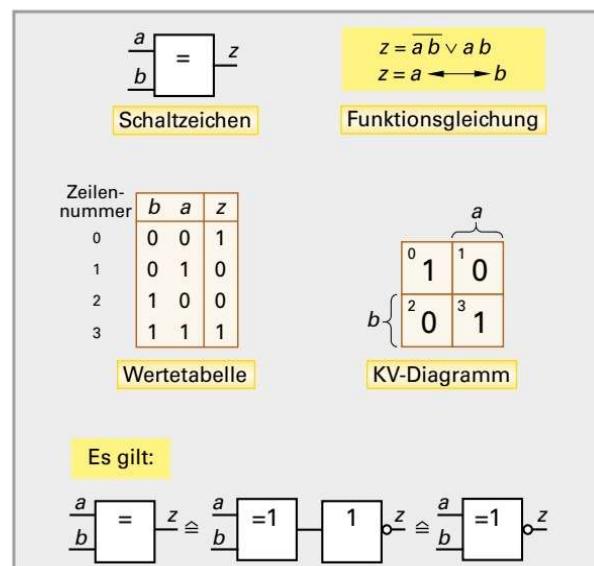


Bild 2: Darstellungsformen der XNOR-Funktion

Tabelle 1: Schaltzeichen nach verschiedenen Normen

Funktion	IEC-Norm	am. Norm ASA (Typ B)	alte deutsche Norm	am. Norm ASA (Typ A)
UND				
ODER				
NICHT				
NAND				
NOR				
XOR				
XNOR				

6.1.5 Boole'sche Algebra

Rechengesetze

Rechengesetze der Boole'schen¹ Algebra (Schaltalgebra) sind das Kommutativgesetz KG, das Assoziativgesetz AG und das Distributivgesetz DG (Tabelle 1).

Das KG, auch Vertauschungsgesetz, besagt, dass bei logischen Elementen die Eingänge gleichwertig sind und somit vertauscht werden dürfen. Nach dem AG, auch Verbindungsgesetz, dürfen UND-Elemente oder ODER-Elemente mit mehreren Eingängen durch mehrere Elemente mit nur 2 Eingängen ersetzt werden. Man nennt dies auch Kaskadieren von Elementen (Tabelle 1). Mithilfe des DG, auch Verteilungsgesetz, kann ein UND-Element eingespart werden (Tabelle 1).

Rechenregeln

Eine häufig verwendete Schaltung ist die Torschaltung. Sie besteht in der Regel aus einem UND-Element (**Bild 1**). Ein Eingang des UND-Tores bildet den Steuereingang zur Freigabe und zum Sperren des Tores. Ist das Tor freigegeben, ist am Torausgang dasselbe Signal wie am Toreingang, egal welchen Signalwert es besitzt. Es gilt $a \wedge 1 = a$. Bei gesperrtem Tor gilt $a \wedge 0 = 0$. Entsprechende Gleichungen erhält man beim ODER-Element (**Bild 1**).

Wird bei einem Logikelement ein Eingang nicht benötigt, kann er mit einem benachbarten Eingang überbrückt werden. Daraus lassen sich die Rechenregeln aus **Bild 2** ableiten.

Wird ein Signal a mit sich selbst UND-verknüpft oder ODER-verknüpft, ist das Ergebnis das Signal a .

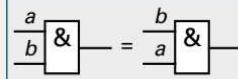
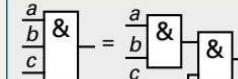
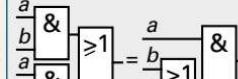
Bei programmierbaren Logikelementen kommt es vor, dass ein Signal mit der eigenen Invertierung verknüpft werden muss (**Bild 3**). Je nach logischer Funktion erhält man den Wert 0 oder 1.

Wird ein Signal nach dem Invertieren nochmals invertiert, erhält es seinen ursprünglichen Wert (**Bild 4**). Es gilt $\bar{\bar{a}} = a$.

Mit den Gesetzen nach de Morgan² (**Bild 5**) lassen sich logische Schaltungen algebraisch oder auch grafisch vereinfachen. Somit ist eine NAND-Funktion gleich einer ODER-Funktion mit negierten Eingängen und eine NOR-Funktion ist gleich einer UND-Funktion mit negierten Eingängen.

Merke: Break the line, change the sign.

Tabelle 1: Rechengesetze

Art	Gleichungen	Beispiele
KG	$a \wedge b = b \wedge a$ $a \vee b = b \vee a$	
AG	$a \wedge b \wedge c = (a \wedge b) \wedge c$ $a \vee b \vee c = a \vee (b \vee c)$	
DG	$a \wedge b \vee a \wedge c = a \wedge (b \vee c)$ bzw. $ab \vee ac = a(b \vee c)$	

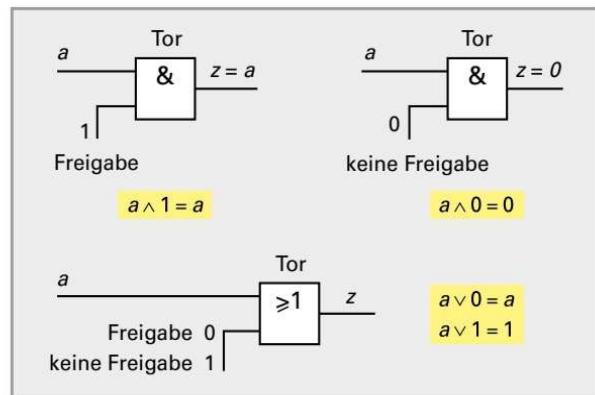


Bild 1: Torschaltungen

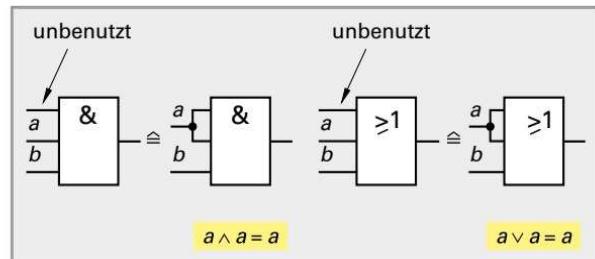


Bild 2: Beschaltung unbenutzter Eingänge

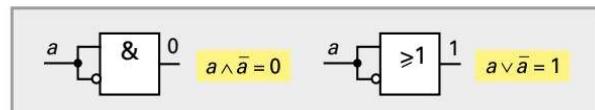


Bild 3: Verknüpfungen von a mit \bar{a}

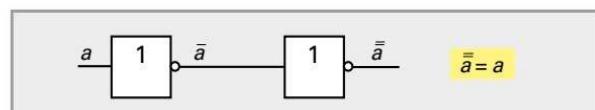


Bild 4: Doppelte Invertierung

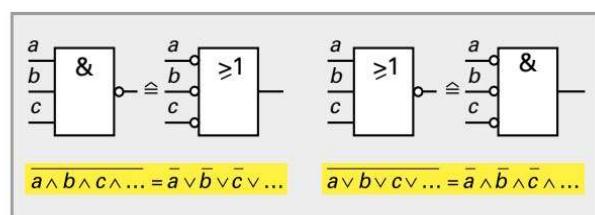


Bild 5: Gesetze nach de Morgan

¹ George Boole, englischer Mathematiker und Philosoph, 1815 bis 1864

² Augustus de Morgan, englischer Mathematiker, 1806 bis 1871

Beispiel 1: Schaltungen vereinfachen

Zum Aufbau der Logikschaltung (**Bild 1**) benötigt man 3 verschiedene IC. Die Schaltung soll so verändert werden, dass sie mit dem IC 7400 (Bild 1) verwirklicht werden kann. Vereinfachen Sie die Schaltung nach den Regeln der Boole'schen Algebra a) auf schaltalgebraischem Weg, b) auf grafischem Weg.

Lösung:

a) 1. Schritt: Gleichung aufstellen.

$$z = a \wedge b \vee \bar{c}$$

2. Schritt: Doppelt negieren.

$$z = a \wedge b \vee \bar{\bar{c}}$$

3. Schritt: Gesetz nach de Morgan anwenden.

$$z = a \wedge b \wedge \bar{c}$$

4. Schritt: Doppelte Negierung aufheben.

$$z = a \wedge b \wedge c$$

z ist durch 2 NAND-Elemente mit 2 Eingängen realisierbar.

b) **Bild 2**

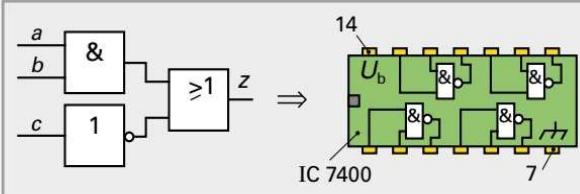


Bild 1: Problemstellung zu Beispiel 1

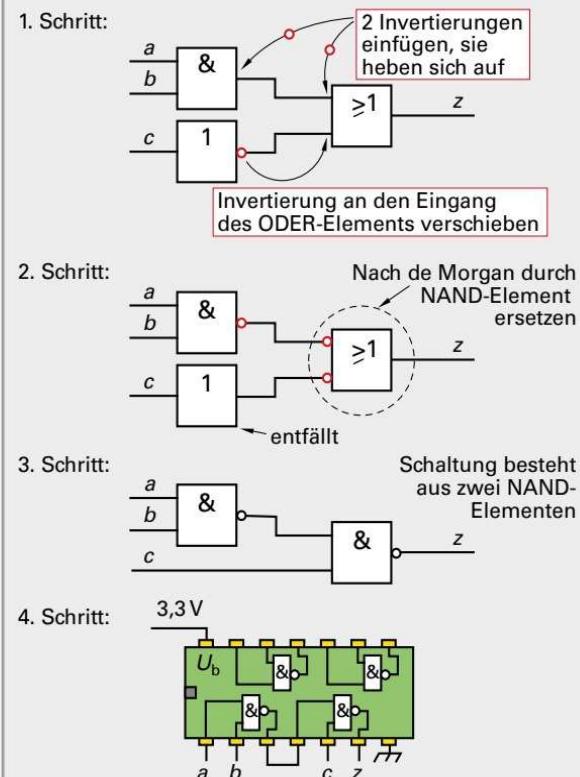


Bild 2: Grafische Vereinfachung nach de Morgan

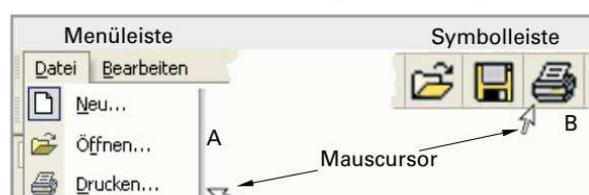


Bild 3: Erzeugen des Signals z für Drucken

Der Freiheitsgrad X kann wahlweise auf 0 oder auf 1 gesetzt werden.

Man wird den Freiheitsgrad immer so wählen, dass sich die logische Schaltung vereinfacht, wenn dies möglich ist.

Beispiel 2: Logische Funktion ermitteln

Ermitteln Sie die Funktion für das Signal z (Drucken).

Lösung:

Bild 4, $z = a \vee b$

1. Schritt: Erstellen einer Wertetabelle

Zeilennummer	b	a	z
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	X

Für X kann wahlweise der Signalwert 0 oder der Signalwert 1 gesetzt werden.

2. Schritt: Ersetzen von X durch einen Wert

- a) $X = 0$: XOR-Verknüpfung b) $X = 1$: ODER-Verknüpfung

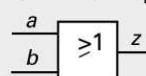
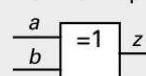


Bild 4: Ermitteln der Verknüpfung für das Signal z

6.1.6 Entwicklung logischer Schaltungen

ODER-Normalform DNF

In einem Tunnel befinden sich drei Belüftungsventilatoren (**Bild 1**). Der Ventilator A hat eine höhere Leistung als die Ventilatoren B und C. Fällt der Ventilator A aus ($a = 0$) oder fällt mehr als ein Ventilator aus, leuchtet die Alarmleuchte Z auf ($z = 1$).

Die Wertetabelle zur Ermittlung der Schaltung hat drei Eingangsvariablen und damit $2^3 = 8$ Zeilen, welche von 0 bis 7 durchnummeriert werden.

Beispiel 1: Wertetabelle erstellen

Stellen Sie die Wertetabelle für das Alarmsignal auf.

Lösung: Bild 2

Die Eingangsterme der Zeilen, bei welchen $z = 1$ ist, sind Teilmengen der Lösung für z. Das Signal z ist die ODER-Verknüpfung dieser Terme (**Bild 3**). Man nennt diese Verknüpfung ODER-Normalform oder DNF (disjunktive Normalform, lat. disjunctus = getrennt).

Die DNF ist die ODER-Verknüpfung der Eingangsterme, für welche der Ausgang den Wert 1 führt.

Um die gewonnene Gleichung für z schaltungstechnisch zu realisieren, benötigt man NICHT-Elemente, UND-Elemente und ODER-Elemente, im speziellen Fall insgesamt vier ICs. Platzsparender ist, die Schaltung in einem programmierbaren IC bzw. PLD (programmable logic device = programmierbare Logikschaltung) zu programmieren. Ein PLD besitzt intern bereits die Verknüpfungsstruktur einer DNF (**Bild 4**). Programmiert werden müssen nur die Verbindungen zwischen den Eingangssignalen und den Produktlinien. Diese Verbindungen werden durch Kreuze (x) gekennzeichnet (Bild 4). Eine Produktlinie besteht aus vielen einzelnen Leitern, aber nur solche sind wirksam, welche verbunden sind. Z. B. führt die Produktlinie 0 (Bild 4) die Signale \bar{a} , \bar{b} und \bar{c} zum unteren UND-Element.

Ein PLD enthält die Verknüpfungsstruktur der DNF.

Um ein PLD zu programmieren, benötigt man ein Brenngerät, eine Brennsoftware und das Programm, welches die Verbindungen im Bauelement herstellt. Das Programm enthält die Anschlussbelegung der Signale und die Gleichung für z (**Bild 5**). Bei der Programmerstellung werden die Zeichen * und + für UND und ODER und der Slash / für die Negierung verwendet.

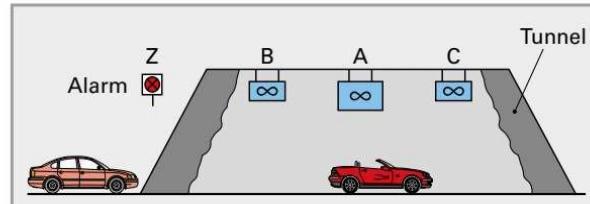


Bild 1: Tunnelbelüftung

Zeilennummer	c	b	a	z	
0	0	0	0	1	$\bar{a} \wedge \bar{b} \wedge \bar{c}$
1	0	0	1	1	$a \wedge \bar{b} \wedge \bar{c}$
2	0	1	0	1	$\bar{a} \wedge b \wedge \bar{c}$
3	0	1	1	0	$\bar{a} \wedge b \wedge c$
4	1	0	0	1	$\bar{a} \wedge \bar{b} \wedge c$
5	1	0	1	0	$a \wedge \bar{b} \wedge c$
6	1	1	0	1	$a \wedge b \wedge \bar{c}$
7	1	1	1	0	$a \wedge b \wedge c$

Terme,
bei welchen
 $z = 1$ ist.

Bild 2: Wertetabelle für das Alarmsignal

$$z = \bar{a}\bar{b}\bar{c} \vee a\bar{b}\bar{c} \vee \bar{a}b\bar{c} \vee \bar{a}\bar{b}c \vee a\bar{b}c$$

Bild 3: DNF für das Alarmsignal

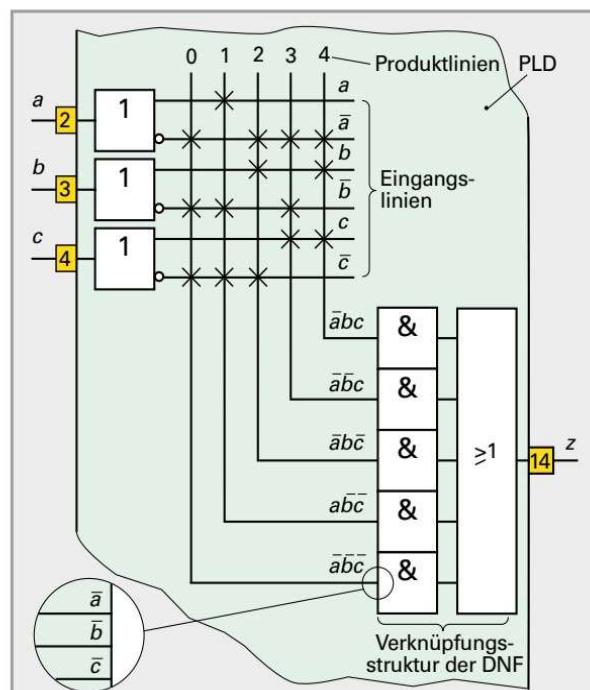


Bild 4: Realisierung der DNF in einem PLD

```

title tunnel
chip tunnel GAL16V8
;1 2 3 4 5 6 7 8 9 10
nc a b c nc nc nc nc nc GND
;11 12 13 14 15 16 17 18 19 20
nc nc nc z nc nc nc nc nc VCC
equations
z=/a*/b*/c+a*/b*/c+/c+a*b*/c
+/a*/b*c+/a*b*c

```

Bild 5: Programmierung eines PLD, hier GAL 16V8

KV-Diagramm

Eine minimierte (kürzeste) Form der Schaltfunktion für das Ausgangssignal z (Bild 3, vorhergehende Seite) liefert das KV-Diagramm. Da die Wertetabelle 8 Zeilen besitzt, besteht das KV-Diagramm aus 8 Quadranten. Die Signalzuordnung zeigt **Bild 1**.

Beispiel 1: KV-Diagramm erstellen

Übertragen Sie die Werte für z in das KV-Diagramm.

Lösung: **Bild 1 und Bild 2**

Zeilenummer	c	b	a	z
0	0	0	0	1
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	0

Bild 1: KV-Diagramm für z

Um die Schaltfunktion für z minimiert zu erhalten, müssen immer möglichst viele Quadrate mit den Werten $z = 1$ zusammengefasst werden. Dabei können benachbarte und gegenüber liegende Quadrate in Vierergruppen oder Zweiergruppen zusammengefasst werden. Die so gewonnenen Terme, hier \bar{a} und $\bar{b} \wedge \bar{c}$, werden ODER-verknüpft (Bild 2).

Zum Aufbau der Schaltung benötigt man drei IC (NICHT, UND, ODER). Die minimierte Schaltfunktion liefert zwar die kürzeste Form der Schaltfunktion, aber nicht den einfachst möglichen Schaltungsaufbau. Dazu wird die Gleichung umgeformt (**Bild 3**). Als Lösung erhält man eine Funktion, die ausschließlich mit NAND-Elementen verwirklicht werden kann, da die Invertierungen \bar{b} und \bar{c} durch NAND-Elemente mit überbrückten Eingängen ersetzt werden können (**Bild 4**).

Beispiel 2: Vereinfachte Schaltung aufbauen

Bauen Sie die Logikschaltung für z mit einem NAND-IC 74LS00 auf.

Lösung: **Bild 5**

Wie das Beispiel der Tunnelbelüftung zeigt, ist die Schaltungsentwicklung mit dem KV-Diagramm im Gegensatz zu der mit der DNF viel rechenintensiver und die Fehlerwahrscheinlichkeit höher. Die DNF liefert zwar die längere Schaltfunktion, diese lässt sich aber in einem einzigen IC, einem PLD, programmieren. Die Schaltzeiten des PLD sind ähnlich schnell wie die des NAND-Bauelements, da auch der PLD eine Hardware-Schaltung ist. Die Programmier-Software wird bei PLD nur benötigt, um die Hardware der Bauelemente zu konfigurieren (einzustellen).

Bei der Schaltungsentwicklung mit PLD ist die Erstellung der Schaltfunktion mit der DNF schneller und sicherer als mit dem KV-Diagramm.

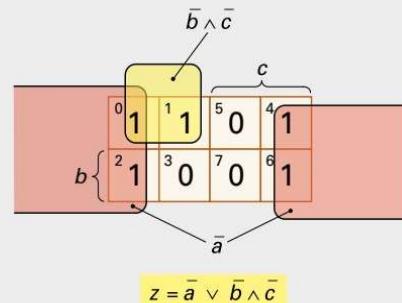


Bild 2: Ermitteln der minimierten Schaltfunktion

1. Schritt: Doppelt negieren

$$z = \overline{\overline{a} \vee \overline{b} \wedge \overline{c}}$$

2. Schritt: Gesetz von De Morgan anwenden

$$z = \overline{\overline{a}} \wedge \overline{\overline{b}} \wedge \overline{\overline{c}}$$

3. Schritt: Doppelte Negation aufheben

$$z = a \wedge b \wedge c$$

Bild 3: Umformen der minimierten Schaltfunktion



Bild 4: NICHT-Verknüpfung mit NAND-Element

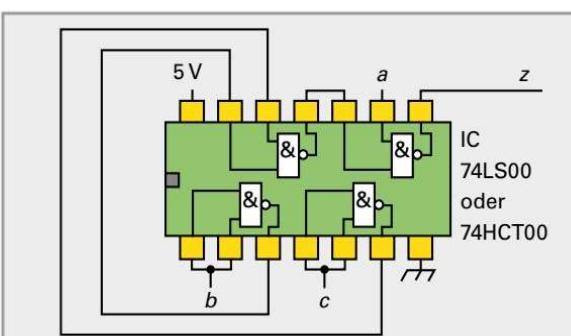


Bild 5: Schaltungsaufbau mit NAND-Bauelement

Die im Aiken-Code (Abschnitt 6.1.3) verschlüsselten Dezimalziffern sollen mit einer 7-Segment-Anzeige dargestellt werden. Die Tetraden liegen am Eingang des Codewandlers (**Bild 1**). Der Codewandler setzt jede Tetrade in ein 7-Bit-Binärwort um. Die Logikschaltung des Codewandlers ist zu entwickeln. Die Ausgänge der Logikbauelemente eignen sich nicht zur direkten Ansteuerung des Anzeigeelements, da sie durch die einzelnen Segmente zu stark belastet würden. Dies hätte zur Folge, dass die Segmente schwächer und ungleichmäßig hell leuchten. Deshalb werden Treiberelemente dazwischengeschaltet.

Die vollständige Wertetabelle für den Codewandler hat 4 Eingangsvariablen und 7 Ausgangsvariablen. Damit hat die Wertetabelle $2^4 = 16$ Zeilen, welche von 0 bis 15 durchnummieriert werden.

Beispiel 1: Wertetabelle erstellen

Stellen Sie die Wertetabelle für das Segmentsignal $e7$ auf.

Lösung: **Bild 2**

Da die Pseudotetraden des Aiken-Codes nie vorkommen, kann bei den Eingangswerten 0101 bis 1010 für das Ausgangssignal $e7$ ein Freiheitsgrad X eingetragen werden.

Den 16 Zeilen der Wertetabelle sind 16 Quadrate im KV-Diagramm zugeordnet. Die dargestellte Zuordnung (**Bild 2**) ist nicht genormt. In der Fachliteratur existieren mehrere Zuordnungsmuster. Abhängig davon wie die Signale a , b , c und d am KV-Diagramm angeordnet werden, ändert sich die Zählfolge der Quadrate. Der Aufwand bei der Ermittlung der minimierten Schaltfunktion ist jedoch bei allen Zuordnungen gleich groß.

Beispiel 2: Minimierte Schaltfunktion ermitteln

- Übertragen Sie die Signalwerte für $e7$ in das KV-Diagramm und fassen Sie die Signalwerte 1 in möglichst großen Gruppen zusammen.
- Geben Sie die minimierte Schaltfunktion an.

Lösung: a) **Bild 3** b) $e7 = \bar{a} \wedge \bar{c} \vee \bar{a} \wedge d$.

Beispiel 3: Logikschaltung zeichnen

Zeichnen Sie die Logikschaltung für das Signal $e7$.

Lösung: **Bild 4**

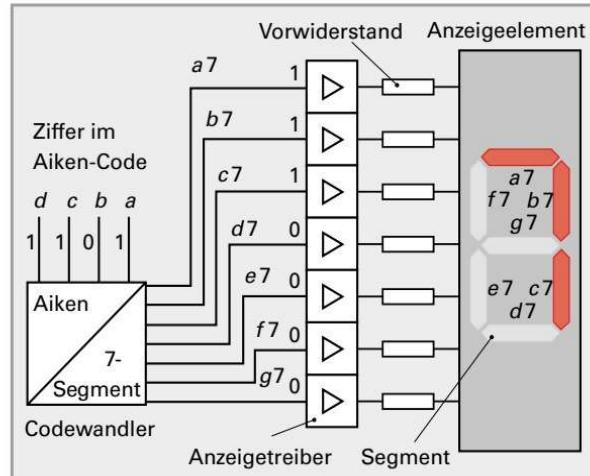


Bild 1: Anzeige der Ziffern des Aiken-Codes

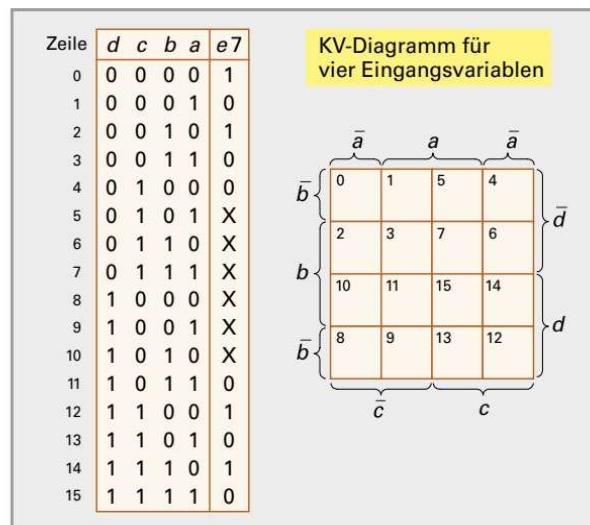


Bild 2: Signalzuordnung Wertetabelle – KV-Diagramm

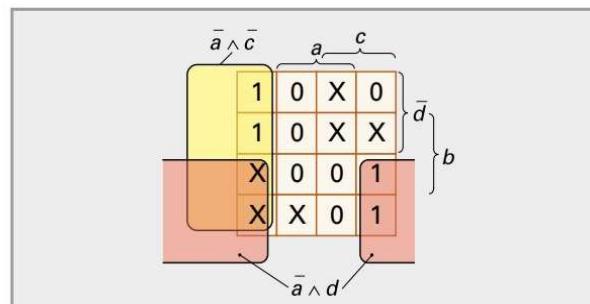


Bild 3: KV-Diagramm für das Segment $e7$

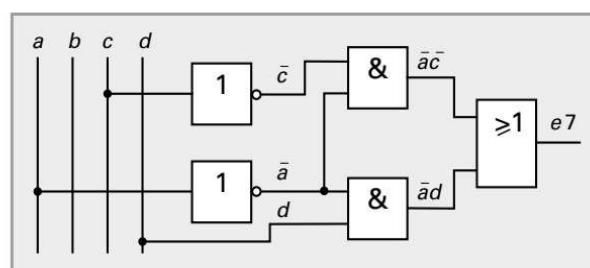


Bild 4: Logikschaltung für das Segment $e7$

Schaltungssimulation mit PSpice

Bevor Schaltungen aufgebaut werden, wird ihre Funktionsweise meist mit einem Simulationsprogramm, z. B. PSpice, simuliert. Für die von vielen Hochschulservern herunterzuladende Version 9.1 mit der Installationsdatei 91pspstu.exe, z. B. bei www.electronics-lab.com, gibt es die meisten Beispielsimulationen kostenlos im Internet. Mit dem Programmmodul Schaltplaneditor wird zunächst die Schaltung gezeichnet, hier die der Tunnelbelüftung (siehe Seite 198 und Bild 1). Durch Klicken auf die Schaltfläche Bauelemente suchen (Bild 2) werden nacheinander die Elemente NAND und DigStim im Schaltplan platziert. Die Elemente DigStim sind die Signalquellen der Eingangssignale *a*, *b* und *c*. Über die Schaltfläche Verdrahten werden die Elemente verbunden. Durch einen Doppelklick mit der linken Maustaste auf die Verbindungsleitungen können diese über ein sich öffnendes Fenster mit Signalnamen versehen werden, z. B. mit *a*.

Übung 1: Schaltplan zeichnen

Zeichnen Sie den Schaltplan der Tunnelbelüftung mithilfe des Schaltplaneditors von PSpice.

Lösung: Bild 1

Ist die Schaltung gezeichnet, muss sie abgespeichert werden. Bei der Vergabe des Programmnamens darf kein Umlaut verwendet werden, z. B. ä oder ü, da amerikanische Software diese Zeichen nicht verarbeitet. Die Simulation würde verweigert werden.

Zum Einstellen der Signalquellen DigStim werden diese doppelt angeklickt. Es öffnet sich das Programmmodul Stimulus Editor (Bild 3). Über ein Fenster wird für jede Signalquelle die Periodendauer (Period) und die Impulsdauer (On time) in Sekunden eingegeben. Dabei hat, wie in der Wertetabelle von oben nach unten betrachtet, *a* einen doppelt so schnellen Wechsel der Werte 0 und 1 wie *b* und *b* einen doppelt so schnellen Wechsel der Werte 0 und 1 wie *c*.

Übung 2: Signalquellen konfigurieren

Stellen Sie die Signalquellen DigStim ein.

Lösung: Bild 3

Um ein Zeitablaufdiagramm der Signale zu erhalten, muss die Transienten-Analyse aktiviert werden. Dazu wird die Schaltfläche Analyseart wählen (Bild 2) und die Schaltfläche Transient... angeklickt. Es öffnet sich das Fenster (Bild 4). Eingegeben werden müssen die Endzeit

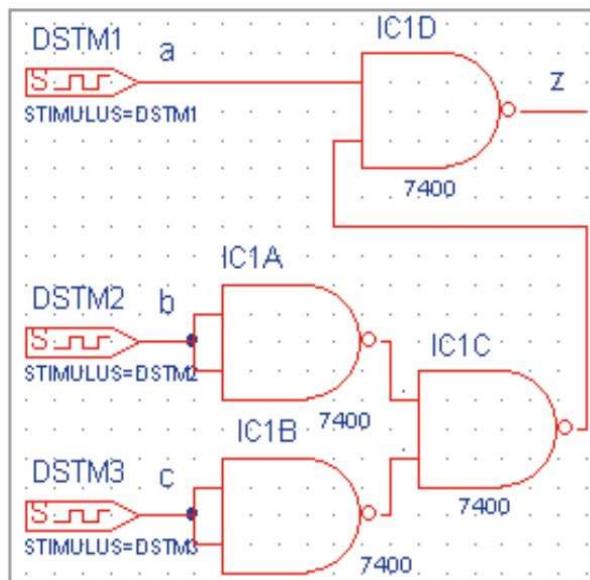


Bild 1: Schaltplan der Tunnelbelüftung

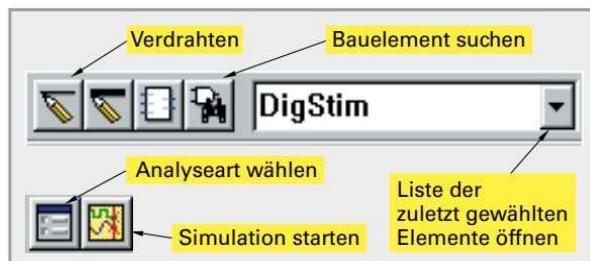


Bild 2: Schaltflächen der Menüleiste

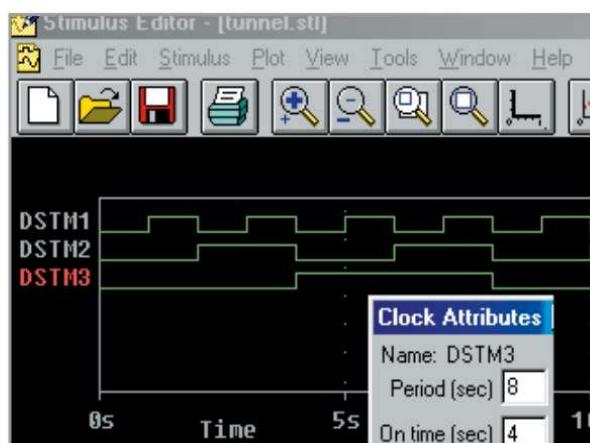


Bild 3: Stimulus Editor

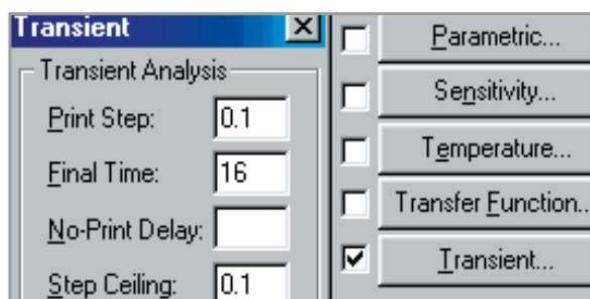


Bild 4: Transienten-Analyse einstellen

der Simulation (Final Time), die Zeichengenauigkeit und Rechengenauigkeit (Print Step und Step Ceiling). Durch Wahl der Final Time wird festgelegt, wie viele Signalzyklen dargestellt werden. Mit der Eingabe 16 s werden 2 Zyklen simuliert.

Das Programmmodul PSpice wird durch Mausklick auf die Schaltfläche Simulation starten aktiviert (Bild 2, vorhergehende Seite). Wurde die Rechengenauigkeit sehr hoch eingestellt, was bei Digitalschaltungen nicht notwendig ist, dauert der Rechenvorgang ein paar Sekunden. Danach öffnet sich automatisch das Programmmodul Probe (Bild 1), jedoch noch ohne einen der Signalverläufe. Durch Klicken auf die Schaltfläche Kurve hinzufügen (Add trace, Bild 1) öffnet sich ein Fenster zur Auswahl der Signale (Bild 2). Die Signale werden unter Simulation Output – Variables (Simulations-Ausgangsvariablen) in der Reihenfolge angeklickt, wie sie grafisch dargestellt werden sollen. Mit jeder Signalauswahl wird die Signalbezeichnung in die Zeile Trace Expression (Kurvenausdruck) aufgenommen (Bild 2). Durch Klicken auf die Schaltfläche OK wird das Zeitablaufdiagramm der Signale dargestellt.

Übung 1: Zeitablaufdiagramm zeichnen

Stellen Sie das Zeitablaufdiagramm der Signale a , b , c und z dar.

Lösung: Bild 1

Durch Mausklick auf die Schaltfläche Fadenkreuz (Bild 1) erscheint eine Signalerfassungslinie, welche mit der Maus horizontal hin und her bewegt werden kann. Die so erfassten Signalwerte werden neben den Signalbezeichnungen angezeigt (Bild 1). Man stellt fest, dass die Werte im dargestellten Diagramm der Wertetabelle entsprechen. Damit ist sichergestellt, dass die entwickelte Schaltung die angestrebte Schaltfunktion erfüllt.

6.1.7 Digitalschaltungen mit speicherndem Verhalten

Enthalten Digitalschaltungen Signalrückkopplungen von einem Ausgang auf einen Eingang, besitzen sie Speicherfähigkeit (Bild 3). Solche Schaltungen nennt man auch sequentielle Schaltungen, während Schaltungen ohne Speicherverhalten kombinatorische Schaltungen genannt werden. In der Schaltung Bild 3 kann für $a = 0$ und $b = 1$ das Ausgangssignal z sowohl den Wert 0 als auch den Wert 1 führen.

Welchen Wert das Signal z führt, hängt davon ab, welche Werte die Signale a und b hatten, bevor $a = 0$ und $b = 1$ wurden.

Kombinatorische Schaltung = Schaltung ohne speicherndes Verhalten.
Sequentielle Schaltung = Schaltung mit speicherndem Verhalten.

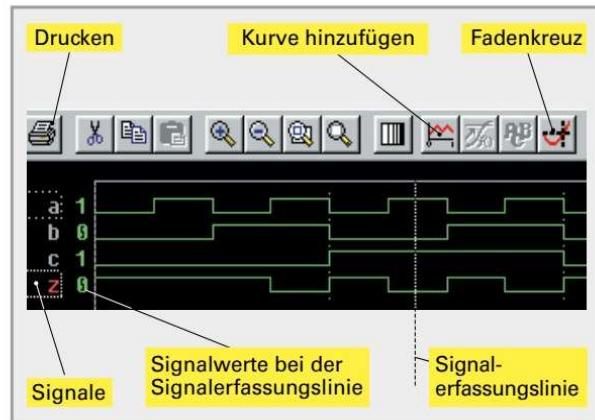


Bild 1: Kurvenausgabe

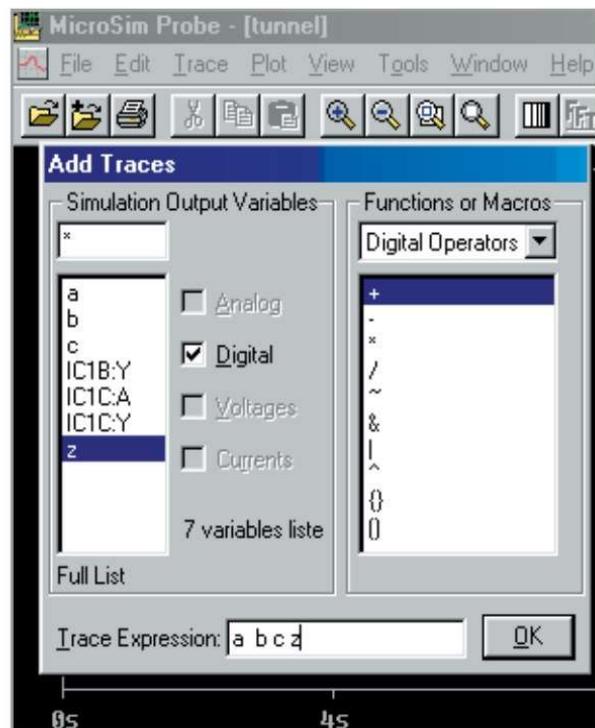


Bild 2: Kurvenauswahl

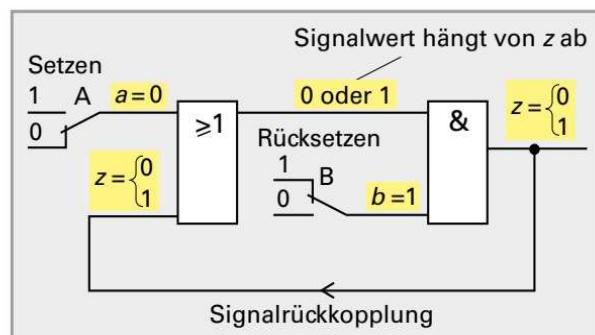


Bild 3: Schaltung mit speicherndem Verhalten

Haben a und b den Wert 1, wird z auf den Wert 1 gesetzt, haben a und b den Wert 0, wird z auf den Wert 0 zurückgesetzt. Für $a = 0$ und $b = 1$ wird der entsprechende Wert für z gespeichert. Einen solchen 1-Bit-Speicher nennt man auch Kippelement, Kippglied oder bistabile Kippschaltung.

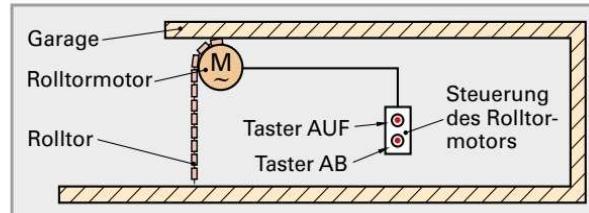


Bild 1: Rolltorsteuerung

Durch Betätigen des Tasters AUF wird das Rolltor einer Garage geöffnet (**Bild 1**). Durch Betätigen des Tasters AB wird es wieder geschlossen. Im betätigten Zustand der Taster, kann sich das Tor öffnen oder schließen.

Die notwendige Schaltung besteht aus einem RS-Kippelement ($RS = \text{reset, set bzw. Rücksetzen, Setzen}$). Dessen Ausgang q wird mit $s = 1$ gesetzt und $r = 1$ zurückgesetzt (**Bild 2**). Der Signalzustand $r = s = 1$ ist verboten. Dieser Zustand wird vermieden, indem man eine zusätzliche Ansteuerlogik vor den Eingängen r und s des RS-Kippelements anbringt. Dabei wird zwischen Setzdominanz (von lat. dominare = vorherrschen) und Rücksetzdominanz unterschieden (**Bild 3**).

Für $r = s = 1$ hat der Ausgang des setzdominannten RS-Kippelements den Wert 1, der des rücksetzdominannten RS-Kippelements den Wert 0.

Taktabhängige Kippelemente

Auf einer Prozessorplatine (**Bild 4**) befindet sich ein Taktgenerator mit einem Schwingquarz, welcher die Taktfrequenz 4 MHz ausgibt. Mithilfe von Kippelementen soll die Taktfrequenz auf 1 MHz vermindert werden.

Schaltungen aus Kippelementen, welche Taktfrequenzen vermindern, heißen Frequenzteiler.

Zum Aufbau von Frequenzteilern eignen sich JK-MS-Kippelemente (**Bild 5**). Sie besitzen einen Takteingang c (von $\text{clock} = \text{Takt}$), die Eingänge j und k und den Ausgang q sowie dessen Invertierung \bar{q} . Die Bezeichnung MS (von Master Slave = Meister Sklave) weist darauf hin, dass das JK-MS-Kippelement intern aus einem Master-Kippelement und einem Slave-Kippelement aufgebaut ist. Die Ausgänge q und \bar{q} solcher Kippelemente sind zweiflankengesteuert. Mit der Taktvorderflanke des Taktsignals c werden die Werte der Signale j und k erfasst, z. B. $j = 1$ und $k = 0$ (**Bild 5**), mit der Taktrückflanke des Taktsignals wird der Wert der Ausgangssignale q und \bar{q} ent-

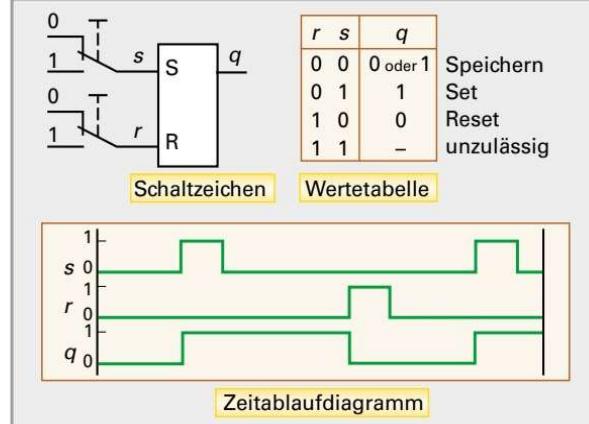


Bild 2: Darstellungsformen des RS-Kippelements

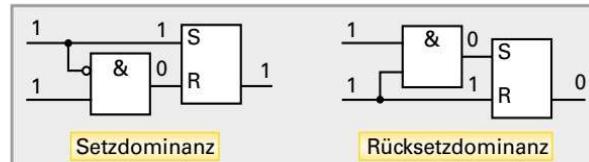


Bild 3: RS-Kippelemente ohne verbotenem Zustand

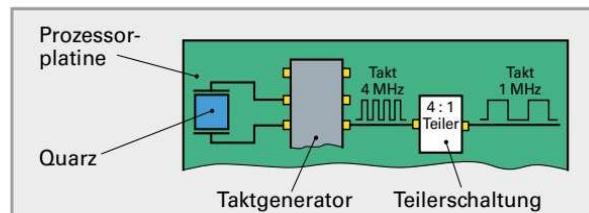


Bild 4: Frequenzteilung

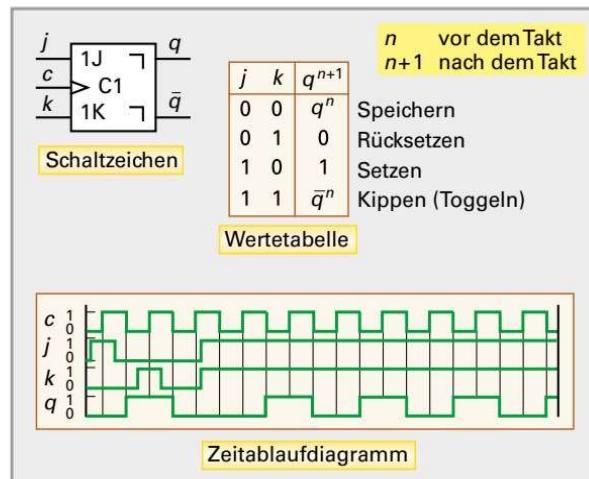


Bild 5: Darstellungsformen des JK-MS-Kippelements

sprechend der Wertetabelle verändert, z. B. $z = 1$ (Setzen). Die Wertetabelle zeigt, dass j dem Signal s und k dem Signal r des RS-Kippelements entsprechen. Z. B. ist für $j = k = 0$ das Ausgangssignal nach dem Takt q^{n+1} gleich dem vor dem Takt q^n . Der Signalwert bleibt also speichernd erhalten. Beim JK-MS-Kippelement gibt es keinen verbetenen Zustand. Für $j = k = 1$ ist das Ausgangssignal nach dem Takt q^{n+1} gleich der Invertierung des Ausgangssignals vor dem Takt q^n . Das Zeitablaufdiagramm zeigt, dass das Kippelement für $j = k = 1$ mit jedem Taktimpuls kippt (Bild 5, vorhergehende Seite). Dieses Verhalten nennt man Kippen oder Toggeln.

Im Kippbetrieb halbiert das JK-MS-Kippelement die Taktfrequenz.

Beispiel 1: Frequenzteiler entwerfen

Entwerfen Sie einen Frequenzteiler, der die Taktfrequenz eines Taktgenerators viertelt.

Lösung: Bild 1

Die Signalwerte eines 4-Bit-Datenwortes sollen gleichzeitig mit der Vorderflanke eines Taktimpulses zur Anzeige gebracht werden (**Bild 2**). Die Ausgabeeinheit besteht aus vier D-Kippelementen (D von delay = Verzögerung). Diese übergeben mit der Taktvorderflanke des Taktimpulses den Signalwert von d an den Ausgang q (**Bild 3**).

Die Kippelementarten RS, JK und D gibt es in verschiedenen Taktsteuerungsarten. Diese werden dem Schaltzeichen entnommen (**Tabelle 1**).

Zähler (counter)

Bei einer Tablettenabfüllanlage werden immer 10 Tabletten je Röhrchen abgezählt (**Bild 4**). Mit dem Wert 0 am Setzeingang wird die an den Eingängen e_1 bis e_4 anliegende Zahl 10 an die Ausgänge q_1 bis q_4 übertragen. Unterbrechen die in das Röhrchen fallenden Tabletten die Lichtschranke, entstehen positive Zählimpulse. Mit jeder steigenden Impulsflanke des Signals c wird der Zählerstand an den Ausgängen dekrementiert (um den Binärwert 1 vermindert). Sind 10 Tabletten im Röhrchen, ist der Ausgangszählerstand 0000. Es entsteht am Abwärtsübertrag $\bar{ü}$ ein kurzer negativer Impuls. Dieser wird dazu verwendet, über das Signal *set* den Zählerstand 10 wieder neu zu laden und die Tablettenzufuhr zu unterbrechen.

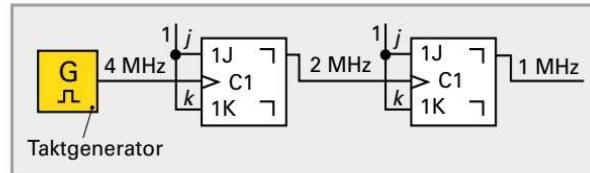


Bild 1: Frequenzteilung 4 : 1

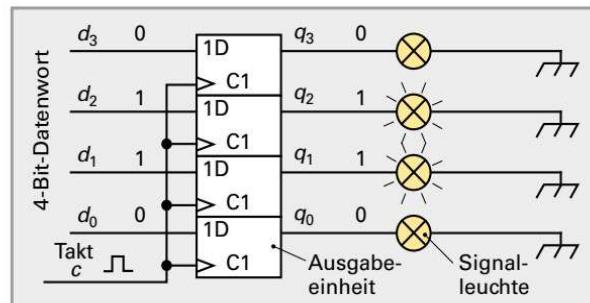


Bild 2: Ausgabe eines Datenwortes

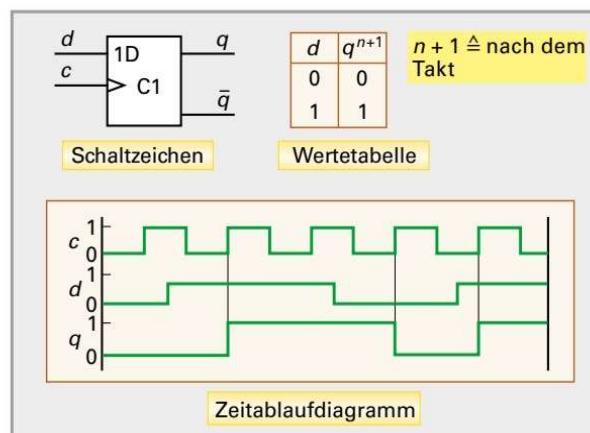


Bild 3: Darstellungsformen des D-Kippelements

Tabelle 1: Taktsteuerungsarten				
Art	Vorder-flanke	Rück-flanke	Zwei Flan-ken (MS)	Taktzu-stand
Sym-bolik				

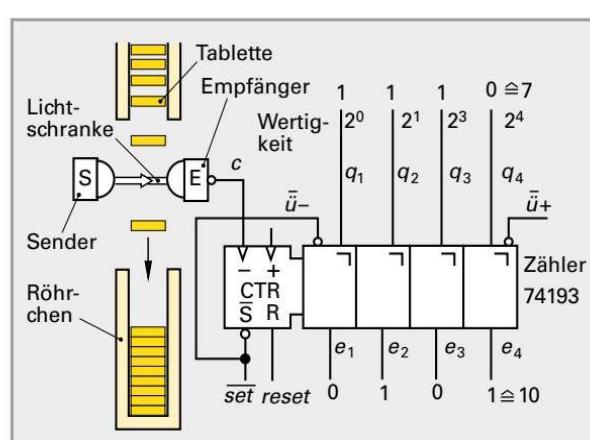


Bild 4: Tablettenabfüllanlage

Der Zähler ist ein IC, in welchem intern der Takt c alle Ausgangskippelemente synchron (gleichzeitig) ansteuert. Aufwärtszählen und Abwärtszählen wird bei einigen IC über einen Takteingang durchgeführt. Ein zusätzlicher Steuereingang legt dann die Zählrichtung fest. Neben Binärzählern gibt es auch Dezimalzähler.

Bei Dezimalzählern ist der höchste Zählerstand 9.

Schieberegister

Mithilfe von zwei 4-Bit-Schieberegistern soll das Binärwort 1111 seriell übertragen werden (**Bild 1**). Zuvor wird das Binärwort in das erste Schieberegister parallel eingelesen. Zum Schaltungsaufbau stehen zwei IC 74194 zur Verfügung (**Bild 2**). Bei ihnen wird die Betriebsart über die Wertekombination der Steuersignale s_1 und s_0 eingestellt. Die Betriebsart wird mit der steigenden Flanke des Taktsignals c wirksam. Bei der Betriebsart Rechtsschieben wird der Wert von d_{sr} (von data shift right = Daten schieben rechts) nach q_1 eingelesen. Die Werte von q_2 bis q_4 werden durch die Werte von q_1 bis q_3 ersetzt. Das Signal dsl (von data shift left = Daten schieben links) ist entsprechend beim Linksschieben aktiviert.

Beispiel 1: Schaltplan Datenübertragung entwickeln

Entwickeln Sie die Schaltung zur seriellen Datenübertragung des Binärwertes 1111.

Lösung: **Bild 3**

Der Schaltplaneditor von PSpice besitzt die Elemente HI = high = 1 und LO = low = 0, um Eingänge mit Signalwerten zu belegen. Damit die Schaltung simuliert werden kann, muss die Schaltfläche Analyseart wählen angeklickt werden und im

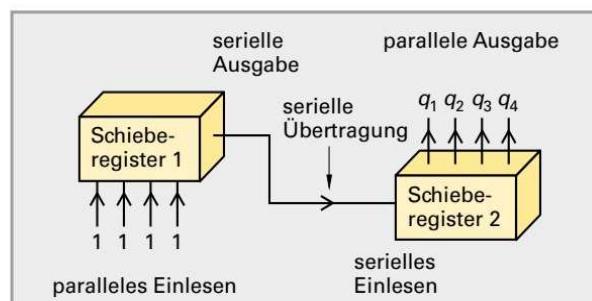


Bild 1: Serielle Datenübertragung mit Schieberegistern

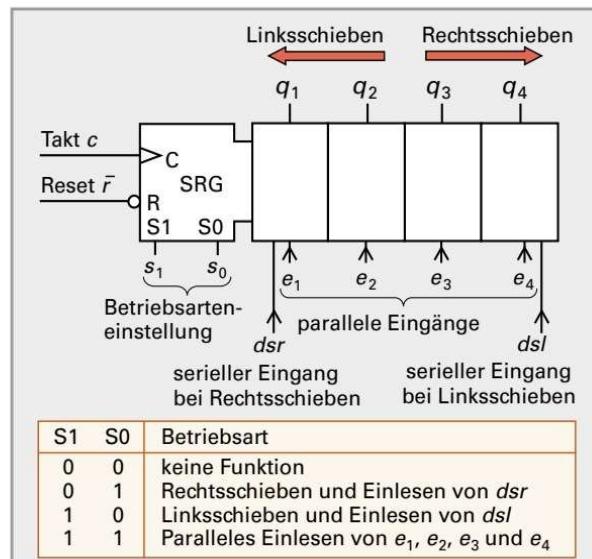


Bild 2: Schieberegister-IC 74194

Fenster Digital Setup die Flipflop Initialization (Kippelement-Einstellung) auf All 0 gestellt werden. Dadurch haben alle acht Schieberegisterausgänge zum Simulationsbeginn den Signalwert 0. Durch Doppelklick auf die Leitungen werden die Signalnamen eingegeben. Zwei verschiedene Leitungen dürfen nicht denselben Namen erhalten. Leitungsnamen dürfen keine Leerzeichen oder Sonderzeichen enthalten.

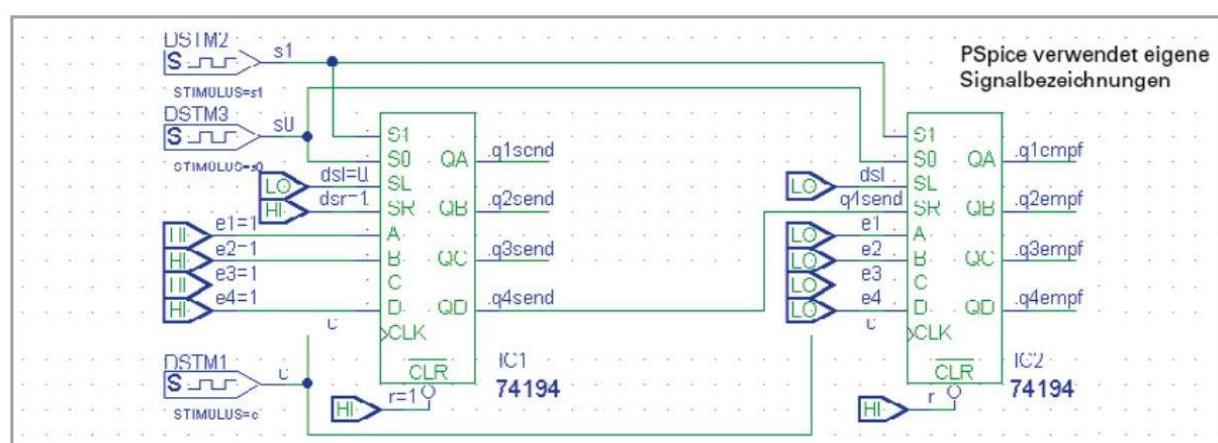


Bild 3: Schaltplan für die serielle Datenübertragung

Übung 1: Zeitablaufdiagramm ausgeben

Stellen Sie die Signalquellen DigStim ein und simulieren Sie zwei Signalzyklen der seriellen Datenübertragung.

Lösung: Bild 1

6.1.8 Tristate-Schaltelemente

Zwei Speicherbauelemente liegen an einem Adressbus an, welcher an den beiden Adressleitern A1 und A0 die Adresse $11 \hat{=} 3$ führt (Bild 2). Unter dieser Adresse ist im ersten Speicherbauelement das Datenwort 1010 und im zweiten Speicherbauelement das Datenwort 0010 abgelegt. Würden beide Speicherbauelemente gleichzeitig auf den Datenbus zugreifen, würde auf den Datenleiter D3 zugleich der Signalwert 1, z. B. 3,3 V wie auch der Signalwert 0, d. h. 0 V angelegt werden, was einen elektrischen Kurzschluss bedeuten würde. Deshalb darf immer nur ein Speicherbauelement auf den Datenbus zugreifen. Die Freigabe erfolgt über den Adressleiter A2 der zum Speicheranschluss CS (von chip select = Bauelement auswählen) führt (Bild 2). Die Datenausgänge aller nicht ausgewählten Speicherbauelemente werden hochohmig geschaltet. Dies geschieht über Tristate-Elemente (tristate = drei Zustände), welche sich an den Datenausgängen befinden (Tabelle 1).

Tristate-Elemente nehmen am Ausgang die Signalwerte 0, 1 oder hochohmig an.

Das Signal \overline{cs} eines Speicherbauelements wird intern den Steueranschlüssen EN (von to enable = freigegeben) der Tristate-Elemente an den Datenausgängen zugeführt.

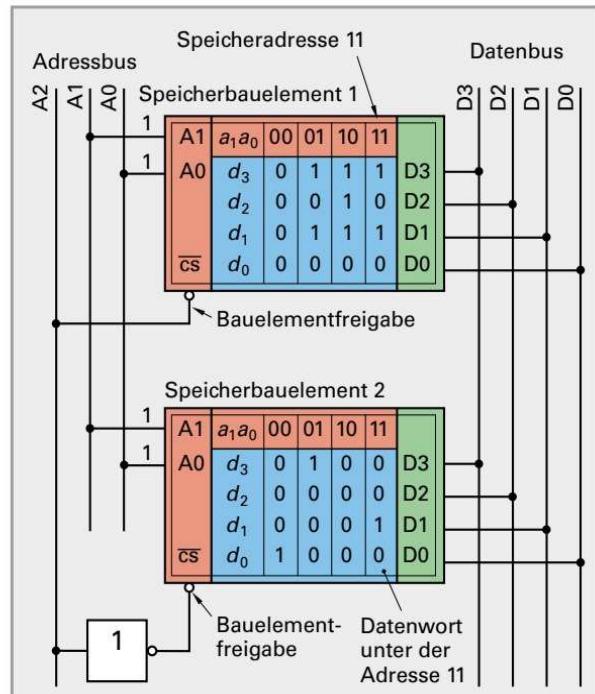


Bild 2: Prinzip der Speicheradressierung

Tabelle 1: Tristate-Elemente

Art	Tristate-Treiber mit positiver Ansteuerung (high-aktiv)	Tristate-Treiber mit negierter Ansteuerung (low-aktiv)	Tristate-Inverter mit positiver Ansteuerung (high-aktiv)																																				
Schaltzeichen																																							
Wertetabelle	<table border="1"> <thead> <tr> <th>en</th> <th>a</th> <th>z</th> </tr> </thead> <tbody> <tr> <td>0 X</td> <td>hochohmig</td> <td></td> </tr> <tr> <td>1 0</td> <td>0</td> <td></td> </tr> <tr> <td>1 1</td> <td>1</td> <td>$= a$</td> </tr> </tbody> </table>	en	a	z	0 X	hochohmig		1 0	0		1 1	1	$= a$	<table border="1"> <thead> <tr> <th>en</th> <th>a</th> <th>z</th> </tr> </thead> <tbody> <tr> <td>0 0</td> <td>0</td> <td></td> </tr> <tr> <td>0 1</td> <td>1</td> <td>$= a$</td> </tr> <tr> <td>1 X</td> <td>hochohmig</td> <td></td> </tr> </tbody> </table>	en	a	z	0 0	0		0 1	1	$= a$	1 X	hochohmig		<table border="1"> <thead> <tr> <th>en</th> <th>a</th> <th>z</th> </tr> </thead> <tbody> <tr> <td>0 X</td> <td>hochohmig</td> <td></td> </tr> <tr> <td>1 0</td> <td>1</td> <td></td> </tr> <tr> <td>1 1</td> <td>0</td> <td>$= \bar{a}$</td> </tr> </tbody> </table>	en	a	z	0 X	hochohmig		1 0	1		1 1	0	$= \bar{a}$
en	a	z																																					
0 X	hochohmig																																						
1 0	0																																						
1 1	1	$= a$																																					
en	a	z																																					
0 0	0																																						
0 1	1	$= a$																																					
1 X	hochohmig																																						
en	a	z																																					
0 X	hochohmig																																						
1 0	1																																						
1 1	0	$= \bar{a}$																																					

X = 0 oder 1 (Freiheitsgrad)

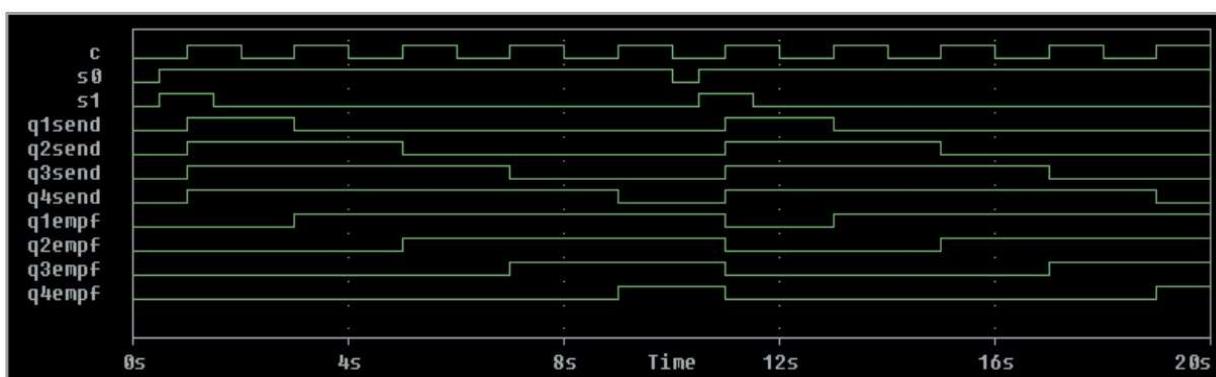


Bild 1: Zeitablaufdiagramm der seriellen Datenübertragung mit Schieberegistern

6.1.9 Multiplexer, Demultiplexer

Die Signale von vier Datenkanälen sollen über eine einzige Leitung übertragen werden und am Leitungsende wieder auf vier Datenkanäle verteilt werden (**Bild 1**). Die Übertragungsleitung steht dazu jedem Datenkanal zyklisch für eine kurze Zeitspanne zur Verfügung. Diese Mehrfachausnutzung der Übertragungsleitung nennt man Zeitmultiplex.

Beim Zeitmultiplexverfahren werden die Daten mehrerer Kanäle zeitlich versetzt auf einer Leitung übertragen.

Das Abtasten der Kanäle geschieht durch einen Multiplexer, das Verteilen auf die Kanäle am Übertragungsende durch einen Demultiplexer. Beide müssen im Gleichtakt auf die Kanäle zugreifen. Diese Synchronisierung erfolgt durch Adressdecoder. Sie erzeugen nacheinander für jeden Datenkanal je einen Freigabeimpuls, z.B. k_1, k_2, k_3 und k_4 .

Beispiel 1: Wertetabelle eines Adressdecoders erstellen

Entwickeln Sie die Wertetabelle eines Adressdecoders für vier Datenkanäle.

Lösung: Bild 2

Die Freigabeimpulse eines Adressdecoders geben nacheinander die vier UND-Tore des Multiplexers frei (**Bild 3**). Ein nachgeschaltetes ODER-Element leitet das Signal des freigegebenen Kanals auf die Übertragungsleitung. Der Demultiplexer besteht aus Torschaltungen, die gleichermaßen durch einen Adressdecoder zyklisch freigegeben werden (**Bild 3**). Beide Adressdecoder werden über dieselben Adresssignale a und b adressiert.

Beispiel 2: Adresseingänge berechnen

Wie viele Adresseingänge benötigen die Adressdecoder beim Zeitmultiplex von 16 Datenkanälen?

Lösung:

Da $2^4 = 16$, benötigt man **4 Adresseingänge**.

Ein Multiplexer für 16 Datenkanäle besteht aus einem 1-aus-16-Adressdecoder und einem 16-Kanal-Datenselektor (von to select = auswählen, **Bild 4**). Das Ausgangssignal \bar{q} des Multiplexers wird über einen Tristate-Inverter gesteuert. Der Steuereingang Strobe (Blitz) dient zur zusätzlichen Synchronisierung des Multiplexers mit einem Demultiplexer.

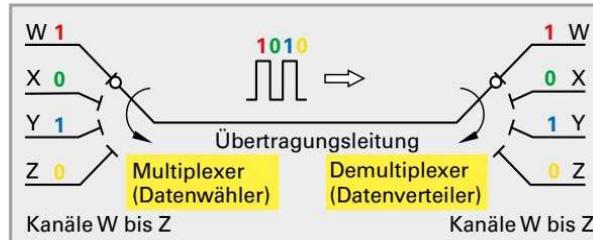


Bild 1: Zeitmultiplex

a	b	k_1	k_2	k_3	k_4
		0 0	1 0	0 0	0 0
1-aus-4- Adress- decoder		0 1	0 1	0 0	0 0
		1 0	0 0	0 1	0 0
		1 1	0 0	0 0	1 0
					0 1

Bild 2: 1-aus-4-Adressdecoder

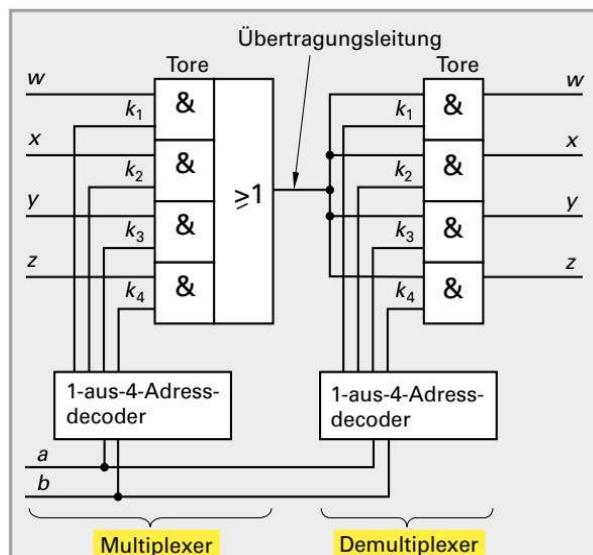


Bild 3: Prinzipschaltung der Datenübertragung

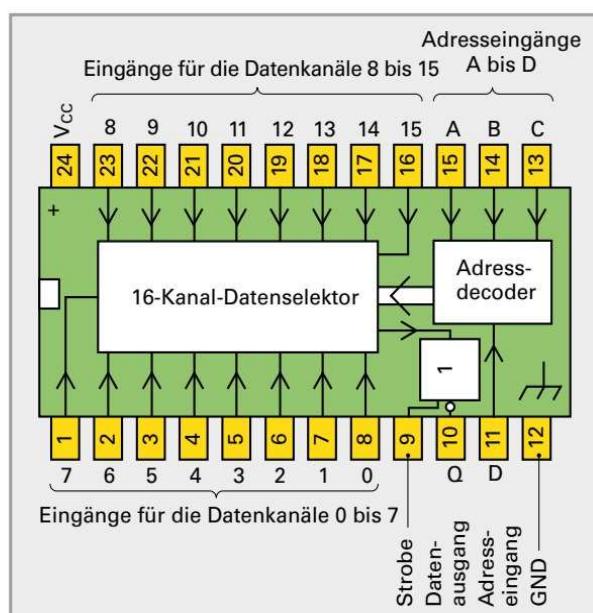


Bild 4: 1-aus-16-Multiplexer