Fachhochschule Karlsruhe
Hochschule für Technik
Fachbereich Nachrichtentechnik

Skript für Digitaltechnik I
(N1)

Stand: 18.04.1997

Prof. Dr.-Ing. J. Stöckle
Geb. E, Raum 307, Tel. (0721) 925-1574

Inhaltsverzeichnis

Kapitel	Bezeichnung	Seite
0	Einführung und Überblick	1
0.1	Analog und Digital; Abgrenzung und Vergleich	1
0.2	Was ist ein Bit ?	1
0.3	Anwendungen in der Digitaltechnik	2
1	Zahlensysteme	3
1.1	Polyadische Zahlensysteme	4
1.2	Zahlenumwandlung	4
1.3	Arithmetik im Dualsystem	6
1.4	Darstellung negativer Zahlen	7
1.4.1	Einser-Komplement	8
1.4.2	Zweier-Komplement	8
2	Kodierung	10
2.1	Numerische Codes	12
2.1.1	Wortcodes	12
2.1.1.1	Dualcode	12
2.1.1.2	Gray-Code	13
2.1.2	Zifferncodes	14
2.1.2.1	8-4-2-1-Code	14
2.1.2.2	BCD-Zählcode	15
2.1.2.3	Aiken-Code	15
2.1.2.4	1-aus-10-Code	16
2.1.2.5	Stiebitz- (3-Excess-) Code	16
2.1.2.6	Libow-Craig-Code	17
2.1.2.7	Ziffernsicherungscode Nr.3	17
2.2	Alphanumerische Codes	18
2.2.1	ASCII-Code	18
2.2.2	Fernschreibcode Nr.2 (Boudot-Code)	19
2.3	Codesicherung	19
2.3.1	Fehlererkennungs-Codes	20
2.3.1.1	Parity-Check (Paritätskontrolle)	20
2.3.1.2	m-aus-n-Codes	21
2.3.2	Fehlerkorrigierende Codes	21
2.3.2.1	Verfahren der Blocksicherung	21
2.3.2.2	Hamming-Codes	21

Kapitel	Bezeichnung	Seite
3	Schaltalgebra	24
3.1	Schaltvariable und Schaltfunktion	24
3.2	Rechenregeln	24
3.2.1	Rechenregeln für Konstanten	25
3.2.2	Rechenregeln für Variablen und Konstanten	25
3.2.3	Verknüpfungen und Schaltsymbole	26
3.2.4	Vorrangregeln (DIN 66000)	31
3.2.5	Normalformen	32
3.2.5.1	Disjunktive Normalform	33
3.2.5.2	Konjunktive Normalform	34
4	Logisch-physikalische Zusammenhänge	35
4.1	Verwendung von Logikpegeln	35
4.2	Zuordnung von logischen Zuständen zu Logikpegeln	36
4.3	Verwendung von Polaritätsindikatoren	37
5	Minimieren von Schaltfunktionen mit dem KV-Diagramm	38
5.1	KV-Diagramme für zwei Schaltvorgänge	38
5.2	KV-Diagramme für mehr als zwei Variablen	39
5.3	Redundanz	39
5.4	Konjunktive Minimalform	40
5.5	entfällt	
5.6	Vorgehen bei der Minimierung	40
6	Elektronische Realisierung der Schaltelemente	41
6.1	Relais-Logik	41
6.2	Dioden-Logik	42
6.3	Transistor-Logik	43
6.3.1	Resistor-Transistor-Logik (RTL)	43
6.3.2	Dioden-Transistor-Logik (DTL)	43
6.3.3	Transistor-Transistor-Logik (TTL)	44
6.3.4	Integrated Injection Logik (I^2L)	44
6.3.5	Emitter-Coupled-Logik (ECL)	45
6.3.6	NMOS-Logik	45
6.3.7	CMOS-Logik	46
6.4	Ausgangsschaltungen	47
6.5	Eingangsschaltungen	51
6.6	Schaltkreisfamilien	51
6.7	Zusammenschalten verschiedener Familien	52
6.8	EGB-Schutzmaßnahmen	53

Kapitel	Bezeichnung	Seite
7	Analyse von Schaltnetzen	54
8	Synthese von Schaltnetzen	58
8.1	Minimalformen	58
8.2	Häufig benutzte Schaltnetze	60
8.2.1	Vergleicher	60
8.2.2	Code-Umsetzer	63
8.2.3	Multiplexer	65
8.2.4	Demultiplexer	67
8.2.5	Rechenschaltungen	68
8.3	Normalformen	69
8.3.1	Realisierung mit MUX	69
8.3.2	Realisierung mit ROM	72
9	Schaltwerke	74
9.1	Einführung	74
9.1.1	Schaltnetze und Schaltwerke	74
9.1.2	Zustandsdiagramme und Zustandsfolgetabelle	74
9.1.3	Struktur von Schaltwerken	77
9.1.4	Asynchroner und synchroner Betrieb	78
9.2	Flipflop's (FF's)	79
9.2.1	RS-Flipflop	79
9.2.2	FF's mit Taktsteuerung	81
9.2.2.1	Taktzustandsgesteuerte Einspeicher-FF	81
9.2.2.2	Taktzustandsgesteuerte Zweispeicher-FF	83
9.2.2.3	Taktflankengesteuerte FF's	85
9.2.3	Konvertierung von FF's	87
9.2.4	Überblick	88
9.3	Zähler	89
9.3.1	Synchrone Zähler	89
9.3.2	Asynchrone Zähler	90
9.4	Register	91
9.4.1	Auffangregister (Latch)	91
9.4.2	Schieberegister	92
9.4.3	Schieberegister mit Rückführung	93
Anhang A	Index-Verzeichnis	96

Digitaltechnik

0 Einführung und Überblick

0.1 Analog und Digital; Abgrenzung und Vergleich

	Duden	Technik
analog =	ähnlich, entsprechend	kontinuierlich, nicht abzählbar
digital =	mit Ziffern, mit Fingern	abzählbar

analoge Signale : Temperatur, Druck, Spannung, Strom, ...
digitale Signale : Einwohnerzahl, Kontostand, ...

Messung analoger Signale :

analoge Messung — Jeder Meßgröße wird ein prozentual gleicher Meßwert zugeordnet. Jede Zwischenstufe ist möglich. (20% Meßgröße entsprechen 20% Meßwert)

digitale Messung — Jede Meßgröße wird quantisiert und einer Quantisierungsstufe zugeordnet. (0-10% entsprechen der ersten Quantisierungsstufe, 11-20% entsprec hen der zweiten Quantisierungsstufe, u.s.w.)

Anzeige der Meßwerte:

analog : - Leuchtband („Katzenauge“)
- Zeigerinstrument
digital : - Ziffernanzeige (LCD-Display)

Analoge und digitale Verfahren:

Verfahren	analog	digital
Drehspulinstrument	x	
Digitalvoltmeter		x
Quecksilberthermometer	x	
Maßstab	x	
Tacho (PKW)	(x)	(x)
Kilometerstand (PKW)		x
Rechenschieber	x	
Taschenrechner		x

0.2 Was ist ein Bit ?

binär = zweiwertig, zwei Zustände (0 oder 1)

		Binärer Zustand
Schalter	ein	1
Schalter	aus	0
Antwort	richtig	1
Antwort	falsch	0
Mensch	männlich	1
Mensch	weiblich	0
Spannung	≥ 10V	1
Spannung	≤ 10V	0

1 Bit (Binary Digit) = ein binärer Zustand

Zusammenfassung mehrerer Bits :

Tetrade	=	4 Bit	Bit 3	Bit 2	Bit 1	Bit 0				
Byte	=	8 Bit	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	=	2 Tetraden	1	0	1	0	1	1	0	0
			MSB							LSB
			Most Significant Bit						Least Significant Bit	

Vorzüge der digitalen Darstellung :

- leichte Ablesbarkeit
- höhere Genauigkeit bei geringen Kosten (ab Genauigkeit 10^{-3})
- einfache und zuverlässige Speicherung digitaler Meßwerte

0.3 Anwendungen der Digitaltechnik

- PC
- Gameboy
- CD-Player
- Digital Audio Tape (DAT)
- Digitaler Rundfunk (ADR, DSR)
- Digitales Fernsehen (DF1)
- Fahrkartenautomat
- Ampelsteuerung
- Speicher-Programmierbare-Steuerung (SPS)
- Digitales Telefon (ISDN)
- Briefsortieranlage
- Computerized Numeric Control (CNC)

1 Zahlensysteme

Problem : Zählen, Aufzeichnen des Resultats, Rechnen

Abzählschreibweise : z.B. Erbsenzählen

°	l
°°	ll
°°°	lll
°°°°	llll
°°°°°	~~llll~~

= Bündelung (5er-Bündel), Stelle gleichgültig

Nachteile : - Nur für kleine Mengen !
- keine Rechnung möglich !

Römisches Zahlensystem :

2er- und 5er- Bündelung

I	VI	XI
II	VII	XII
III	VIII	....
IV	IX	
V	X	

V = 5, X = 10, L = 50, C = 100, D = 500, M = 1000

Stellung relativ zu anderen Symbolen wichtig !

Nachteil : Rechnung sehr umständlich

Zahlensysteme und Stellenschreibweise

$$z = \sum_{i=1}^{n} a_i w_i$$

a_i : Ziffer (Vielfaches des Stellenwertes)

w_i : Stellenwert

Beispiel :

Z = 2 Wochen	+	1 Tag	+	4 Stunden	= 214
$2 \cdot (24 \cdot 7)$	+	$1 \cdot 24$	+	$4 \cdot 1$	= 364 Std.
a_3 w_3		a_2 w_2		a_1 w_1	

Vorteil : Einfaches Rechnen, aber Stellenüberträge aufwendig, da unterschiedliche Anzahl von Ziffern pro Stelle.

Beispiel : s.o. a_1 = 0-23
a_2 = 0-6
a_3 = 0-51

1.1 Polyadische Zahlensysteme

$$z = \sum_{i=-m}^{n} a_i B_i$$

a_i : Ziffer (B Ziffern)
B_i: Basis des Zahlensystems

Zahlensystem	Basis	a_i	Bezeichnung
Dualsystem	2	0,1	B / $_2$
Oktalsystem	8	0,1,2,3,4,5,6,7	O / $_8$
Dezimalsystem	10	0,1,2,3,4,5,6,7,8,9	D / $_{10}$
Hexadezimalsystem	16	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F	H / $_{16}$

Beispiel :

1543 D = $1 \cdot 10^3 + 5 \cdot 10^2 + 4 \cdot 10^1 + 3 \cdot 10^0$

10011 B= $1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$

134A H = $1 \cdot 16^3 + 3 \cdot 16^2 + 4 \cdot 16^1 + 10 \cdot 16^0$

Wichtige Potenzen :

Potenz	Dezimal	Hexadezimal	Bezeichnung
2^0	1	1	
2^1	2	2	
2^2	4	4	
2^3	8	8	
2^4	16	10	
2^5	32	20	
2^6	64	40	
2^7	128	80	
2^8	256	100	
2^9	512	200	
2^{10}	1024	400	1k
2^{11}	2048	800	2k
2^{12}	4096	1000	4k
2^{13}	8192	2000	8k
2^{14}	16384	4000	16k
2^{15}	32768	8000	32k
2^{16}	65536	10000	64k
....	....	....	....
2^{20}	1048576	100000	1M

1.2 Zahlenumwandlung

$$z = \sum_{i=-m}^{n} a_i B^i = \underbrace{\sum_{i=0}^{n} a_i B^i}_{\text{ganzzahliger Anteil}} + \underbrace{\sum_{i=-m}^{-1} a_i B^i}_{\text{Nachkommastellen}}$$

a.) $B_1 = 10 \rightarrow B_2$ (hier: $B_2 = 2$)

1.) Aufteilen in ganzzahligen Anteil und Nachkommastellen

Bsp.: 190,24 = 190 + 0,24

2.) Fortlaufende Division des ganzzahligen Anteils mit B_2 bis Resultat = 0

Bsp.: 190 : 2 = 95 Rest 0 2^0
95 : 2 = 47 Rest 1 2^1
47 : 2 = 23 Rest 1 2^2
23 : 2 = 11 Rest 1 2^3
11 : 2 = 5 Rest 1 2^4
5 : 2 = 2 Rest 1 2^5
2 : 2 = 1 Rest 0 2^6
1 : 2 = 0 Rest 1 2^7
190 D = 1011 1110 B

3.) Fortlaufende Multiplikation der Nachkommastellen mit B_2 bis Rest = 0 oder die gewünschte Genauigkeit erreicht ist.

Bsp.: 0,24 * 2 = 0,48 + 0 2^{-1}
0,48 * 2 = 0,96 + 0 2^{-2}
0,96 * 2 = 0,92 + 1 2^{-3}
0,92 * 2 = 0,84 + 1 2^{-4}
0,84 * 2 = 0,68 + 1 2^{-5}
0,68 * 2 = 0,36 + 1 2^{-6}
0,36 * 2 = 0,72 + 0 2^{-7}
0,72 * 2 = 0,44 + 1 2^{-8}
0,24 D = 0,00111101 B

4.) Zusammenfassen der Ergebnisse aus 2.) und 3.)

Bsp.: 190,24 D = 1011 1110, 0011 1101 ... B

b.) $B_1 \rightarrow B_2 = 10$ (hier : $B_1 = 2$)

1.) Aufteilen in ganzzahligen Anteil und Nachkommastellen

Bsp.: 1011 1110,0011 1101 B = 1011 1110 B + 0,0011 1101 B

2.) Schrittweise ganzzahligen Anteil mit B_1 multiplizieren

Bsp.: 1 0 1 1 1 1 1 0

2 * 1+0 = 2
2 * 2 + 1 = 5
2 * 5 + 1 =11
2 * 11 + 1 =23
2 * 23 + 1 =47
2 * 47 + 1 =95
2 * 95 + 0 =190

3.) Schrittweise Nachkommastellen mit B_1 dividieren

Bsp.: 0, 0 0 1 1 1 1 0 1

1 / 2 = 0,5
0 + 0,5 / 2 = 0,25
1 + 0,25 / 2 = 0,625
1 + 0,625 / 2 = 0,81...
1 + 0,81... / 2 = 0,90...
1 + 0,90... / 2 = 0,95...
0 + 0,95... / 2 = 0,47...
0 + 0,47... / 2 = 0,23828125 D

4.) Zusammenfassen der Resultate aus 2.) und 3.)

Bsp.: 1011 1110,0011 1101 B = 190,23828125 D

$$F_r = \frac{190{,}23828125 - 190{,}24}{190{,}24} \cdot 100\% = -0{,}0009\%$$

Sonderfälle : $B_1^p = B_2, \quad p>1$

a.) $B_1 = 2, B_2 = 16, \rightarrow p = 4$

1011 1110,0011 1101 B

B E , 3 D H

b.) $B_1 = 2, B_2 = 8, \rightarrow p = 3$

010 111 110, 001 111 010 B

2 7 6 , 1 7 2 H

1.3 Arithmetik im Dualsystem

Addition :

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 10 = Übertrag (Carry-Bit)

Beispiel :

```
  0 1 0 1
 +1 0 1 0
 1 1        (carry)
1 0 0 0 1
```

```
  0 0 0 1,1
+ 1 0 1 0,1
    1 1 1   (carry)
1 1 0 0,0
```

Subtraktion :

0 - 0 = 0

0 - 1 = 1 = Borgen von höherer Stelle (Borrow-Bit)

1 - 0 = 1

1 - 1 = 0

Beispiel :

```
  1 1 0 0
- 0 1 0 1
    1 1 1   (borrow)
  0 1 1 1
```

```
  1 1 1 0,1 1
- 0 1 0 1,0 1
      1         (borrow)
1 0 0 1,1 0
```

Multiplikation : 0 * 0 = 0

0 * 1 = 0

1 * 0 = 0

1 * 1 = 1

Beispiel : 1 0 1 * 11

1 0 1

\+ 1 0 1

1 1 1 1

1 0,0 1 * 1 0,1

1 0 0 1

\+ 1 0 0 1

1 0 1,1 0 1

Division : Wie bei Dezimalsystem

Beispiel : 1 0 0 1 : 11 = 11

$_1$ 1$_1$1

0 0 1 1

1 1

0 0

1 1 0 1 : 1 0 1 = 1 0,1 0 0 1 1 0 0 ...

1 0$_1$1

0 0 1 1 0

1 0$_1$1

0 0 1 0 0 0

$_1$1$_1$0$_1$1

0 0 1 1 0

1 0$_1$1

0 0 1 0 0 0

....

1.4 Darstellung negativer Zahlen

Dezimal	Dual
0	0
1	1
2	10
3	11
4	100
...	...

Aber: -3 D = -11 B

Darstellung des Vorzeichens auf dem Rechner ?

Idee: Feste Stellenzahl (Anzahl Bits) + Vorzeichen-Bit

Vorzeichen-Bit = 0 ⇒ positiv

Vorzeichen-Bit = 1 ⇒ negativ

Bit 3 (VZ)	Bit 2	Bit1	Bit 0	Dezimal
0	0	0	0	+0
0	0	0	1	+1
0	0	1	0	+2
0	0	1	1	+3
...	...	...	...	...
0	1	1	1	+7
1	0	0	0	-0
1	0	0	1	-1
1	0	1	0	-2
...	...	...	...	...
1	1	1	1	-7

Problem: Rechnen schwierig !!

7 + (-7) = 0 0 1 1 1

$_1$1$_1$1$_1$1

1 0 1 1 0 ≠ 0 !!!

1.4.1 Einser-Komplement

Bit 3 (VZ)	Bit 2	Bit 1	Bit 0	Dezimal
0	0	0	0	+0
0	0	0	1	+1
...	...	...	...	...
0	1	1	1	+7
1	0	0	0	-7
1	0	0	1	-6
...	...	...	...	...
1	1	1	0	-1
1	1	1	1	-0

Bildung: 0 1 0 1 +5
1 0 1 0 -5
Jedes Bit wird invertiert

Beispiel: +7 + (-7) = 0

```
  0 1 1 1
+ 1 0 0 0
  1 1 1 1 = -0
```

Aber: Zwei mal die Null als Zahl vorhanden !!

Schreibweise : $K_1(x)$

1.4.2 Zweier-Komplement

Bit 3 (VZ)	Bit 2	Bit 1	Bit 0	Dezimal
0	0	0	0	0
0	0	0	1	+1
0	0	1	0	+2
...	...	...	...	...
0	1	1	1	+7
1	0	0	0	-8
1	0	0	1	-7
...	...	...	...	...
1	1	1	1	-1

Bildung:

```
  0 1 1 1      +7
  1 0 0 0      Einser-Komplement
+ 0 0 0 1
  0 0 0 1 = -7
```

Beispiel: +7 + (-7) = 0

```
  0 1 1 1   +7
  1 0 1 0 1 -7
1 0 0 0 0    0
```

1 = entfällt; Beschränkung auf 4 Stellen (4 Bit)

Schreibweise : $K_2(x)$

Definition: (Echtes) Komplement einer n-stelligen Zahl der Basis B

$K(x) = B^n - x$
x: = positive Zahl
B^n:= Anzahl der darstellbaren Zahlen

$\Rightarrow K(x) + x = B^n$

Vorteil: Subtraktionen lassen sich auf Additionen zurückführen
denn: $x + K(y) = x + B^n - y = x - y + B^n$
B^n: = Übertrag auf die (n+1).Stelle

Dezimalsystem : $B = 10$, $n = 3$, $B^n = 1000$

K(1)	= 1000 - 1	= 999
K(2)	= 1000 - 2	= 998
	....	
K(499)	= 1000 - 499	= 501
K(500)	= 1000 - 500	= 500

10^2	10^1	10^0	Dezimal
0	0	0	0
0	0	1	+1
...	...	...	...
4	9	9	+499
5	0	0	-500
5	0	1	-499
...	...	...	...
9	9	9	-1

+ 1	0 0 1
+(-1)	$_1 9_1 9_1 9$
0	1 0 0 0 = 0

Dualsystem : $B = 2$, $n = 4$, $B^n = 2^4 = 16$

K(1)	= 1 0 0 0 0	- 1	= 1 1 1 1
K(2)	= 1 0 0 0 0	- 1 0	= 1 1 1 0
K(3)	= 1 0 0 0 0	- 1 1	= 1 1 0 1
	.....		

$K(x) = 2^n - 1 - x + 1$ (-1 und +1 sind Erweiterungen der Gleichung)

$K_2(x)$ $K_1(x)$

$K_2(x) = K_1(x) + 1$

2. Codierung

Definition: Code (DIN 44 300)
Unter einem Code versteht man die eindeutige Zuordnung der Zeichen eines Zeichenvorrats zu denjenigen eines anderen Zeichenvorrats (Bildmenge)

Definition: Binärcode
Bei einem Binärcode besteht jedes Zeichen der Bildmenge aus einer Kombination der Binärzeichen „0“ und „1“

	Code	
	numerisch	**alphanumerisch**
darstellbare Ziffern	Ziffern, Zahlen	Ziffern, Zahlen, Buchstaben, Sonderzeichen

Beispiel: 8-4-2-1-Code (numerischer Code)

Codierung einer Ziffer: = Codewort (CW)

Bit 3 2 1 0	Ziffer
8 4 2 1	Wertigkeit
0 0 0 0	0
0 0 0 1	1
....	
1 0 0 1	9
1 0 1 0	nicht benutzt !
....	....
1 1 1 1	nicht benutzt !

Codierung mehrerer Ziffern:

	10^2	10^1	10^0
123 D =	0001	0010	0011

BCD-Code (Binary Coded Decimal)

Definition: Bewertbarer Code
Jede Stelle besitzt eine Wertigkeit (Stellenwert)
Beispiel: 8 - 4 - 2 - 1

Nicht bewertbarer Code
Anordnungscode

Definition: Gewicht G
Gewicht G eines Codewortes ist die Anzahl der mit „1“ belegten Stellen
Beispiel: 0 1 1 1 G = 3

Definition: Distanz D
Die Distanz D (Hamming-Distanz) zwischen zwei Codeworten ist die Anzahl der Binärstellen, in denen sich die Codeworte unterscheiden.
Beispiel: 0 0 0 0; 0 1 0 1 D = 2

Definition: Die Minimal-Distanz D_{min} und die Maximal-Distanz D_{max} eines Codes ist gleich die minimale bzw. Maximale Distanz bei Vergleich aller verwendeten Codeworte des Codes.

Beispiel:

8	4	2	1	D_{min}	D_{max}
0	0	0	0	1	3
0	0	0	1	1	3
0	0	1	0	1	3
0	0	1	1	1	3
0	1	0	0	1	3
0	1	0	1	1	3
0	1	1	0	1	4
0	1	1	1	1	4
1	0	0	0	1	4
1	0	0	1	1	4

Definition: Stetiger Code

Ein Code ist stetig, wenn die Distanz zwischen benachbarten Codewörtern konstant ist.

Beispiel: 8-4-2-1-Code

$0000 \rightarrow 0001$ D = 1
$0001 \rightarrow 0010$ D = 2
$0010 \rightarrow 0011$ D = 1
$0011 \rightarrow 0100$ D = 3
$\Rightarrow$ Code ist nicht stetig !!!

Definition: Vollständiger Code

Ein Code wird als vollständiger Code bezeichnet, wenn alle möglichen Kombinationen als Codeworte definiert werden.

Beispiel: 8-4-2-1-Code
16 mögliche Kombinationen, 10 Codeworte verwendet
$\Rightarrow$ Code nicht vollständig !!

Definition: Redundanz

Der Code besitzt mehr Kombinationsmöglichkeiten, als für die Codierung genutzt werden.

$R = ld\ N_{pot} - ld\ N$

N_{pot} := alle Kombinationen
N := benutzte Kombinationen
ld := Logarithmus dualis

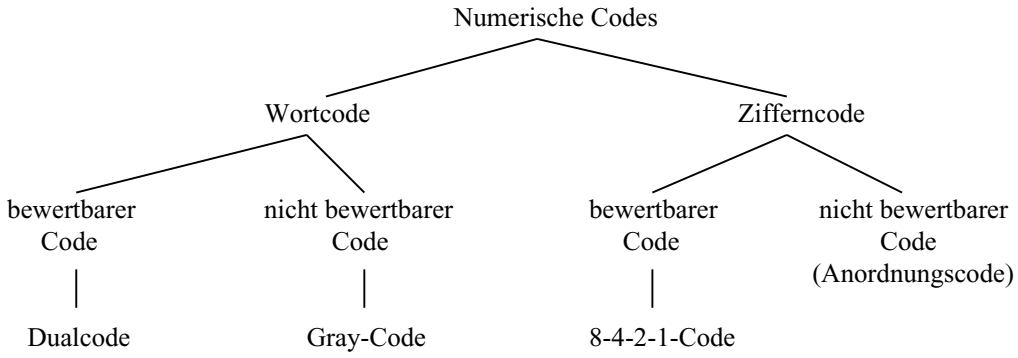
$$x = 2^{ldx} = 10^{\log x} = e^{\ln x}$$
$$ld\ x \cdot \log 2 = \log x$$
$$ld\ x = \frac{\log x}{\log 2} = 3{,}32 \log x$$

Beispiel: 8-4-2-1-Code
$N_{pot} = 16$
N = 10
$R = ld\ 16 - ld\ 10 = ld\ 2^4 - ld\ 10 = 4 - 3{,}32 = 0{,}68$

Interpretation: R = 0,68 Bit zu viel! Zur Codierung der Ziffern genügen 3,32 Bit (theoretisch)

Nutzung der Redundanz: Fehlererkennung, Fehlerkorrektur

2.1 Numerische Codes



Wortcode: Codierung einer Zahl als ganzes, nicht ziffernweise
Zifferncode: getrennte Codierung jeder Ziffer einer Zahl.

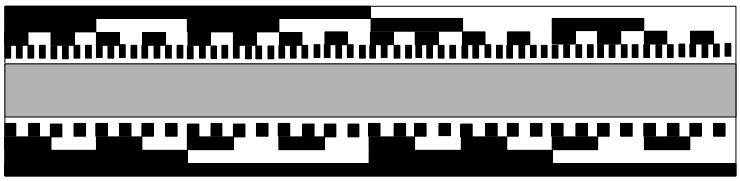
2.1.1 Wortcodes

2.1.1.1 Dualcode

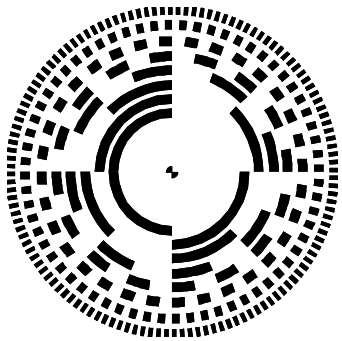
	Bit 3	Bit 2	Bit 1	Bit 0	Dezimal
	0	0	0	0	0
	0	0	0	1	1
	0	0	1	0	2
	0	0	1	1	3
	...	...	...	...	...
1	0	0	0	1	17

Eigenschaften: - aufwendige Ein/Ausgabe (Konvertierung von Dezimal ↔ Dual)
- Rechenoperationen einfach → schnell
- bewertbar, nicht stetig, keine Redundanz

Anwendung:

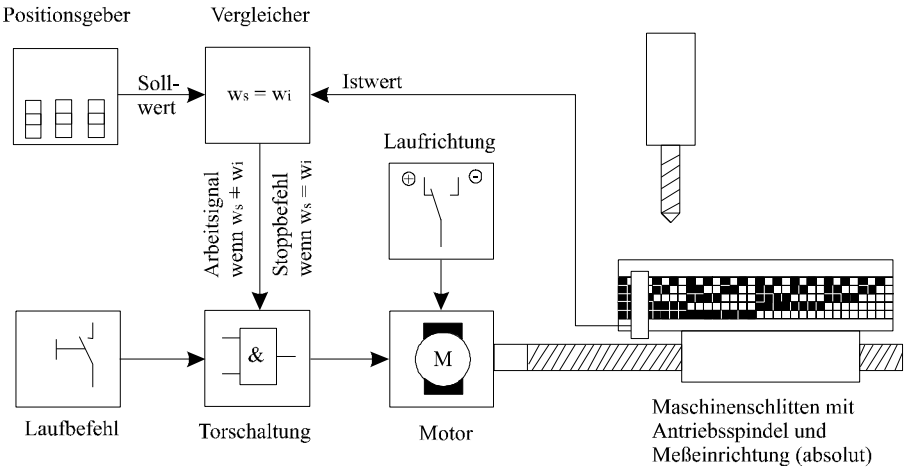


a.) Code-Lineal



b.) Code-Scheibe

Beispiel:



Schema einer Vorschubsteuerung mit drehzahlgeregeltem Motor und absoluter Wegmeßeinrichtung

2.1.1.2 Gray-Code

Umwandlung: Dualcode → Gray-Code : 1 in nächsthöherer Stelle des Dualcodes → vorhandene Gray-Codeworte in umgekehrter Reihenfolge rechts neben die 1 setzen.

Dezimal	Dual	Gray-Code
0	0000	0
1	0001	1
2	0010	1 1
3	0011	1 0
4	0100	1 1 0
5	0101	1 1 1
6	0110	1 0 1
7	0111	1 0 0
8	1000	1 1 0 0
9	1001	1 1 0 1
...	...	...

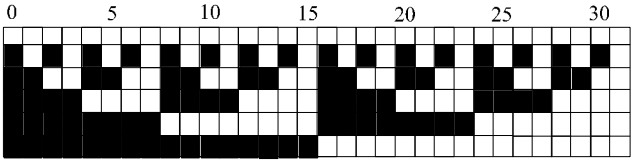
nicht erweiterbar !!

denn: 1 D = 0 1 (Gray) ; W = 1
3 D = 1 0 (Gray) ; W = 3
Aber: 2 D = 1 1 (Gray) , W = 1 + 3 = 4 ! $4 \neq 2$ Widerspruch!!!

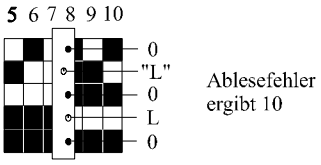
Stetig, denn D = 1 für benachbarte Codeworte → einschrittiger Code !!

Anwendung: 1.) Code-Lineale (Wegmessung)
2.) Code-Scheibe (Winkelmessung)

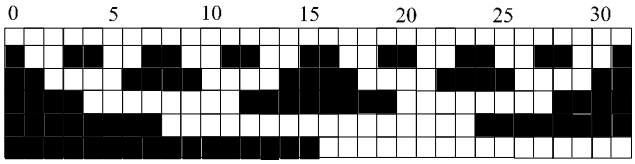
Problem: Übergang zwischen zwei Codeworten
Dualcode → Fehlerhafte Codeworte
Gray-Code → stets eins der beiden Codeworte



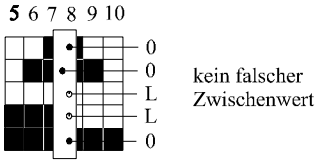
a.) Dualcode (Ablesemuster für Absolutmeßverfahren)



Ablesefehler beim Dualcode



b.) Gray-Code (Ablesemuster für Absolutmeßverfahren)



Ablesefehler-Vermeidung
beim Gray-Code

weis = L; schwarz = 0

2.1.2 Zifferncodes

2.1.2.1 8-4-2-1-Code

Zugeordnete Ziffer	Wertigkeit 8 4 2 1
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
	1 0 1 1
	1 1 0 0
	1 1 0 1
	1 1 1 0
	1 1 1 1

Stellenzahl	: 4
Bewertbar	: ja
Gewicht	: 1...3
Minimaldistanz	: 1
Maximaldistanz	: 4
Hamming-Distanz zweier benachbarter CW	: 1...4
Stetig	: nein
Redundanz	: 0,7

2.1.2.2 BCD-Zähl-Code

Zugeordnete Ziffer	Wertigkeit 1 1 1 1 1 1 1 1 1 1
1	0 0 0 0 0 0 0 0 0 1
2	0 0 0 0 0 0 0 0 1 1
3	0 0 0 0 0 0 0 1 1 1
4	0 0 0 0 0 0 1 1 1 1
5	0 0 0 0 0 1 1 1 1 1
6	0 0 0 0 1 1 1 1 1 1
7	0 0 0 1 1 1 1 1 1 1
8	0 0 1 1 1 1 1 1 1 1
9	0 1 1 1 1 1 1 1 1 1
0	1 1 1 1 1 1 1 1 1 1

Stellenzahl : 10
Bewertbar : ja
Gewicht : 1...10
Minimaldistanz : 1
Maximaldistanz : 9
Hamming Distanz zweier benachbarter CW: 1
Stetig : ja
Redundanz : 6,7

Anwendung: Wählvorgang beim Telefon (Impulswahlverfahren)

2.1.2.3 Aiken-Code

Zugeordnete Ziffer	Wertigkeit 2 4 2 1
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
nicht zugeordnet !	0 1 0 1
nicht zugeordnet !	0 1 1 0
nicht zugeordnet !	0 1 1 1
nicht zugeordnet !	1 0 0 0
nicht zugeordnet !	1 0 0 1
nicht zugeordnet !	1 0 1 0
5	1 0 1 1
6	1 1 0 0
7	1 1 0 1
8	1 1 1 0
9	1 1 1 1

Stellenzahl : 4
Bewertbar : ja
Gewicht : 0...4
Minimaldistanz : 1
Maximaldistanz : 4
Hamming Distanz zweier benachbarter CW: 1...4
Stetig : nein
Redundanz : 0,7

2.1.2.4 1aus10-Code

Zugeordnete Ziffern	Wertigkeit 9 8 7 6 5 4 3 2 1 0
0	0 0 0 0 0 0 0 0 0 1
1	0 0 0 0 0 0 0 0 1 0
2	0 0 0 0 0 0 0 1 0 0
3	0 0 0 0 0 0 1 0 0 0
4	0 0 0 0 0 1 0 0 0 0
5	0 0 0 0 1 0 0 0 0 0
6	0 0 0 1 0 0 0 0 0 0
7	0 0 1 0 0 0 0 0 0 0
8	0 1 0 0 0 0 0 0 0 0
9	1 0 0 0 0 0 0 0 0 0

Stellenzahl : 10
Bewertbar : ja
Gewicht : 1
Minimaldistanz : 2
Maximaldistanz : 2
Hamming Distanz zweier benachbarter CW: 2
Stetig : ja
Redundanz : 6,7

2.1.2.5 Stiebitz- (3-Excess-) Code

Zugeordnete Ziffer	Wertigkeit - - - -
nicht zugeordnet !	0 0 0 0
nicht zugeordnet !	0 0 0 1
nicht zugeordnet !	0 0 1 0
0	0 0 1 1
1	0 1 0 0
2	0 1 0 1
3	0 1 1 0
4	0 1 1 1
5	1 0 0 0
6	1 0 0 1
7	1 0 1 0
8	1 0 1 1
9	1 1 0 0
nicht zugeordnet !	1 1 0 1
nicht zugeordnet !	1 1 1 0
nicht zugeordnet !	1 1 1 1

Stellenzahl : 4
Bewertbar : nein
Gewicht : 1...3
Minimaldistanz : 1
Maximaldistanz : 4
Hamming Distanz zweier benachbarter CW: 1...4
Stetig : nein
Redundanz : 0,7

Merkmale: Codeworte „0000“ und „1111“ werden nicht verwendet !
Diese Codeworte können leicht durch Störung auftreten.

2.1.2.6 Libow-Craig-Code

Zugewiesene Ziffer	Wertigkeit - - - - -
0	0 0 0 0 0
1	0 0 0 0 1
2	0 0 0 1 1
3	0 0 1 1 1
4	0 1 1 1 1
5	1 1 1 1 1
6	1 1 1 1 0
7	1 1 1 0 0
8	1 1 0 0 0
9	1 0 0 0 0

Stellenzahl : 5
Bewertbar : nein
Gewicht : 1...5
Minimaldistanz : 1
Maximaldistanz : 5
Hamming Distanz zweier benachbarter CW: 1
Stetig : ja
Redundanz : 1,7

Merkmale: „gleitende" Bitmuster → Erzeugung mit Schieberegistern.

2.1.2.7 Ziffernsicherungscode Nr.3

Zugewiesene Ziffer	Wertigkeit - - - - -
0	0 1 0 1 1
1	1 1 1 0 0
2	1 1 0 1 0
3	1 1 0 0 1
4	1 0 1 1 0
5	1 0 1 0 1
6	1 0 0 1 1
7	0 0 1 1 1
8	0 1 1 1 0
9	0 1 1 0 1

Stellenzahl : 5
Bewertbar : nein
Gewicht : 3
Minimaldistanz : 2
Maximaldistanz : 4
Hamming Distanz zweier benachbarter CW: 2; 4
Stetig : nein
Redundanz : 1,7

2.2 Alphanumerische Codes
2.2.1 ASCII-Code

ASCII = American Standart Code for Information Interchange

Stellenzahl: 7 (Standart ASCII)
8 (Extended ASCII)

	HEX	0	1	2	3	4	5	6	7
HEX	BIN	0000	0001	0010	0011	0100	0101	0110	0111
0	0000	NUL	DLE	SP	0	@	P	`	p
1	0001	SOH	DC1	!	1	A	Q	a	q
2	0010	STX	DC2	“	2	B	R	b	r
3	0011	ETX	DC3	#	3	C	S	c	s
4	0100	EOT	DC4	$	4	D	T	d	t
5	0101	ENQ	NAK	%	5	E	U	e	u
6	0110	ACK	SYN	&	6	F	V	f	v
7	0111	BEL	ETB	‘	7	G	W	g	w
8	1000	BS	CAN	(	8	H	X	h	x
9	1001	HT	EM	)	9	I	Y	i	y
A	1010	LF	SUB	*	:	J	Z	j	z
B	1011	VT	ESC	+	;	K	[	k	{
C	1100	FF	FS	,	<	L	\	l	|
D	1101	CR	GS	-	=	M	]	m	}
E	1110	SO	RS	.	>	N	^	n	~
F	1111	SI	US	/	?	O	_	o	DEL

ASCII-Code, 7-Bit-Code nach DIN 66003

Erklärungen:

ACK	Positive Rückmeldung (Acknowledge)
BEL	Klingel (Bell)
BS	Rückwärtsschritt (Backspace)
CAN	Ungültig (Cancel)
CR	Wagenrücklauf (Carriage Return)
DC	Gerätesteuerzeichen (Device Control Characters)
DEL	Löschen (Delete)
DLE	Datenübertragungsumschaltung (Data Link Escape)
EM	Ende der Aufzeichnung (End of Medium)
ENQ	Stationsanforderung (Enquiry)
EOT	Ende der Übertragung (End of Transmission)
ESC	Code-Umschaltung (Escape)
ETB	Ende des Datenübertragungsblocks (End of Transmission Block)
ETX	Ende des Textes (End of Text)
FF	Formularvorschub (Form Feed)
FS	Hauptgruppen-Trennzeichen (File Separator)
GS	Gruppen-Trennzeichen (Group Separator)
HT	Horizontal-Tabulator (Horizontal Tabulation)
LF	Zeilenvorschub (Line Feed)
NAK	Negative Rückmeldung (Negative Acknowledge)
NUL	Füllzeichen (Null)
RS	Untergruppen-Trennzeichen (Record Separator)
SI	Rückschaltung (Shift-in)
SO	Dauerumschaltung (Shift-out)
SOH	Anfang des Kopfes (Start of Heading)
SP	Zwischenraum (Space)
STX	Anfang des Textes (Start of Text)

SUB	Substitutionszeichen (Substitute Character)
SYN	Synchronisierung (Synchronous Idle)
US	Teilgruppen-Trennzeichen (Unit Separator)
VT	Vertikal-Tabulator (Vertical Tabulation)

2.2.2 Fernschreibcode Nr.2 (Boudot-Code)

Stellenzahl: 5 (5-Spur-Lochstreifen)
↓
$2^5 = 32$ Zeichen , nicht ausreichend !!
→ Buchstaben-Gruppe (Einschalten mit Bu)
Ziffern-Gruppe (Einschalten mit Zi)

Problem: Fehler bei Bu- und Zi-Umschaltung → bis zur nächsten Umschaltung werden alle Zeichen falsch empfangen !

Nr.	Bu	Zi	1	2	T	3	4	5		Nr.	Bu	Zi	1	2	T	3	4	5
1	A	-	●	●	°					17	Q	1	●	●	°	●		●
2	B	?	●		°		●	●		18	R	4		●	°		●	
3	C	:		●	°	●	●			19	S	‘	●		°	●		
4	D	-F1-	●		°		●			20	T	5			°			●
5	E	3	●		°					21	U	7	●	●	°	●		
6	F	-F2-	●		°	●	●			22	V	=		●	°	●	●	●
7	G	-F2-		●	°		●	●		23	W	2	●	●	°			●
8	H	-F2-			°	●		●		24	X	/	●		°	●	●	●
9	I	8		●	°	●				25	Y	6	●		°	●		●
10	J	-F3-	●	●	°		●			26	Z	+	●		°			●
11	K	(	●	●	°	●	●			27	-F4-				°		●	
12	L	)		●	°			●		28	-F5-			●	°			
13	M	.			°	●	●	●		29	-F6-		●	●	°	●	●	●
14	N	,			°	●	●			30	-F7-		●	●	°		●	●
15	O	9			°		●	●		31	-F8-				°	●		
16	P	0		●	°	●		●		32	-F9-				°			

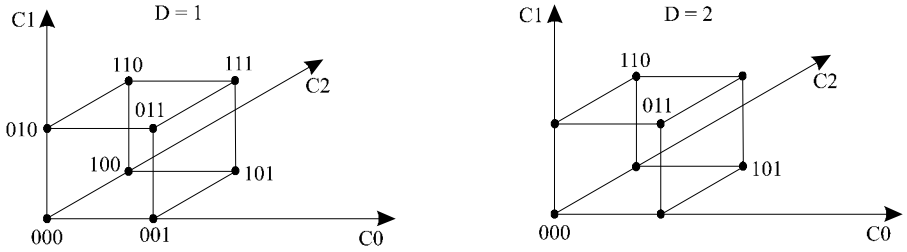
Erklärungen:

-F1-:	Wer da ?	-F6-:	Bu = Buchstabenumschaltung
-F2-:	(nicht belegt)	-F7-:	Zi = Ziffernumschaltung
-F3-:	Klingel	-F8-:	Zwischenraum
-F4-:	Wagenrücklauf	-F9-:	(nicht benutzt)
-F5-:	Zeilenvorschub		

2.3 Code-Sicherung

Problem: Fehlererkennung , Fehlerkorrektur
↓
Zusätzliche Information ! (R>0)

Beispiel: stetige Codes mit D = 1, D = 2
D = 1: 1 Bitfehler → gültiges Codewort → Fehler nicht erkennbar ! (R = 0)
D = 2: 1 Bitfehler → ungültiges Codewort → Fehler erkennbar !
2 Bitfehler → gültiges Codewort → Fehler nicht erkennbar !
→1-Fehler-erkennbarer Code !



3-Bit-Code, Codewörter ($C_2\ C_1\ C_0$)

$R = ld2^3 - ld2^2 = 3 - 2 = 1$

D = 3: 1,2 Bitfehler → ungültiges Codewort → Fehler erkennbar !
3 Bitfehler → gültiges Codewort → Fehler nicht erkennbar !
→ 2-Fehler-erkennbarer Code !

2.3.1 Fehlererkennungs-Codes

2.3.1.1 Parity-Check (Paritätskontrolle)

Vorgehen: Erweitern des Codes um eine Stelle (Paritätsbit). Ergänzen mit 0,1 so, daß gerade/ungerade Anzahl Einsen vorhanden (Parity e ven / Parity odd)
1 Bitfehler → Falsche Parität !

Beispiel: 8-4-2-1-Code
Es gibt in diesem Fall : $D_{min} \geq 2$ (inklusive Parity-Bit)
Denn: ohne Parity-Bit : $D_{min} \geq 1$

	$x_1,x_2.....x_n$	y
Codewort 1	1 1.....0	1
Codewort 2	1 1.....1	0
	D=1	

Dezimal-Ziffer	Wertigkeit 8 4 2 1	P_u	P_g
0	0 0 0 0	1	0
1	0 0 0 1	0	1
2	0 0 1 0	0	1
3	0 0 1 1	1	0
4	0 1 0 0	0	1
5	0 1 0 1	1	0
6	0 1 1 0	1	0
7	0 1 1 1	0	1
8	1 0 0 0	0	1
9	1 0 0 1	1	0

Redundanz: ohne Parity-Bit: $R = ldN_{pot} - ldN = ld2^n - ldN = n - ldN$

mit Parity-Bit: $R' = ldN_{pot}' - ldN = ld2^{n+1} - ldN = n + 1 - ldN$

R'= R+1

2.3.1.2 m-aus-n-Codes

Merkmal: Gleichgewichtige Codes, d.h. alle Codeworte besitzen identisches Gewicht G = m

Beispiel: 1-aus-10-Code, Ziffernsicherungscode Nr.3
→ identische Parität bei allen Codeworten !
→ keine zusätzliche Stelle für Parität erforderlich !

2.3.2 Fehlerkorrigierende Codes

2.3.2.1 Verfahren der Blocksicherung

Beispiel: Übertragung von 5743D mit Ziffernsicherungscode Nr.3
(z.B. ungerade Parität)

Ziffer	Codewort	G=3
5	1 0 1 0 1	
7	0 0 1 1 1	
4	1 0 0 1 0	← P_{Even}
3	1 1 0 0 1	
BCC	0 0 0 1 0	
	↑ P_{Even}	

BCC = Block Check Charakter
Hier: Ergänzen auf ungerade Parität → Längsparität (im Gegensatz zu Querparität)
oder: Ergänzen auf gerade Parität

keine Störung: Zeilen (ohne BCC) und Spalten (mit BCC) besitzen ungerade Parität
1Fehler 0: P_{Even} Spalte, P_{Even} Zeile → Fehler korrigierbar !
2 Fehler 0: P_{Even} in zwei Spalten → Fehler erkennbar, aber nicht korrigierbar !

Für Fehlererkennung alternative BCC möglich !
z.B.: CRC (Cyclic Redundancy Check)

2.3.2.2 Hamming-Codes

systematischer Code: n Info-Bits $x_1,x_2,.....x_n$ k Kontroll-Bits $y_1,y_2,.....y_n$
Gesamtzahl der Bits: n + k = m

Problem: Wieviele Kontroll-Bits sind erforderlich, damit 1 Fehler korrigiert werden kann ?

Hamming: k Stellen → 2^k Zustände
Codeprüfung → m + 1 Tatbestände
1.) Codewort korrekt (1. Tatbestand)
2.) fehlerhaftes Bit (m Tatbestände)

$$\rightarrow 2^k > n + k + 1$$

n	k
2	3
3	3
4	3
5	4
...	...
10	4

Beispiel: $n = 4 \rightarrow k = 3 \rightarrow m = 7$

Hamming-Code

$\underbrace{n \text{ Infobits}}_{x_1 \rightarrow x_n} + \underbrace{k \text{ Kontrollbits}}_{y_1 \rightarrow y_n}$

1 Fehler korrigierbar $\Downarrow$

$$2^k \geq n + k + 1$$

Beispiel: $n = 4 \rightarrow k = 3 \rightarrow m = 7$

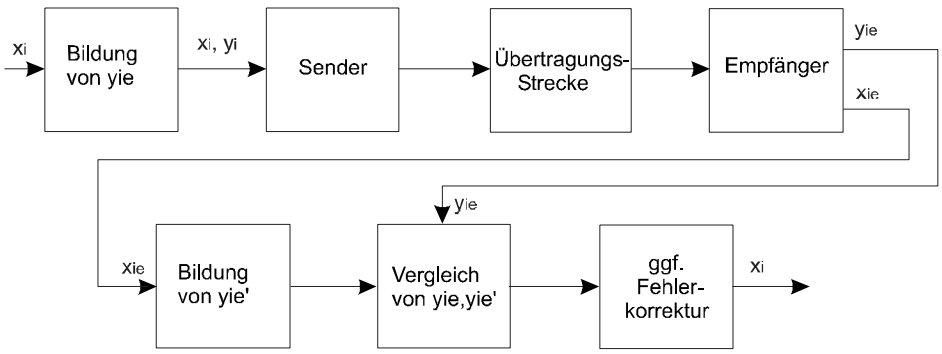
1.Schritt	Bit Duale Kennzahl	1 001	2 010	3 011	4 100	5 101	6 110	7 111
2.Schritt	1.Stellenbelegung Wert y_1 (PE)	y_1 0		x_1 0		x_2 1		x_3 1
3.Schritt	2.Stellenbelegung Wert y_2 (PE)		y_2 1	x_1 0			x_4 0	x_3 1
4.Schritt	3.Stellenbelegung Wert y_3 (PE)				y_3 0	x_2 1	x_4 0	x_3 1

Codes, die nach diesem Schema gebildet werden, nennt man Hamming-Codes

Verfälschung der Infobits:

x_1	x_2	x_3	x_4	y_1	y_2	y_3	
0	1	1	0	0	1	0	korrekt
0	1	1	1	0	1	0	falsch empfangen
				0	0	1	Berechnung der y_i aus falschem Codewort
				$\Downarrow$ kein Fehler in x_1, x_2, x_3	$\Downarrow$ x_4 falsch	$\Downarrow$ x_4 falsch	Annahme: Nur ein Fehler!!

Blockstruktur:



Verfälschung der Kontrollbits:

x_1	x_2	x_3	x_4	y_1	y_2	y_3	
0	1	1	0	0	1	0	korrekt
0	1	1	0	1	1	0	falsch empfangen
				0	1	0	Berechnung der y_i aus falschem Codewort
					⇓ x_1, x_3, x_4 korrekt	⇓ x_2, x_3, x_4 korrekt	Alle Info-Bits korrekt! → Korrekturbit falsch!! Annahme: Nur <u>ein</u> Fehler!!

3. Schaltalgebra

George Boole (1815-1864): Boole'sche Algebra (zweiwertige Logik)
C. E. Shannon (~ 1940) : Schaltalgebra (Beschreibung technischer binärer Schaltungen)

3.1 Schaltvariable und Schaltfunktion

Konstante: 0 , 1
Schaltvariable (logische Variable): Zustände, Werte = 0 , 1

Zuordnung	Sonnenschein
Sonne scheint nicht	0
Sonne scheint	1

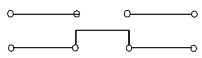
Schaltfunktion: Verknüpfung von Schaltvariablen

Beispiel: $y = \bar{a} = \neg a = \#a = /a$ (a nicht)
$y = a \vee b$ (a oder b)
$y = a \wedge b$ (a und b)

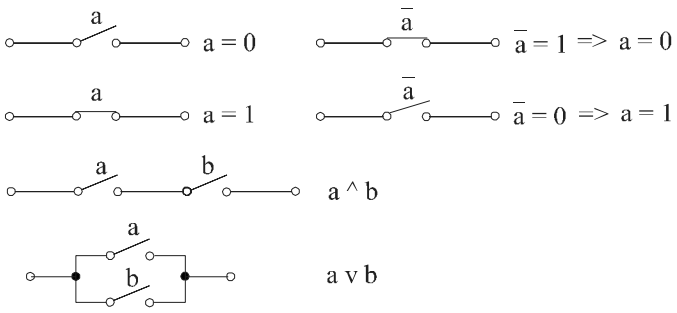
3.2 Rechenregeln

Anschaulich: Elektrische Verbindungen und Kontakte
Konstante: Verbindungen

0: Unterbrechung (kein Strom)
1: Kurzschluß (Strom)



Variable



Wahrheitstabelle

a	b	$\bar{a}$	$\bar{b}$	$a \vee b$	$a \wedge b$
0	0	1	1	0	0
0	1	1	0	1	0
1	0	0	1	1	0
1	1	0	0	1	1

Anzahl der Zeilen : $z = 2^n$
n: Anzahl der Variablen $\Rightarrow$ hier: $n = 2 \rightarrow z = 4$

3.2.1 Rechenregeln für Konstanten

1.) $0 \wedge 0 = 0$
2.) $0 \wedge 1 = 0$
3.) $1 \wedge 0 = 0$
4.) $1 \wedge 1 = 1$
5.) $0 \vee 0 = 0$
6.) $0 \vee 1 = 1$
7.) $1 \vee 0 = 1$
8.) $1 \vee 1 = 1$
9.) $\bar{0} = 1$
10.) $\bar{1} = 0$

3.2.2 Rechenregeln für Variablen und Konstanten

11.) $0 \wedge a = 0$
12.) $1 \wedge a = a$
13.) $a \wedge a = a$
14.) $\bar{a} \wedge a = 0$
15.) $0 \vee a = a$
16.) $1 \vee a = 1$
17.) $a \vee a = a$
18.) $\bar{a} \vee a = 1$
19.) $\bar{\bar{0}} = 0$
20.) $\bar{\bar{1}} = 1$
21.) $\bar{\bar{a}} = a$

Kommutatives Gesetz (Vertauschungsgesetz)

22.) $a \wedge b = b \wedge a$
23.) $a \vee b = b \vee a$

Assoziatives Gesetz (Zuordnungsgesetz)

24.) $(a \wedge b) \wedge c = a \wedge (b \wedge c) = a \wedge b \wedge c$
25.) $(a \vee b) \vee c = a \vee (b \vee c) = a \vee b \vee c$
Achtung! $(a \wedge b) \vee c \neq a \wedge (b \vee c)$

Distributives Gesetz (Verteilungsgesetz)

26.) $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$
27.) $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$

De Morgan'sches Theorem

28.) $\overline{a \vee b \vee c} = \bar{a} \wedge \bar{b} \wedge \bar{c}$

Beweis:

a	b	c	$a \vee b \vee c$	$\overline{a \vee b \vee c}$	$\bar{a}$	$\bar{b}$	$\bar{c}$	$\bar{a} \wedge \bar{b} \wedge \bar{c}$
0	0	0	0	1	1	1	1	1
0	0	1	1	0	1	1	0	0
0	1	0	1	0	1	0	1	0
0	1	1	1	0	1	0	0	0
1	0	0	1	0	0	1	1	0
1	0	1	1	0	0	1	0	0
1	1	0	1	0	0	0	1	0
1	1	1	1	0	0	0	0	0

Spalte 3 ist mit Spalte 5 identisch!!

allgemein: $\overline{a_1 \vee a_2 \vee a_3 \vee \ldots\ldots \vee a_n} = \bar{a}_1 \wedge \bar{a}_2 \wedge \bar{a}_3 \wedge \ldots\ldots \wedge \bar{a}_n$

Beweis: Sukzessive Anwendung des Theorems für 3 Variablen

29.) $\overline{a \wedge b \wedge c} = \bar{a} \vee \bar{b} \vee \bar{c}$

allgemein: $\overline{a_1 \wedge a_2 \wedge a_3 \wedge \ldots\ldots \wedge a_n} = \bar{a}_1 \vee \bar{a}_2 \vee \bar{a}_3 \vee \ldots\ldots \vee \bar{a}_n$

Beweis: wie bei 28.)

30.) $\overline{f(a_1, a_2, a_3, \ldots, a_n; \vee, \wedge)} = f(\bar{a}_1, \bar{a}_2, \bar{a}_3, \ldots, \bar{a}_n; \wedge, \vee)$

Beweis: $\overline{a \; op \; b} = \bar{a} \; \overline{op} \; \bar{b} \qquad op = \{\wedge, \vee\}$

$$y = \overline{\underbrace{a_1 \; op_1 \; a_2 \; op_2 \; a_3 \; op_3 \ldots\ldots a_{n-1}}_{u} \; op_{n-1} \; a_n} = \overline{u \; op_{n-1} \; a_n} = \bar{u} \; \overline{op_{n-1}} \; \bar{a}_n$$

$$\bar{u} = \overline{\underbrace{a_1 \; op_1 \; a_2 \; op_2 \; a_3 \; op_3 \ldots\ldots a_{n-2}}_{v} \; op_{n-2} \; a_{n-1}} = \overline{v \; op_{n-2} \; a_{n-1}} = \bar{v} \; \overline{op_{n-2}} \; \overline{a_{n-1}}$$

$$\Downarrow$$

$$y = \bar{u} \; \overline{op_{n-1}} \; \bar{a}_n = \bar{v} \; \overline{op_{n-2}} \; \overline{a_{n-1}} \; \overline{op_{n-1}} \; \bar{a}_n = \bar{a}_1 \; \overline{op_1} \; \bar{a}_2 \; \overline{op_2} \; \bar{a}_3 \; \ldots\ldots \; \overline{op_{n-1}} \; \bar{a}_n$$

Beispiel: $\overline{(\bar{a} \vee b) \wedge \bar{c}} = (a \wedge \bar{b}) \vee c$

Kürzungsregeln

31.) $a \vee (a \wedge b) = (a \wedge 1) \vee (a \wedge b) = a \wedge \underbrace{(1 \vee b)}_{=1} = a$

32.) $a \wedge (a \vee b) = \underbrace{(a \wedge a)}_{=a} \vee (a \wedge b) = a \vee (a \wedge b) = a$

33.) $a \vee (\bar{a} \wedge b) = \underbrace{(a \vee \bar{a})}_{=1} \wedge (a \vee b) = a \vee b$

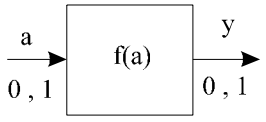
34.) $a \wedge (\bar{a} \vee b) = \underbrace{(a \wedge \bar{a})}_{=0} \vee (a \wedge b) = a \wedge b$

35.) $(a \wedge b) \vee (a \wedge \bar{b}) = a \wedge \underbrace{(b \vee \bar{b})}_{=1} = a$

36.) $(a \vee b) \wedge (a \vee \bar{b}) = a \vee \underbrace{(b \wedge \bar{b})}_{=0} = a$

3.2.3 Verknüpfungen und Schaltsymbole

Schaltelemente mit einem Eingang



Anzahl der möglichen Verknüpfungen ?

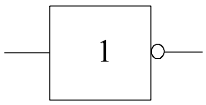
a	$\overbrace{0\quad 1}^{k=2}$
$V=4\begin{cases} y_0 \\ y_1 \\ y_2 \\ y_3 \end{cases}$	$\begin{matrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{matrix}$

k: Anzahl der Zustände am Eingang

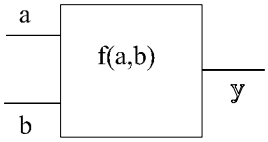
Anzahl der Verknüpfungen: $V = 2^k = 2^2 = 4$

1.) $a = x \rightarrow y = 0$ (Nullfunktion)
2.) $y = a$ (Identität)
3.) $y = \bar{a}$ (Negation)
4.) $a = x \rightarrow y = 1$ (Einsfunktion)

Schaltsysmbol für Negation (DIN 40900):



Schaltelemente mit zwei Eingängen



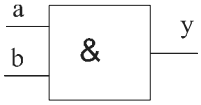
a b	$\begin{matrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{matrix}\}k=4$	y=f(a,b)	Bezeichnung der Funktion	Sprechweise	Gleichwertige Schreibweise
y_0	0 0 0 0	0	Nullfunktion		
y_1	0 0 0 1	$a \wedge b$	UND – Verknüpfung	a und b	
y_2	0 0 1 0	$\neg a \wedge b$	1. Inhibition	nicht a und b	$\bar{a} \wedge b$
y_3	0 0 1 1	b	1. Identität		
y_4	0 1 0 0	$a \wedge \neg b$	2. Inhibition	a und nicht b	$a \wedge \bar{b}$
y_5	0 1 0 1	a	2. Identität		
y_6	0 1 1 0	$\neg(a \leftrightarrow b)$	Antivalenz	a antivalent b	$(a \wedge \bar{b}) \vee (\bar{a} \wedge b)$
y_7	0 1 1 1	$a \vee b$	ODER – Verknüpfung	a oder b	
y_8	1 0 0 0	$\overline{a \vee b}$	NOR – Verknüpfung	a nor b	$\neg(a \vee b)$
y_9	1 0 0 1	$a \leftrightarrow b$	Äquivalenz	a Doppelpfeil b	$(a \wedge b) \vee (\bar{a} \wedge \bar{b})$
y_{10}	1 0 1 0	$\neg a$	1. Negation	nicht a	$\bar{a}$
y_{11}	1 0 1 1	$a \rightarrow b$	1. Implikation	a Pfeil b	$\neg a \vee b$
y_{12}	1 1 0 0	$\neg b$	2. Negation	nicht b	$\bar{b}$
y_{13}	1 1 0 1	$b \rightarrow a$	2. Implikation	b Pfeil a	$a \vee \bar{b}$
y_{14}	1 1 1 0	$\overline{a \wedge b}$	NAND – Verknüpfung	a nand b	$\neg(a \wedge b)$
y_{15}	1 1 1 1	1	Einsfunktion		

$y = 16$

Wichtige Verknüpfungen

1.) **Konjunktion, UND, y_1**

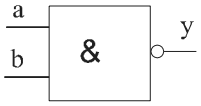
a	b	y
0	0	0
0	1	0
1	0	0
1	1	1



$$y = a \wedge b$$

2.) **NAND, y_{14}**

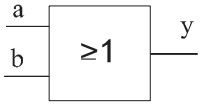
a	b	y
0	0	1
0	1	1
1	0	1
1	1	0



$$y = \overline{a \wedge b} = \bar{a} \vee \bar{b}$$

3.) **Disjunktion, ODER, y_7**

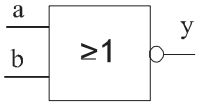
a	b	y
0	0	0
0	1	1
1	0	1
1	1	1



$$y = a \vee b$$

4.) **NOR, y_8**

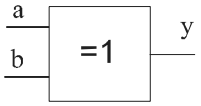
a	b	y
0	0	1
0	1	0
1	0	0
1	1	0



$$y = \overline{a \vee b} = \bar{a} \wedge \bar{b}$$

5.) **Antivalenz, Exklusiv-ODER, EXOR, y_6**

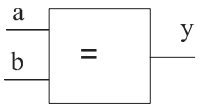
a	b	y
0	0	0
0	1	1
1	0	1
1	1	0



$$y = (\bar{a} \wedge b) \vee (a \wedge \bar{b}) = a \veebar b$$

6.) **Äquivalenz, Exklusiv-NOR, y_9**

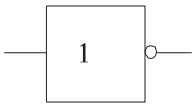
a	b	y
0	0	1
0	1	0
1	0	0
1	1	1



$$y = (\bar{a} \wedge \bar{b}) \vee (a \wedge b) = a \leftrightarrow b$$

7.) **1.Negation, y_{12}**

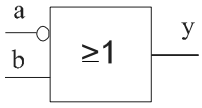
a	b	y
0	0	1
0	1	1
1	0	0
1	1	0



$$y = \bar{a}$$

8.) **1.Implikation, y_{13}**

a	b	y
0	0	1
0	1	1
1	0	0
1	1	1

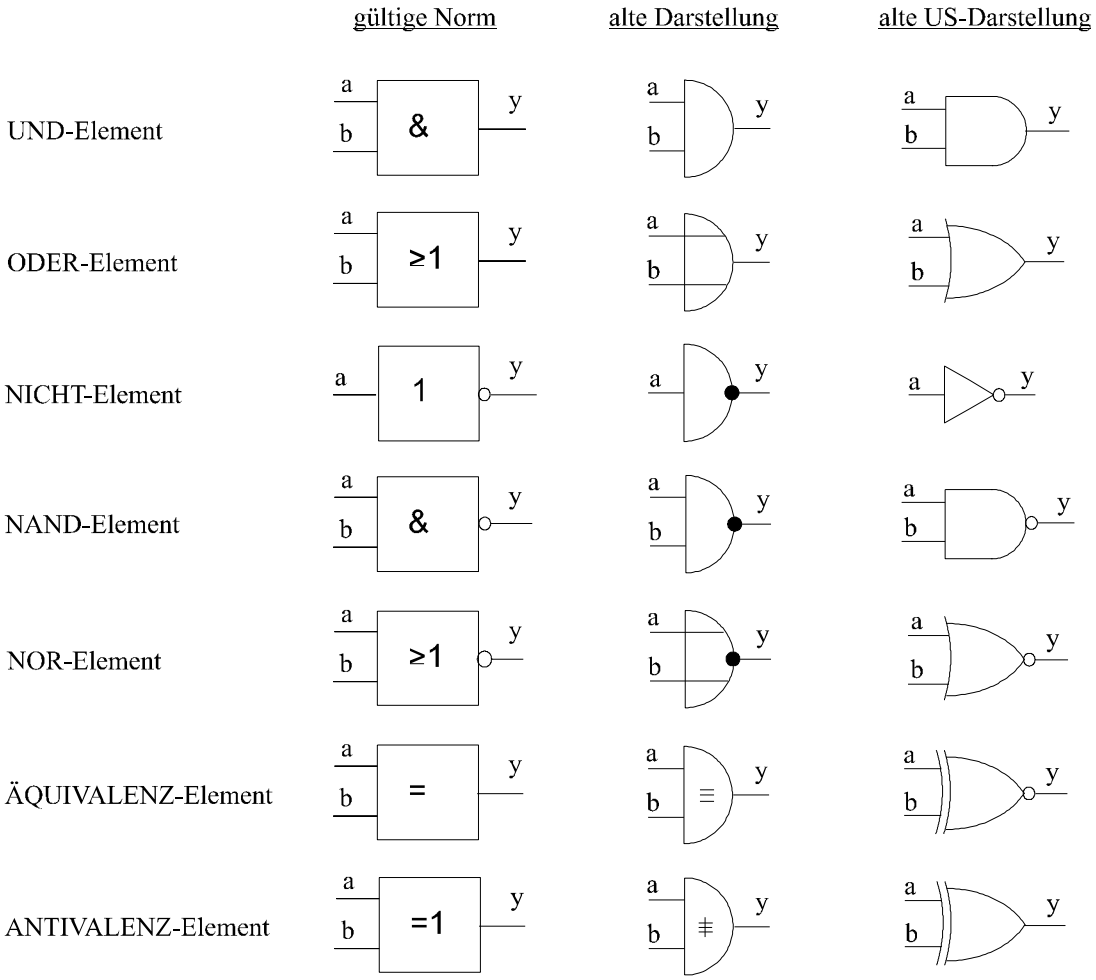


$$y = a \rightarrow b = \bar{a} \vee b$$

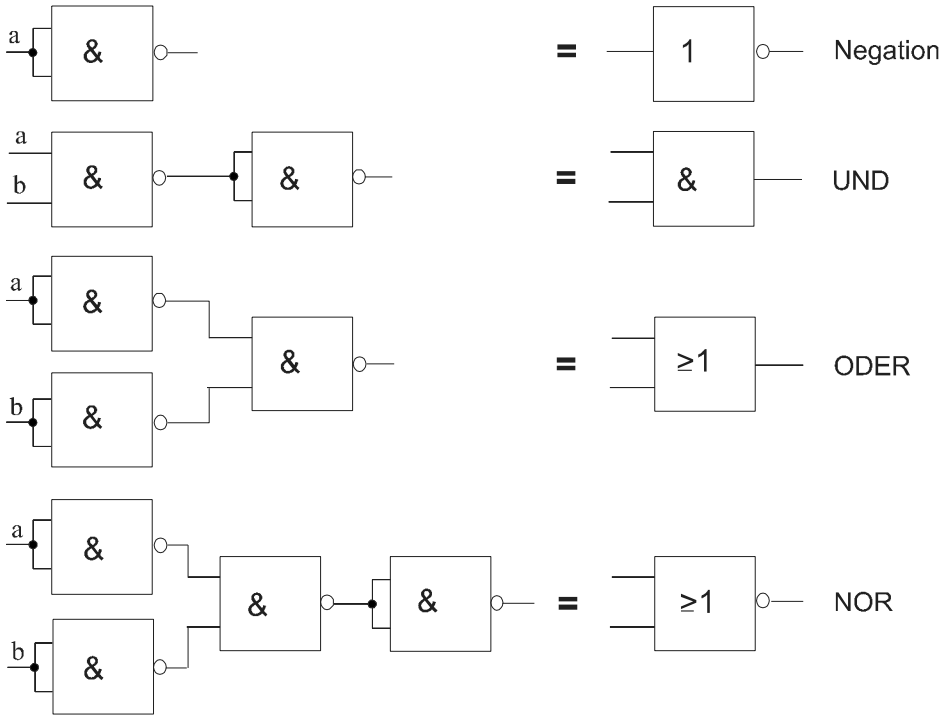
Satz: Alle Verknüpfungen lassen sich zurückführen auf die Basisverknüpfungen UND, ODER und Negation!

Problem: Nur NAND vorhanden!

Satz: Alle Verknüpfungen lassen sich nur mit NAND-Elementen oder nur mit NOR-Elementen realisieren.

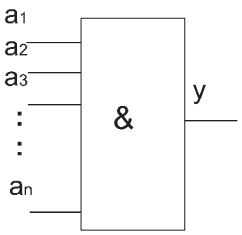


Gültige Schaltsymbole (DIN 40900, Teil 12) und alte Symbole

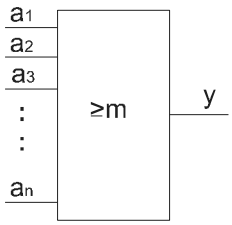


Schaltsysmbole mit mehr als zwei Eingängen:

1.) **UND:**

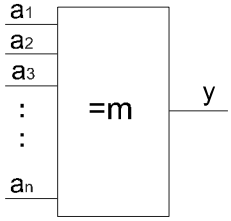


2.) **Schwellwert:**



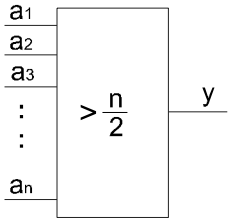
$$y = \begin{cases} \textbf{1, wenn } \sum_{i=1}^{n} a_i \geq m \\ \textbf{0, sonst} \end{cases}$$

3.) **m aus n:**



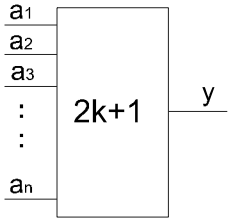
$$y = \begin{cases} 1, \text{ wenn } \sum_{i=1}^{n} a_i = m \\ 0, \text{ sonst} \end{cases}$$

4.) **Majorität:**



$$\mathbf{y = \begin{cases} 1, \text{ wenn } \sum_{i=1}^{n} a_i > \frac{n}{2} \\ 0, \text{ sonst} \end{cases}}$$

5.) **Ungerade:**



$$\mathbf{y = \begin{cases} 1, \text{ wenn } \sum_{i=1}^{n} a_i \text{ ungerade} \\ 0, \text{ wenn } \sum_{i=1}^{n} a_i \text{ gerade} \end{cases}}$$

3.2.4 Vorrangregeln (DIN 66000)

1.) Außenklammern einer einzeln stehenden Funktion können entfallen
2.) ¬ bindet stärker als alle anderen Zeichen; UND, ODER, NAND, NOR binden stärker als →, ↔, ¿.
3.) UND, ODER, NAND, NOR binden unter sich gleich stark
4.) Außenklammern einer UND-Verknüpfung, die selbst Teil einer UND-Verknüpfung ist, können entfallen. Ebenso: →, ↔, ¿

Beispiel: (* bezeichnet die zu bearbeitenden Klammern)

$$*\Big(\big(\big((a \wedge b) \vee c\big) \vee d\big) \leftrightarrow \big((a \wedge b) \wedge \neg(c \vee d)\big)\Big)*$$

$$\Downarrow \text{ 1.)}$$

$$*\Big(\big((a \wedge b) \vee c\big) \vee d\Big)* \leftrightarrow *\big((a \wedge b) \wedge \neg(c \vee d)\big)*$$

$$\Downarrow \text{ 2.), 3.)}$$

$$*\big((a \wedge b) \vee c\big)* \vee d \leftrightarrow *(a \wedge b)* \vee \neg(c \vee d)$$

$$\Downarrow \text{ 4.)}$$

$$(a \wedge b) \vee c \vee d \leftrightarrow a \wedge b \vee \neg(c \vee d)$$

3.2.5 Normalformen

Beispiel: Aufzugsteuerung

1.) Formulieren des Problems

- Gebäude mit 8 Etagen
- mehrere Aufzüge
- ein Aufzug soll die Etagen 0, 4, 5, 6, 7 verbinden
- die Etagennummer 0 bis 7 steht dual zur Verfügung
- Bildung eines Signals aus der Etagennummer, das den Wert 1 annimmt, wenn der Aufzug halten darf.

2.) Definition der Variablen

Etagen-Nr.: $\mathbf{2^2\ 2^1\ 2^0}$ / **c b a**

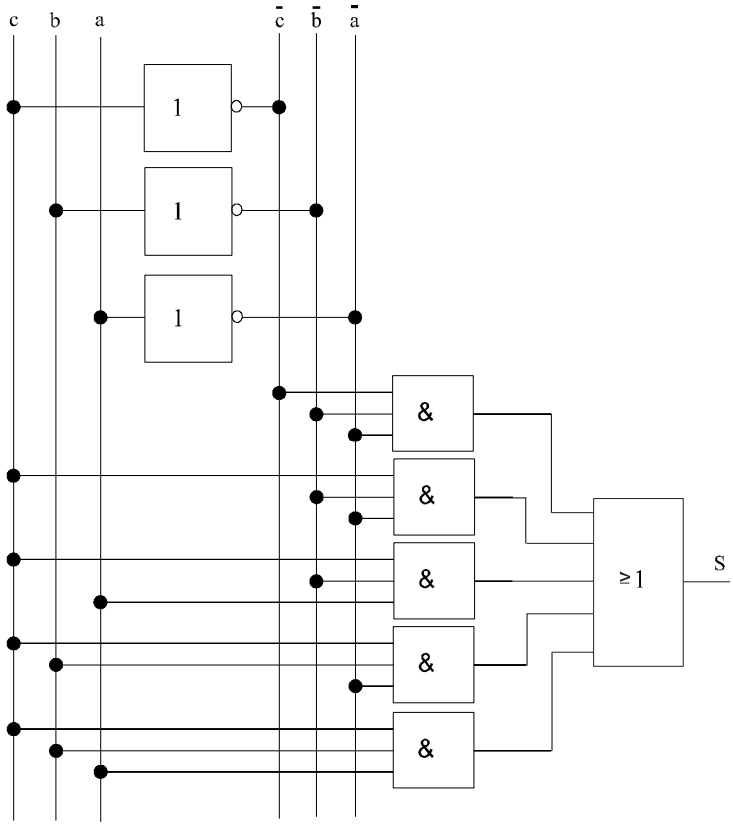
Signal $\mathbf{S} = \begin{cases} \mathbf{0: kein\ Halt} \\ \mathbf{1: Halt} \end{cases}$

3.) Wahrheitstabelle

Etage	c	b	a	S
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	0
4	1	0	0	1
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

4.) Schaltfunktion

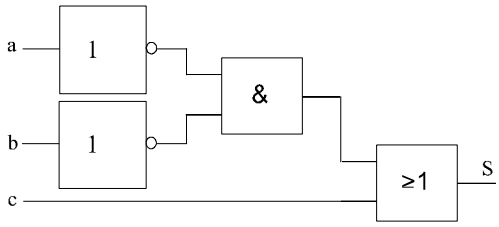
$$\mathbf{S = (\bar{a} \wedge \bar{b} \wedge \bar{c}) \vee (\bar{a} \wedge \bar{b} \wedge c) \vee (a \wedge \bar{b} \wedge c) \vee (\bar{a} \wedge b \wedge c) \vee (a \wedge b \wedge c)}$$



Schaltung Aufzugsteuerung ohne Minimierung

5.) Minimieren der Schaltfunktion

$$\rightarrow S = \left(\bar{a} \wedge \bar{b}\right) \vee c$$



Schaltung Aufzug mit Minimierung

3.2.5.1 Disjunktive Normalform (DNF)

Definition: Die Disjunktive Normalform einer Schaltfunktion erhält man, wenn man alle Eintragungen in der Wahrheitstabelle, für die die Funktion den Wert 1 annimmt, disjunktiv (ODER) miteinander verknüpft. Die Eingangsvariablen jeder Eintragung werden konjunktiv (UND) verknüpft.

Beispiel:

Etage	c b a	m_0 m_1 m_2 ... m_7	y	$\bar{y}$
0	0 0 0	1 0 0 ... 0	1	0
1	0 0 1	0 1 0 ... 0	0	1
2	0 1 0	0 0 1 ... 0	0	1
3	0 1 1	0 0 0 ... 0	0	1
4	1 0 0	0 0 0 ... 0	1	0
5	1 0 1	0 0 0 ... 0	1	0
6	1 1 0	0 0 0 ... 0	1	0
7	1 1 1	0 0 0 ... 1	1	0
		Minterme		

$$y = \underbrace{(\bar{a} \wedge \bar{b} \wedge \bar{c})}_{m_0} \vee \underbrace{(\bar{a} \wedge \bar{b} \wedge c)}_{m_4} \vee \underbrace{(a \wedge \bar{b} \wedge c)}_{m_5} \vee \underbrace{(\bar{a} \wedge b \wedge c)}_{m_6} \vee \underbrace{(a \wedge b \wedge c)}_{m_7}$$

$$= m_0 \vee m_4 \vee m_5 \vee m_6 \vee m_7 \text{ : DNF}$$

Minterm : Die zugehörige Schaltfunktion nimmt nur für eine einzige Eingangskombination den Wert 1 an.

$\bar{y} = m_1 \vee m_2 \vee m_3$: DNF der Umkehrfunktion

3.2.5.2 Konjunktive Normalform (KNF)

Definition: Die konjunktive Normalform einer Schaltfunktion erhält man, wenn man alle Eintragungen in der Wahrheitstabelle, für die die Funktion den Wert 0 annimmt, konjunktiv (UND) miteinander verknüpft. Die Eingangsvariablen jeder Eintragung werden disjunktiv (ODER) verknüpft.

Beispiel:

Etage	c b a	M_7 M_6 M_5 ... M_0	y	$\bar{y}$
0	0 0 0	0 1 1 ... 1	1	0
1	0 0 1	1 0 1 ... 1	0	1
2	0 1 0	1 1 0 ... 1	0	1
3	0 1 1	1 1 1 ... 1	0	1
4	1 0 0	1 1 1 ... 1	1	0
5	1 0 1	1 1 1 ... 1	1	0
6	1 1 0	1 1 1 ... 1	1	0
7	1 1 1	1 1 1 ... 0	1	0
		Maxterme		

$$y = \underbrace{(\bar{a} \vee b \vee c)}_{M_6} \wedge \underbrace{(a \vee \bar{b} \vee c)}_{M_5} \wedge \underbrace{(\bar{a} \vee \bar{b} \vee c)}_{M_4}$$

$$= M_4 \wedge M_5 \wedge M_6 \text{ : KNF}$$

$\bar{y} = M_0 \wedge M_1 \wedge M_2 \wedge M_3 \wedge M_7$: KNF der Umkehrfunktion

Maxterm : Die zugehörige Schaltfunktion nimmt nur für eine einzige Eingangskombination den Wert 0 an.

Beispiel: $M_7 = a \vee b \vee c$

$M_0 = \bar{a} \vee \bar{b} \vee \bar{c}$

4. Logisch-physikalische Zusammenhänge

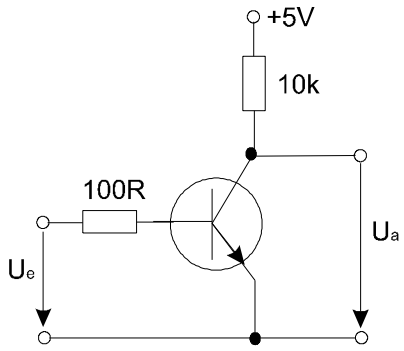
Logische Zustände		Physikalische Zustände
0	↔	Hahn zu, 0...2 bar, 0...+1V
1	↔	Hahn auf, 9....10 bar, +4...+5V (elektr. Realisierung)

Definition: Logikpegel
L (Low) : negativere Spannung
H (High) : positivere Spannung
bei gemeinsamem Bezugspotential !!

Beispiel: L = 0...+1V, H = +4...+5V
L = -10...-8V, H = +8...+10V

4.1 Verwendung von Logikpegeln

Inverter:

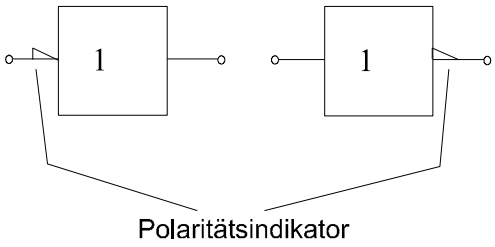


L: 0...+0,5V
H: +4...+5V

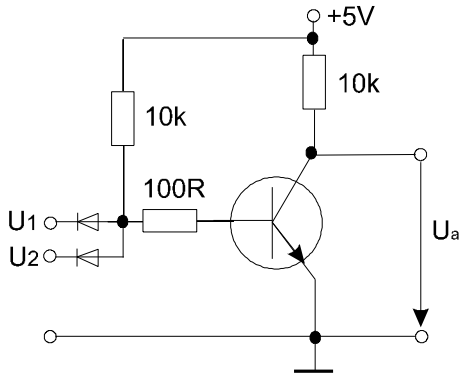
Arbeitstabelle

U_e	U_a
L	H
H	L

Symbole:

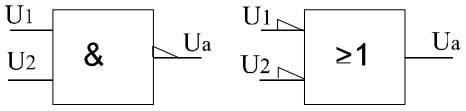


Verknüpfung von zwei Eingängen



U_1	U_2	U_a
L	L	H
L	H	H
H	L	H
H	H	L

Symbole:



4.2 Zuordnung von logischen Zuständen zu Logikpegeln

	positive Logik	negative Logik
0	L	H
1	H	L

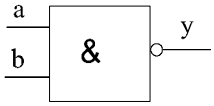
Wahrheitstabelle ↔ Arbeitstabelle

Beispiel: s.o.

pos. Logik

a	b	y
0	0	1
0	1	1
1	0	1
1	1	0

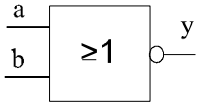
NAND



neg. Logik

a	b	y
1	1	0
1	0	0
0	1	0
0	0	1

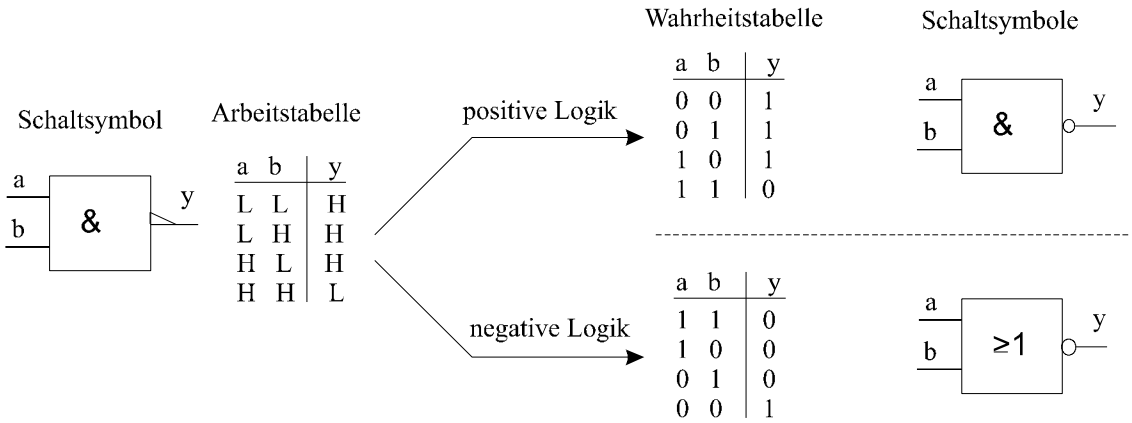
NOR



Positive Logik			Negative Logik	
Verknüpfung	Wahrheitstabelle	Arbeitstabelle	Wahrheitstabelle	Verknüpfung
UND	0 0 0 0 1 0 1 0 0 1 1 1	L L L L H L H L L H H H	1 1 1 1 0 1 0 1 1 0 0 0	ODER
ODER	0 0 0 0 1 1 1 0 1 1 1 1	L L L L H H H L H H H H	1 1 1 1 0 0 0 1 0 0 0 0	UND
NAND	0 0 1 0 1 1 1 0 1 1 1 0	L L H L H H H L H H H L	1 1 0 1 0 0 0 1 0 0 0 1	NOR
Antivalenz	0 0 0 0 1 1 1 0 1 1 1 0	L L L L H H H L H H H L	1 1 1 1 0 0 0 1 0 0 0 1	Äquivalenz
Äquivalenz	0 0 1 0 1 0 1 0 0 1 1 1	L L H L H L H L L H H H	1 1 0 1 0 1 0 1 1 0 0 0	Antivalenz

Umwandlung Pegelsymbol → Logiksymbol

pos. Logik: Alle Polaritätsindikatoren werden durch Inversionskreise ersetzt
neg. Logik: - Anschluß mit Polaritätsindikator: → Polaritätsindikator entfernen
- Anschluß ohne Polaritätsindikator: → Inversionskreis einsetzen



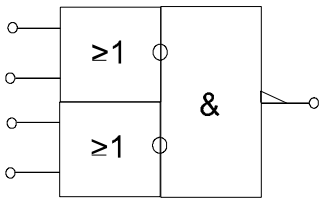
Umformung von Schaltsymbolen mit Polaritätsindikatoren in Schaltsymbole mit Inversionskreisen

4.3 Verwendung von Polaritätsindikatoren

Empfehlung nach DIN 40900 Teil 12: Inversionskreise sind vorzuziehen!

Regel: Inversionskreis und Polaritätsindikator dürfen nicht gleichzeitig in einem Schaltplan auftreten.
Als Ausnahme: Interne Verbindungen

Beispiel:



5.Minimieren von Schaltfunktionen mit dem KV-Diagramm

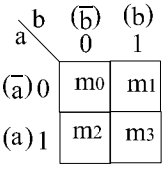
Problem: Vereinfachung (Minimieren) komplexer Schaltfunktionen

Alternativen: - algebraisch
- graphisch
- numerisch

Definition: Karnaugh-Veitch-Diagramm (KV-Diagramm, KV-Tafel)
Grafische Darstellung aller Minterme einer Schaltfunktion

5.1 KV-Diagramme für zwei Schaltvariablen

a	b	Minterme
0	0	$\bar{a} \wedge \bar{b} = m_0$
0	1	$\bar{a} \wedge b = m_1$
1	0	$a \wedge \bar{b} = m_2$
1	1	$a \wedge b = m_3$



Anzahl der Schaltvariablen n = 2

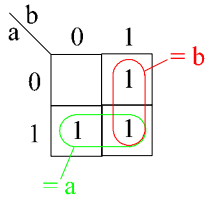
Anzahl der Felder:
$F = 2^n = 4$

Beispiel: Minimieren der Schaltfunktion $y = (a \wedge b) \vee (a \wedge \bar{b}) \vee (\bar{a} \wedge b)$ DNF

a.) algebraisch:

$$y = \underbrace{\left(\underbrace{a \wedge (b \vee \bar{b})}_{=1} \right)}_{a} \vee (\bar{a} \wedge b) = a \vee (\bar{a} \wedge b) = \underbrace{(a \vee \bar{a})}_{=1} \wedge (a \vee b) = \underline{\underline{a \vee b}}$$

b.) graphisch:

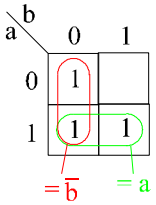


$\rightarrow y = a \vee b$ DMF
(Disjunktive Minimalform)

Vorgehen:
1.) Für jeden Minterm der Schaltfunktion wird eine „1“ in das entsprechende Feld des KV-Diagramm eingetragen.
2.) Zusammenfassung der Felder zu möglichst großen Blöcken (mit je 2^n Feldern). Überschneidungen sind möglich. Alle Felder müssen erfaßt werden.
3.) DMF = Oder-Verknüpfung aller Blöcke

Beispiel: KV-Diagramm aus Wahrheitstabelle

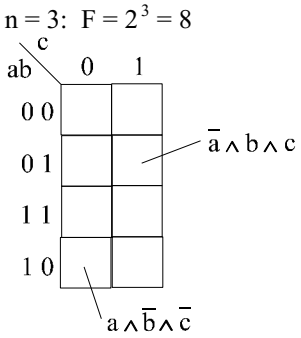
a	b	y
0	0	1
0	1	0
1	0	1
1	1	1



$y = a \vee \bar{b}$

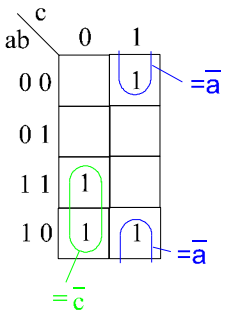
Vorgehen:
1.) Für jeden Zustand y = 1 in der Wahrheitstabelle wird im entsprechenden Feld des KV-Diagramms eine „1“ eingetragen
2.) Zusammenfassung der Felder zu möglichst großen Blöcken (mit je 2^n Feldern). Überschneidungen sind möglich. Alle Felder müssen erfaßt werden.
3.) DMF = Oder-Verknüpfung aller Blöcke

5.2 KV-Diagramme mit mehr als zwei Variablen



Beispiel:

a	b	c	y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0



$$y = (a \wedge \bar{c}) \vee (\bar{b} \wedge c)$$

5.3 Redundanzen

Problem: Bestimmte Kombinationen der Eingangsvariablen können nicht auftreten oder der zugehörige Zustand der Ausgangsvariablen ist beliebig (Don't care-Zustand , Redundanz: x)

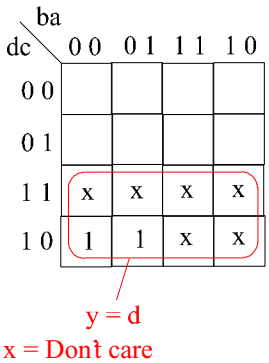
Beispiel: a, b, c, d → y

↑

Ziffern im 8-4-2-1-Code

d	c	b	a	y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	x
1	0	1	1	x
1	1	0	0	x
1	1	0	1	x
1	1	1	0	x
1	1	1	1	x

y = 1, wenn Ziffer ≥ 8, sonst 0



x = Don't care

Vorgehen: x = 0 oder 1, sodaß eine möglichst einfache Schaltfunktion resultiert

5.4 Konjunktive Minimalform

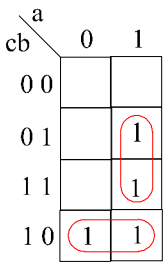
Bisher: KNF von y = invertierte DNF von $\bar{y}$

↓ Minimieren

KMF von y = invertierte DMF von $\bar{y}$

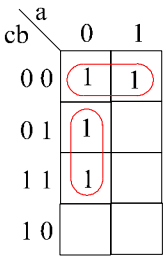
Beispiel:

c	b	a	y	$\bar{y}$
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	1	0
1	0	1	1	0
1	1	0	0	1
1	1	1	1	0



$$y = (a \wedge b) \vee (\bar{b} \wedge c) \quad \text{DMF}$$

$\bar{y}$:

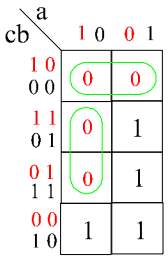


$$\bar{y} = (\bar{a} \wedge b) \vee (\bar{b} \wedge \bar{c})$$

$$y = \overline{(\bar{a} \wedge b) \vee (\bar{b} \wedge \bar{c})} = \overline{(\bar{a} \wedge b)} \wedge \overline{(\bar{b} \wedge \bar{c})} =$$

$$= \underline{\underline{(a \vee \bar{b}) \wedge (b \vee c)}} \quad \text{KMF}$$

Vereinfachung: Ablesen der KMF für y aus dem KV-Diagramm für y durch Betrachtung der Nullen!
Achtung! Äußerste Vorsicht!!



5.6 Vorgehen bei der Minimierung

1.) Wertetabelle
2.) KV-Diagramm
   a.) Bildung möglichst großer weniger Blöcke
   b.) DMF: „1"-Felder und gegebenenfalls „x"-Felder zusammenfassen
   c.) Beginn bei kleinsten Blöcken (isolierte Felder), anschließend 2-fach-Blöcke, dann 4-fach-Blöcke, etc.
   d.) gibt es mehrere Alternativen für Blöcke, so beginnt man mit den Blöcken, für die keine Alternativen bestehen
3.) Ablesen der DMF
4.) Realisierung des Schaltnetzes

KMF: DMF für $\bar{y}$ + invertieren

6. Elektronische Realisierung der Schaltelemente

Logikpegel (Kap.4): H, L
↑ pos./neg. Logik
↓
Logische Zustände: 0, 1

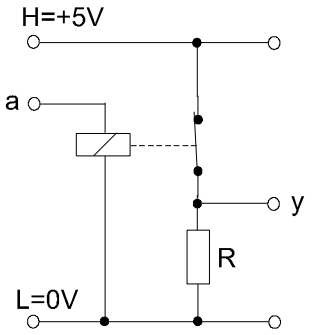
Im folgenden: stets positive Logik! H À 1, L À 0

Es genügt die Realisierung von:
- UND + NEG (NAND)
  oder
- ODER + NEG (NOR)

alle anderen Schaltelemente können hiervon abgeleitet werden.

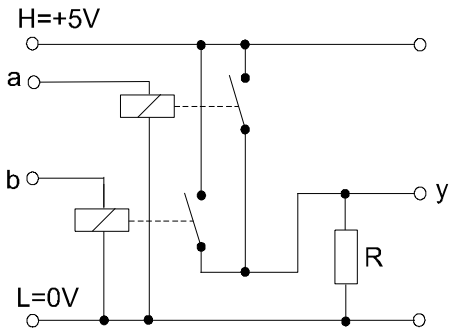
6.1 Relais-Logik

NEG:



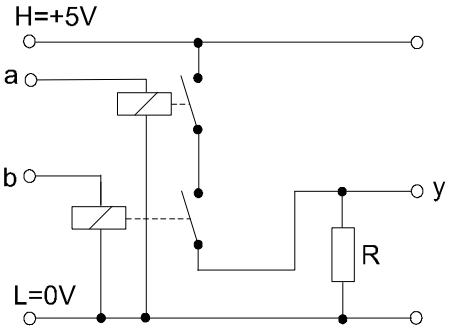
a	y
L	H
H	L

ODER:



a	b	y
L	L	L
L	H	H
H	L	H
H	H	H

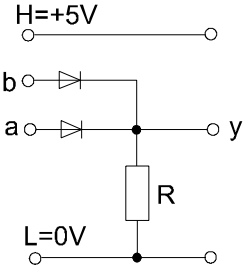
UND:



a	b	y
L	L	L
L	H	L
H	L	L
H	H	H

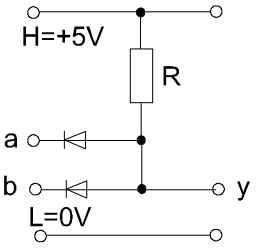
6.2 Dioden-Logik

ODER:



a	b	y
L	L	L
L	H	H
H	L	H
H	H	H

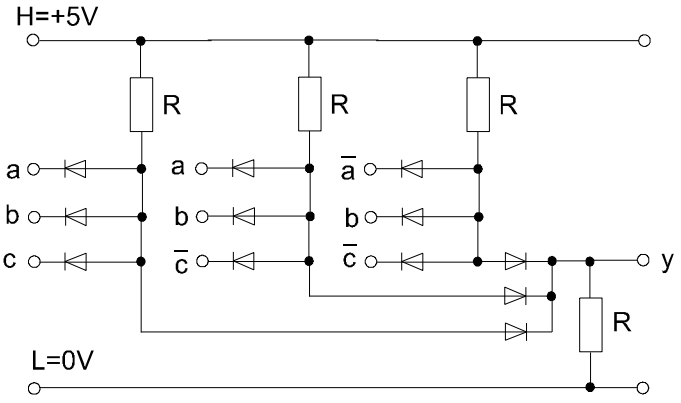
UND:



a	b	y
L	L	L
L	H	L
H	L	L
H	H	H

Negation: nicht realisierbar!

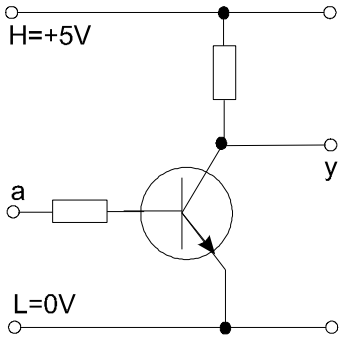
DMF/DNF:



$$y = (a \wedge b \wedge c) \vee (a \wedge b \wedge \bar{c}) \vee (\bar{a} \wedge b \wedge \bar{c})$$

6.3 Transistor-Logik

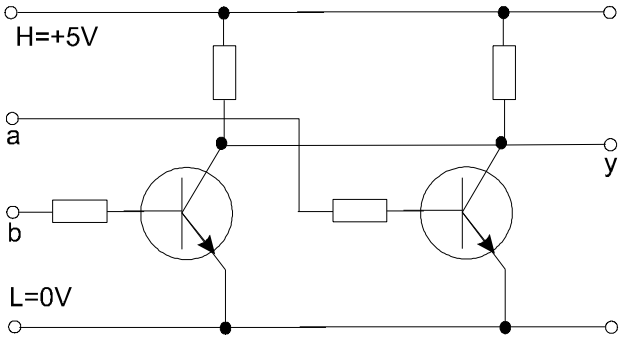
NEG:



a	y
L	H
H	L

6.3.1 Resistor-Transistor-Logik (RTL)

NOR:

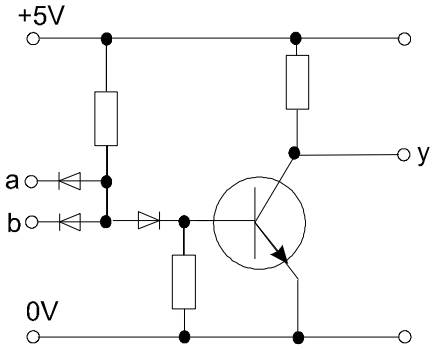


a	b	y
L	L	H
L	H	L
H	L	L
H	H	L

Nachteil: Basisströme relativ hoch, damit Transistoren voll leitend sind!

6.3.2 Dioden-Transistor-Logik (DTL)

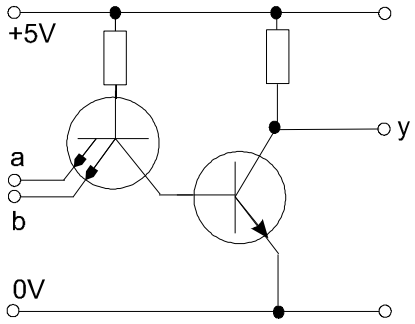
NAND:



a	b	y
L	L	H
L	H	H
H	L	H
H	H	L

6.3.3 Transistor-Transistor-Logik (TTL)

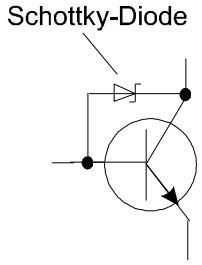
NAND:



a	b	y
L	L	H
L	H	H
H	L	H
H	H	L

Doppel-Emitter-Transistor ersetzt Dioden aus 6.3.2!

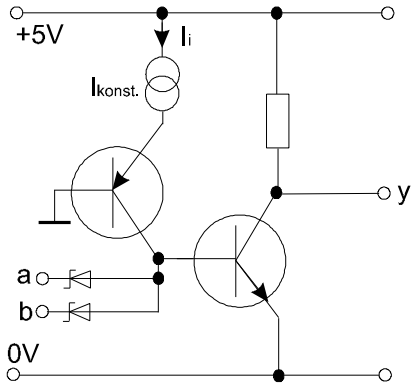
Schottky-TTL:

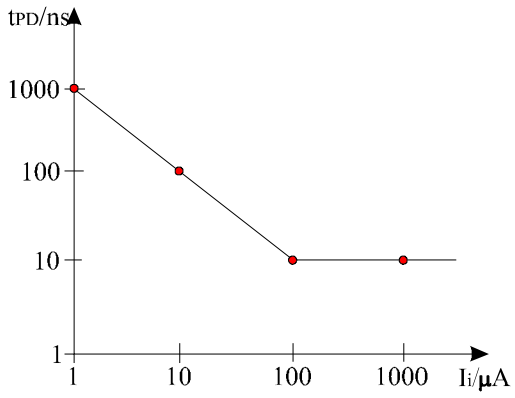


Transistor nicht in Sättigung!
→ schnelleres Umschalten der Logikpegel!

6.3.4 Integrated Injection Logik (I^2L)

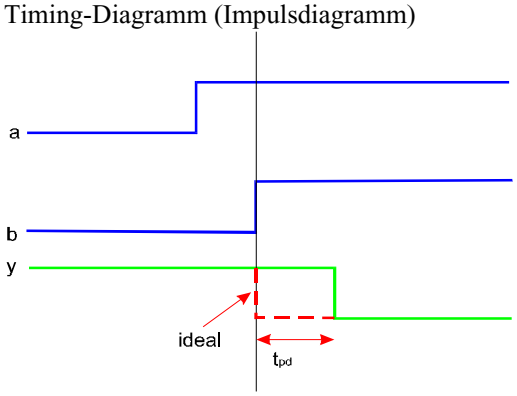
NAND:





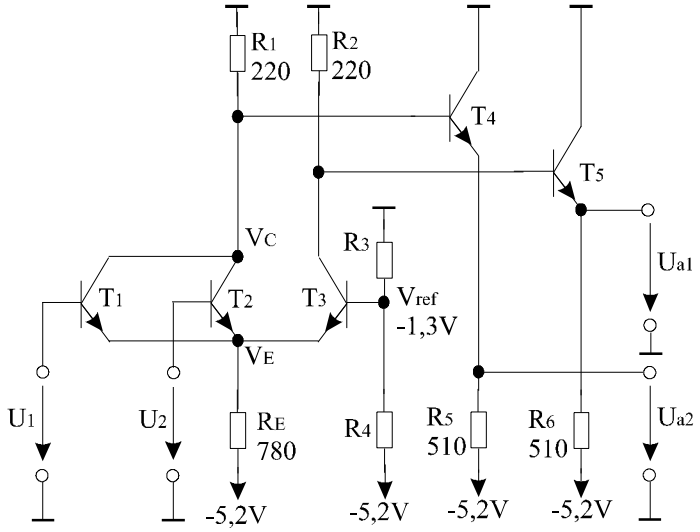
t_{pd}: Gatterlaufzeit (Propergation Delay)

Timing-Diagramm (Impulsdiagramm)



6.3.5 Emitter-Coupled-Logik (ECL)

ODER/NOR:



U_1	U_2	OR U_{a1}	NOR U_{a2}
L	L	L	H
L	H	H	L
H	L	H	L
H	H	H	L

Ausgang: L ≈ -1,55V
H ≈ -0,75V

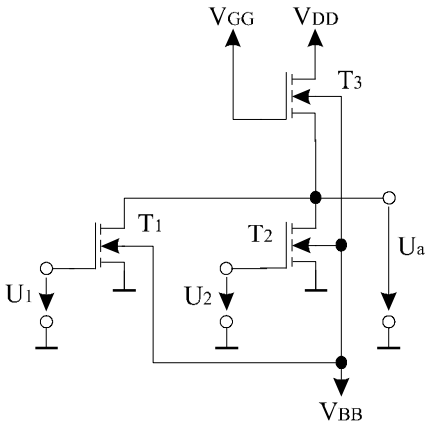
Gemeinsamer Emitterwiderstand R_E → ECL

Eigenschaften: keine Sättigung der Transistoren → kurze Schaltzeiten (t_{pd} ≈ 1ns)

6.3.6 NMOS-Logik

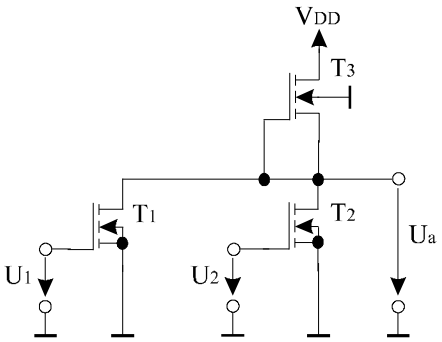
Ähnlich RTL-NOR: Kollektorwiderstand durch Transistor ersetzt!

NMOS: N-Kanal MOS-FET (selbstsperrender Typ)
MOS: Metal Oxyde Semiconductor



Typisch: V_{DD} = +5V
V_{GG} = +12V
V_{BB} = -5V
L ≈ 0V
H ≈ V_{DD}

NOR mit „depletion load“: T_1,T_2 : selbstsperrend
T_3 : selbstleitend



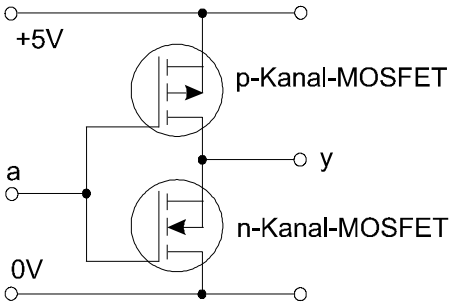
Vorteil: V_{GG}, V_{BB} entfallen!

6.3.7 CMOS-Logik

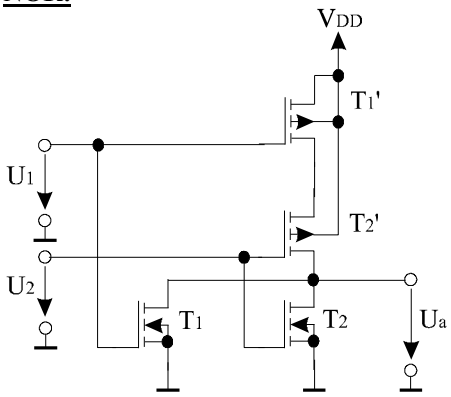
CMOS: Complementary MOS

Vorteil: geringe Stromaufnahme, da stets ein Transistor sperrt!
Problem: Stromstoß beim Umschalten! → Abblock-C!

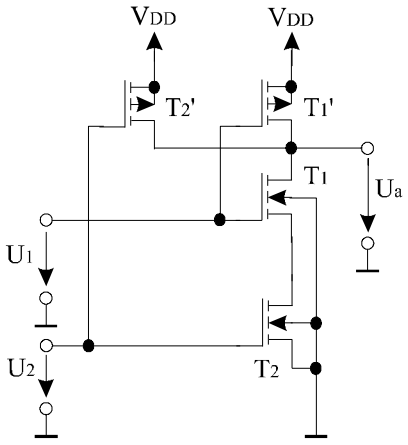
NEG:



NOR:

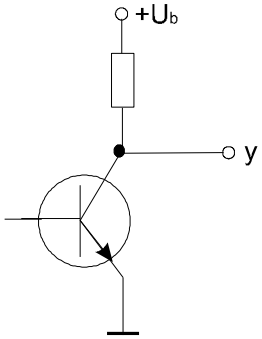


NAND:



6.4 Ausgangsschaltungen

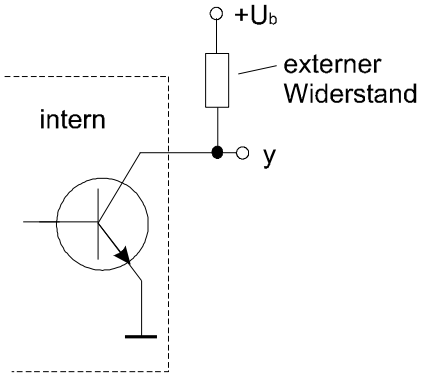
a.) mit Kollektor-Widerstand



Ausgangspegel: $V_{OL} \equiv L$
$V_{OH} \equiv H$

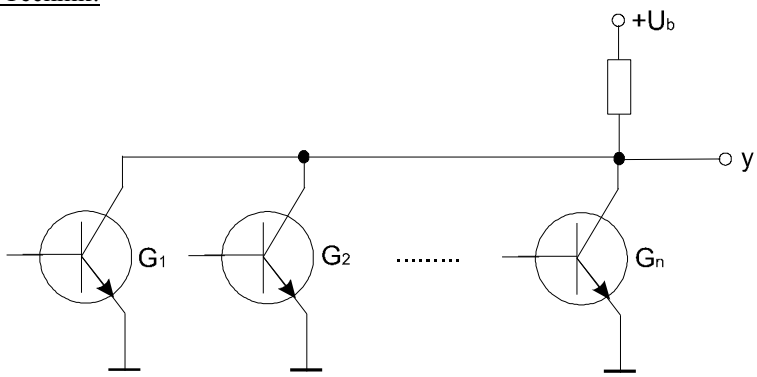
Nachteil: durch Kollektor-Widerstand!

b.) ohne Kollektor-Widerstand (Open Collector, Open Drain)



y = L: Transistor leitet
y = H: Transistor sperrt

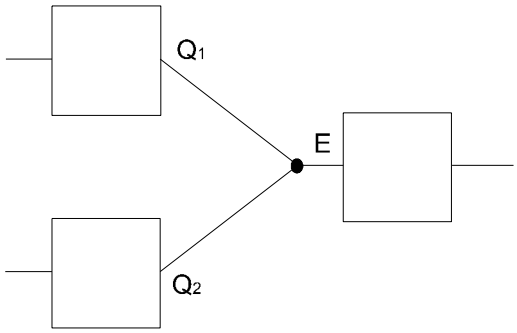
Wired-Technik:



G_1	G_2 ....G_n	y
L	L L	L
L	L .....H	L
:	: :	:
H	H L	L
H	H .....H	H

AND: Wired AND
OR: Wired OR
(pos. Logik)

allgemein:



Problem: $Q_1 \neq Q_2$ vor dem Zusammenschalten $\rightarrow$ E = ? nach dem Zusammenschalten?

H dominiert

Q_1	Q_2	E
L	L	L
L	H	H
H	L	H
H	H	H

ODER

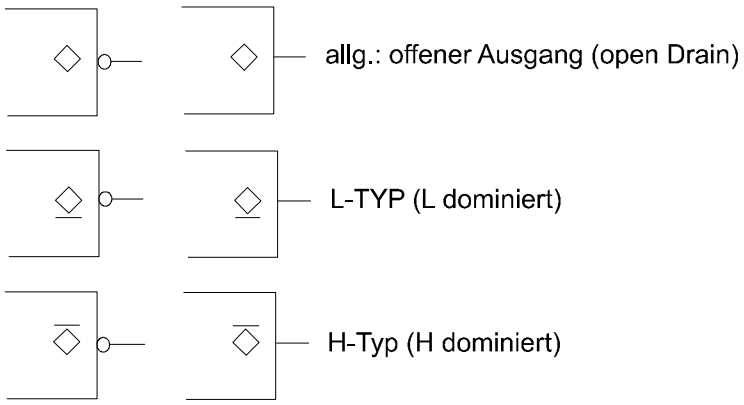
L dominiert

Q_1	Q_2	E
L	L	L
L	H	L
H	L	L
H	H	H

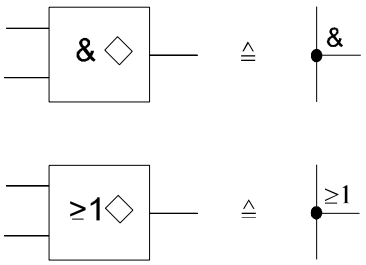
UND

Positive Logik!!

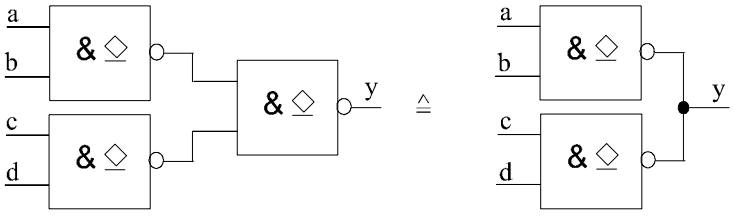
Schaltsymbole für die Ausgangsstufen:



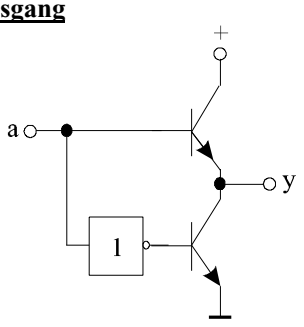
Schaltsymbole für Wired-Verknüpfungen



Beispiel:

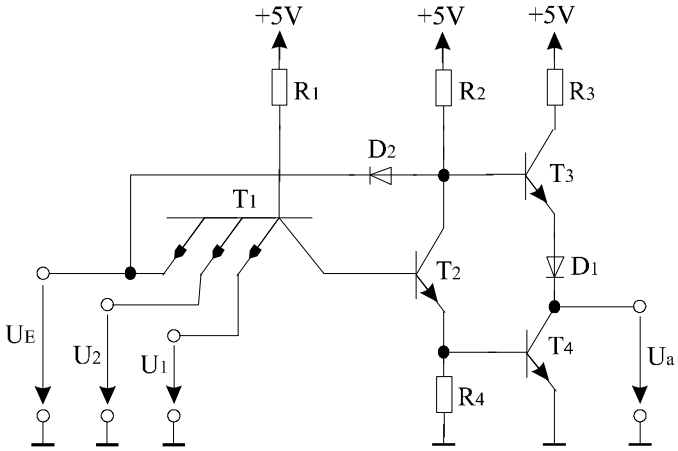


c.) Gegentaktausgang



Vorteil: Geringe Stromaufnahme, da stets ein Transistor sperrt!

d.) Tristate-Ausgang



3 Zustände: H
L
Hi Z (Hochohmig)

Anwendung: Busverbindungen (mehrere Teilnehmer an einer Leitung)

Nummernüberblick:

Nr.	Gatter-Typ	Anz. Eing./Gatter	Anz. Gatter	Bemerkung
7400	NAND	2	4	
7401	NAND	2	4	Open Drain
7402	NOR	2	4	
7403	NOR	2	4	Open Drain
7404	NEG	1	6	
7405	NEG	1	6	Open Drain
7408	AND	2	4	
7409	AND	2	4	Open Drain
7410	NAND	3	3	
7411	AND	3	3	
7414	NEG	1	6	Schmitt Trigger Input
7420	NAND	4	2	
7421	NAND	4	2	
7427	NOR	3	3	
7430	NAND	8	1	
7432	OR	2	4	
7436	NOR	2	4	
7486	EXOR	2	4	
74132	NAND	2	4	Schmitt Trigger Input
74133	NAND	13	1	
74266	NOR	2	4	
74386	EXOR	2	4	
74804	NAND	2	6	
74805	NOR	2	6	
74808	AND	2	6	
74832	OR	2	6	
4002	NOR	4	2	
4075	OR	3	3	
4078A	OR/NOR	8	1	
7001	AND	2	4	Schmitt Trigger Input
7002	NOR	2	4	Schmitt Trigger Input
7006	NAND	4	1	
7008	NAND	2	3	
7032	OR	2	4	Schmitt Trigger Input
7266	EXOR	2	4	

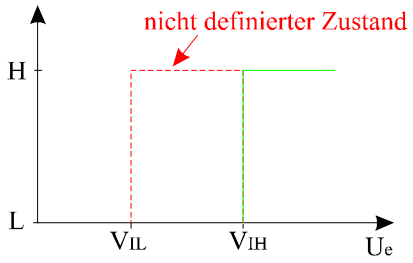
Performance Comparison of High-Speed CMOS with Several Other Logic Families

Technology ⊗	SILICON GATE CMOS	METAL GATE CMOS	STD TTL	LOW- POWER SCHOTTKY TTL	SCHOTTKY TTL	ADVANCED LOW-POWER SCHOTTKY TTL	ADVANCED SCHOTTKY TTL
Device Series	SN74HC	4000	SN74	SN74LS	SN74S	SN74ALS	SN74AS
Power dissipation per gate (mW)							
Static	0,0000025	0,001	10	2	19	1	8,5
At 100kHz	0,17	0,1	10	2	19	1	8,5
Propergation delay time (ns) (C_L=15pF)	8	105	10	10	3	4	1,5
Maximum clock frequency (MHz) (C_L=15pF)	40	12	35	40	125	70	200
Speed/Power product (pJ) (at 100kHz)	1,4	11	100	20	57	4	13
Minimum output drive (mA) (V_O=0,4V)							
Standart outputs	4	1,6	16	8	20	8	20
High-current outputs	6	1,6	48	24	64	24/48	48/64
Fan-out (LS loads)							
Standart outputs	10	4	40	20	50	20	50
High-current outputs	15	4	120	60	160	60/120	120/160
Maximum input current, I_{IL} (mA) (V_I=0,4V)	±0,001	-0,001	-1,6	-0,4	-2,0	-0,1	-0,5

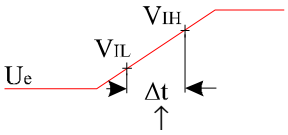
⊗ Family characteristics at 25°C, V_{CC} = 5V; all values typical unless otherwise noted. This table is provided for broad comparisions only. Parameters for specific devices within a family may vary. For detailed comparisons, please consult the appropriate data book.

6.5 Eingangsschaltungen

a.) Standart



Problem: - langsame Pegelübergänge → undefinierter Zustand über längeren Zeitraum

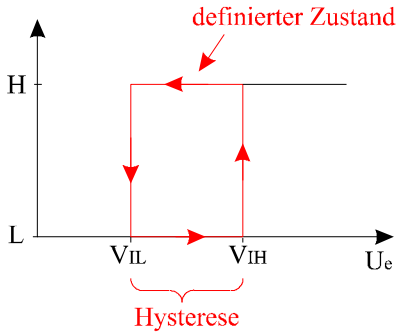


Logikpegel nicht definiert

- $U_{e(t)} = \frac{V_{iH}}{V_{iL}} + r(t)$ → Gefahr für ständigen Wechsel zwischen H und L !!

($r(t) =$ Störsignal)

b.) Schmitt-Trigger



6.6 Schaltkreisfamilien

Name	Beschreibung
ABT	Advanced BiCMOS Technology
AC/ACT	Advanced CMOS
ALS	Advanced Low Power Schottky
AS	Advanced Schottky
ALVC	Advanced Low Voltage CMOS
BCT	BiCMOS Technology
F	Fast TTL
HC/HCT	High Speed CMOS
LS	Low Power Schottky
LV	Low Voltage HCMOS
LVC	Low Voltage CMOS
LVT	Low Voltage Technology
S	Schottky TTL
TTL	Standart TTL
H	High Speed TTL

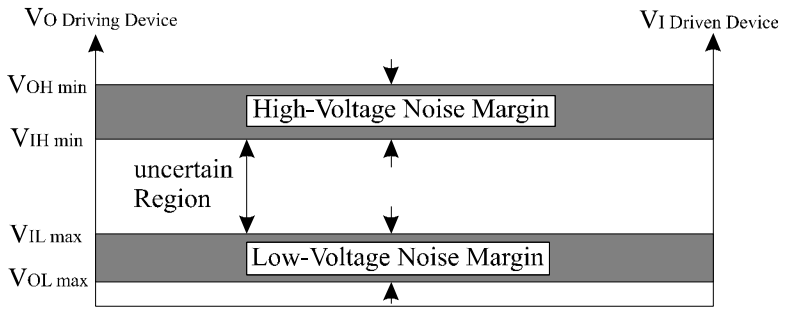
Supply Voltage:

Standart:	4,5...5,5V
Low Voltage:	2,7...3,6V

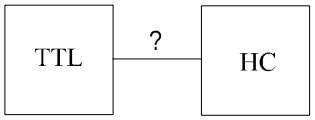
Worst-Case Values of Primary Interfacing Parameters

Parameters	74HCMOS	74TTL	74LSTTL	74ASTTL	74ALSTTL
$V_{IH\ min}$	3,5 V	2 V	2 V	2 V	2 V
$V_{IL\ max}$	1 V	0,8 V	0,8 V	0,8 V	0,8 V
$V_{OH\ min}$	4,9 V	2,4 V	2,7 V	2,7 V	2,7 V
$V_{OL\ max}$	0,1 V	0,4 V	0,4 V	0,4 V	0,4 V
$I_{IH\ max}$	1 µA	40 µA	20 µA	200 µA	20 µA
$I_{IL\ max}$	-1 µA	-1,6 mA	-400 µA	-2 mA	-100 µA
$I_{OH\ max}$	-4 mA	-400 µA	-400 µA	-2 mA	-400 µA
$I_{OL\ max}$	4 mA	16 mA	8 mA	20 mA	4 mA

Noise Margins:



6.7 Zusammenschalten verschiedener Familien

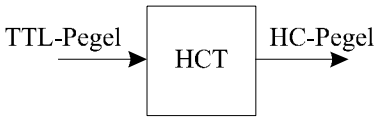


$V_{OH\ min} = 2{,}4V$ $V_{IH\ min} = 3{,}5V$
$V_{OL\ max} = 0{,}4V$ $V_{IL\ max} = 1V$

Problem: $V_{OH\ min}$ bei TTL zu gering für sichere Detektion des H-Pegels durch HC!

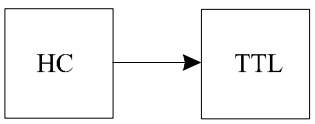
Achtung! Trotzdem oft keine Probleme bei Zimmertemperatur und geringer Last. Störungen jedoch bei Temperatur- und/oder Laständerungen!!!

Abhilfe:



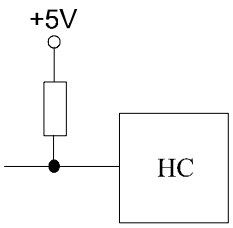
z.B.: 74HCT00

Aber:



$V_{OH\ min} = 4{,}9V$ $V_{IH\ min} = 2V$
$V_{OL\ max} = 0{,}1V$ $V_{IL\ max} = 0{,}8V$

Notlösung:



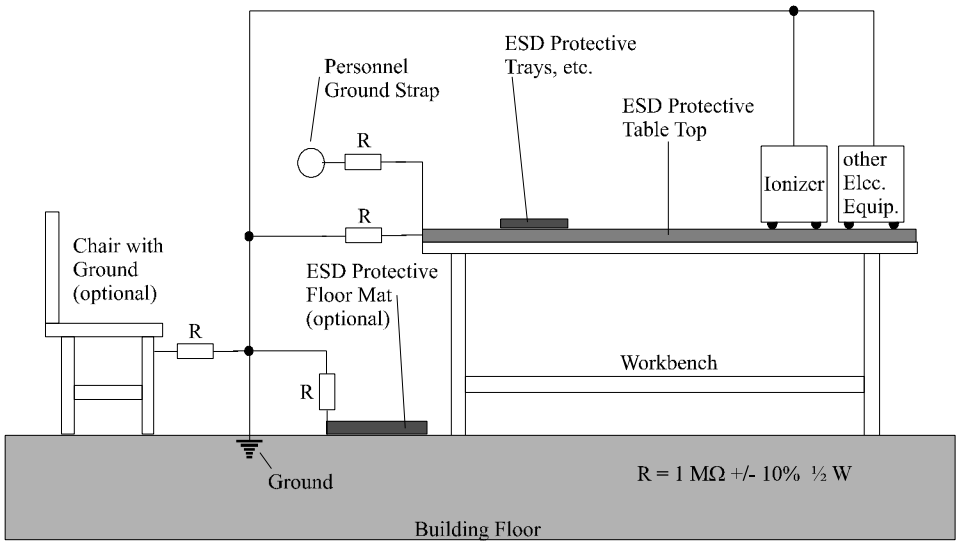
6.8 EGB-Schutzmaßnahmen

Problem: CMOS-Schaltkreise können durch elektrostatische Entladungen zerstört werden!

↓

EGB: Elektrostatisch gefährdetes Bauelement
ESD: Electrostatic Sensitive Device

Deshalb: Schutzmaßnahmen beachten! Elektrostatische Entladungen unbedingt vermeiden!



All electrical equipment sitting on the conductive table top must be hard grounded but must be isolated from the conductive table top.

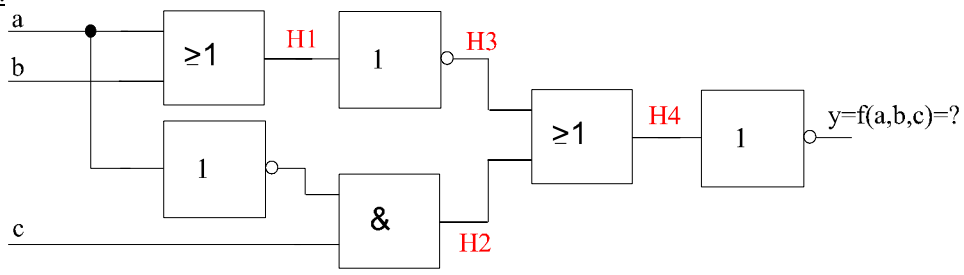
7. Analyse von Schaltnetzen

Problem: Was macht ein gegebenes Schaltnetz?

↓ Analyse

Wahrheitstabelle
DNF, KNF

Beispiel:



1.) Einfügen von Hilfsvariablen
2.) Ablesen der Schaltfunktion y:

$$y = \overline{H_4} = \overline{H_2 \vee H_3} = \overline{(\bar{a} \wedge c) \vee \overline{H_1}} = \overline{(\bar{a} \wedge c) \vee \overline{(a \vee b)}} =$$

$$= \overline{\bar{a} \wedge c} \wedge (a \vee b) = (a \vee \bar{c}) \wedge (a \vee b) = a \vee (b \wedge \bar{c})$$

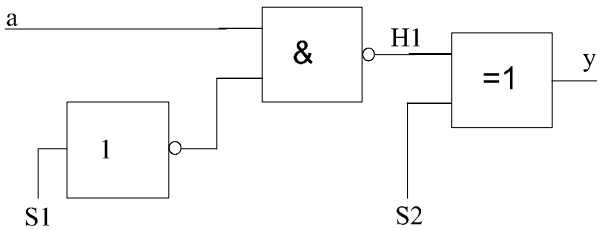
3.) Wahrheitstabelle:

c	b	a	H_1	H_2	H_3	H_4	y
0	0	0	0	0	1	1	0
0	0	1	1	0	0	0	1
0	1	0	1	0	0	0	1
0	1	1	1	0	0	0	1
1	0	0	0	1	1	1	0
1	0	1	1	0	0	0	1
1	1	0	1	1	0	1	0
1	1	1	1	0	0	0	1

4.) DNF: $y = m_1 \vee m_2 \vee m_3 \vee m_5 \vee m_7$

$$y = \overline{\overline{y}} = \overline{\overline{m_1} \wedge \overline{m_2} \wedge \overline{m_3} \wedge \overline{m_5} \wedge \overline{m_7}}$$: Realisierung mit NAND!

Sonderfall: Aufteilen in Daten- und Steuereingänge möglich
Beispiel:



$$y = S_2 \textbf{U} H_1 = S_2 \textbf{U} \overline{(a \wedge \overline{S_1})} = S_2 \textbf{U} (\bar{a} \vee S_1)$$

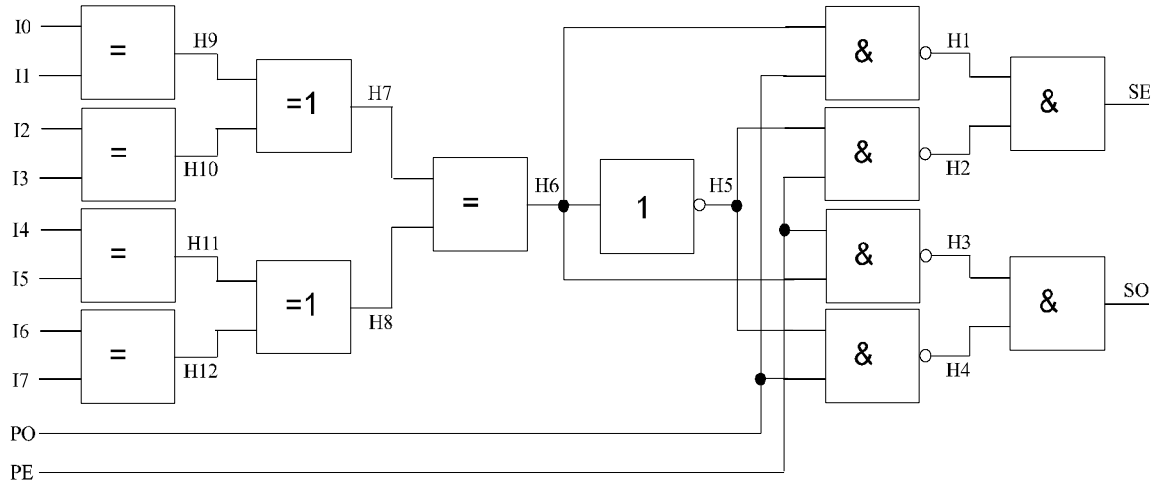
a	S_1	S_2	H_1	y
0	0	0	1	1
0	0	1	1	0
0	1	0	1	1
0	1	1	1	0
1	0	0	0	0
1	0	1	0	1
1	1	0	1	1
1	1	1	1	0

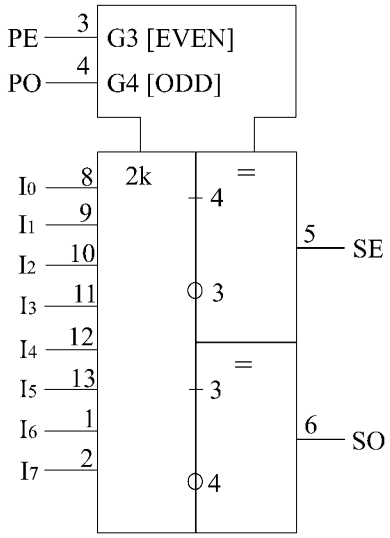
$\rightarrow$

S_1	S_2	y
0	0	$\bar{a}$
0	1	a
1	0	1
1	1	0

Übersichtlicher!

<u>Beispiel:</u> 74180 Odd/Even Parity Generator/Checker
PE, PO : Steuereingänge





Analyse: 1.) $H_6 = H_7 \leftrightarrow H_8 = (H_9 \mathbin{\mathbf{U}} H_{10}) \leftrightarrow (H_{11} \mathbin{\mathbf{U}} H_{12}) =$

$$= \underbrace{(I_0 \leftrightarrow I_1) \mathbin{\mathbf{U}} (I_2 \leftrightarrow I_3)}_{T_1} \leftrightarrow \underbrace{(I_4 \leftrightarrow I_5) \mathbin{\mathbf{U}} (I_6 \leftrightarrow I_7)}_{T_2}$$

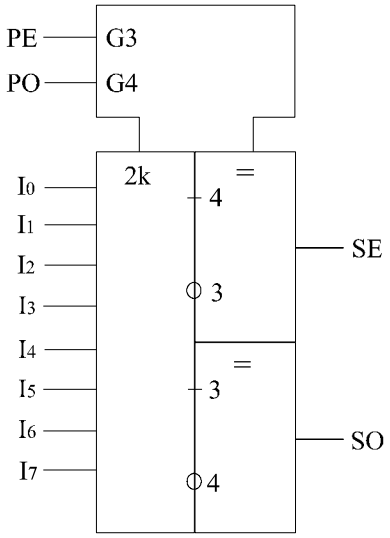
I_0	I_1	I_2	I_3	$I_0 \leftrightarrow I_1$	$I_2 \leftrightarrow I_3$	T_1
L	L	L	L	H	H	L
L	L	L	H	H	L	H
L	L	H	L	H	L	H
L	L	H	H	H	H	L
L	H	L	L	L	H	H
L	H	L	H	L	L	L
L	H	H	L	L	L	L
L	H	H	H	L	H	H
H	L	L	L	L	H	H
H	L	L	H	L	L	L
H	L	H	L	L	L	L
H	L	H	H	L	H	H
H	H	L	L	H	H	L
H	H	L	H	H	L	H
H	H	H	L	H	L	H
H	H	H	H	H	H	L

→ Anzahl H an I_0 - I_3 ungerade: $T_1 = H$
→ Anzahl H an I_0 - I_3 gerade : $T_1 = L$
→ Anzahl H an I_4 - I_7 ungerade: $T_2 = H$
→ Anzahl H an I_4 - I_7 gerade : $T_2 = L$
$H_6 = T_1 \leftrightarrow T_2$: Anzahl H an I_0 - I_7 ungerade: $H_6 = L$ ($T_1 \neq T_2$)
Anzahl H an I_0 - I_7 gerade : $H_6 = H$ ($T_1 = T_2$)

2.) $S_E = H_1 \wedge H_2 = \overline{H_6 \wedge P_O} \wedge \overline{\overline{H_6} \wedge P_E}$

$S_O = H_3 \wedge H_4 = \overline{H_6 \wedge P_E} \wedge \overline{\overline{H_6} \wedge P_O}$

PE	PO	SE	SO
L	L	H	H
L	H	$\overline{H_6}$	H_6
H	L	H_6	$\overline{H_6}$
H	H	L	L



Abhängigkeitsnotation: Bezeichnung steuernder Eingänge mit Buchstaben und Zahl
↓
G: (AND)

$$S_O = (H_6 \wedge P_E) \leftrightarrow (\overline{H_6} \wedge P_O) =$$

$$= \left[(H_6 \wedge P_E) \wedge (\overline{H_6} \wedge P_O)\right] \vee \left[\overline{(H_6 \wedge P_E)} \wedge \overline{(\overline{H_6} \wedge P_O)}\right] =$$

$$= \left[\underbrace{H_6 \wedge P_E \wedge \overline{H_6} \wedge P_O}_{=0}\right] \vee \left[\overline{(H_6 \wedge P_E)} \wedge \overline{(\overline{H_6} \wedge P_O)}\right] =$$

$$= \overline{(H_6 \wedge P_E)} \wedge \overline{(\overline{H_6} \wedge P_O)}$$

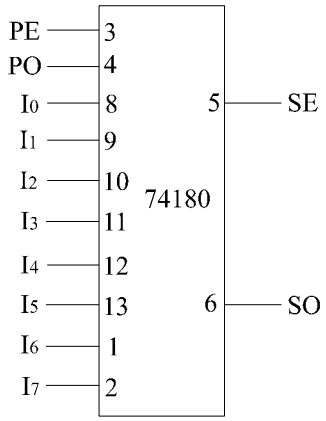
$$S_E = (H_6 \wedge P_O) \leftrightarrow (\overline{H_6} \wedge P_E) =$$

$$= \left[(H_6 \wedge P_O) \wedge (\overline{H_6} \wedge P_E)\right] \vee \left[\overline{(H_6 \wedge P_O)} \wedge \overline{(\overline{H_6} \wedge P_E)}\right] =$$

$$= \left[\underbrace{H_6 \wedge \overline{H_6} \wedge P_O \wedge P_E}_{=0}\right] \vee \left[\overline{H_6 \wedge P_O} \wedge \overline{\overline{H_6} \wedge P_E}\right] =$$

$$= \overline{H_6 \wedge P_O} \wedge \overline{\overline{H_6} \wedge P_E}$$

Praxis:



Arbeitstabelle:

Eingaben		Ausgaben
Anzahl der H-Pegel an I_0 - I_7	P_E P_O	S_E S_O
Gerade	H L	H L
Ungerade	H L	L H
Gerade	L H	L H
Ungerade	L H	H L
x	H H	L L
x	L L	H H

Synthese von Schaltnetzen

8.1 Minimalformen

$y = A \bigcirc\!\!\!\!\downarrow B$

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

DF: $y = (\bar{A} \wedge B) \vee (A \wedge \bar{B})$

KF: $\bar{y} = (\bar{A} \wedge \bar{B}) \vee (A \wedge B) \quad \rightarrow \quad y = \overline{(\bar{A} \wedge \bar{B}) \vee (A \wedge B)} = (A \wedge B) \wedge (\bar{A} \wedge \bar{B})$

Realisierung?

Annahme: $\bar{A}, \bar{B}$ vorhanden

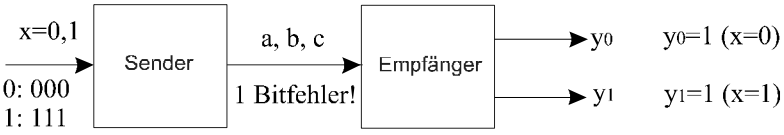
Realisierung mit NAND; $\overline{\overline{DF}} \rightarrow$ 3 Gatter

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

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

Satz: Bei der Realisierung eines Schaltnetzes läßt sich nicht allgemein angeben, ob DF oder KF zum geringsten Aufwand führen. Sucht man ein möglichst einfaches Schaltnetz, dann sind alle Möglichkeiten zu erproben und Lösungen zu vergleichen.

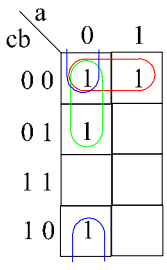
Beispiel:



Forderung: NAND-Logik (mit zwei Eingängen), a, b, c verfügbar

c	b	a	y_0	y_1
0	0	0	1	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

$y_1 = \bar{y}_0$



$$y_0 = (\bar{a} \wedge \bar{c}) \vee (\bar{b} \wedge \bar{c}) \vee (\bar{a} \wedge \bar{b})$$

$$y_0 = \bar{\bar{y}}_0 = \overline{\overline{(\bar{a} \wedge \bar{c}) \vee (\bar{b} \wedge \bar{c}) \vee (\bar{a} \wedge \bar{b})}} =$$

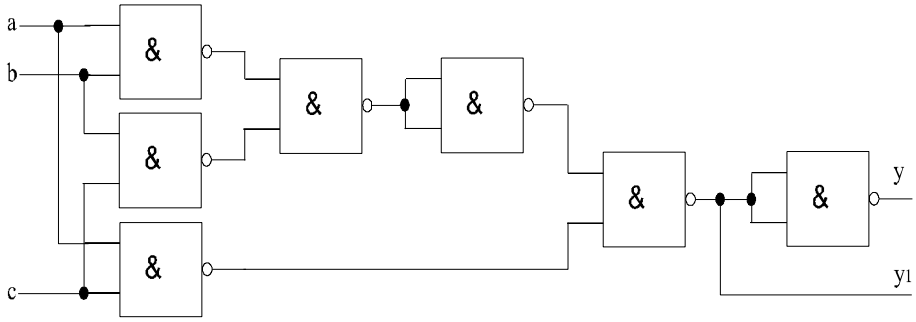
$$= \overline{\overline{(\bar{a} \wedge \bar{b})} \wedge \overline{(\bar{b} \wedge \bar{c})} \wedge \overline{(\bar{a} \wedge \bar{b})}}$$

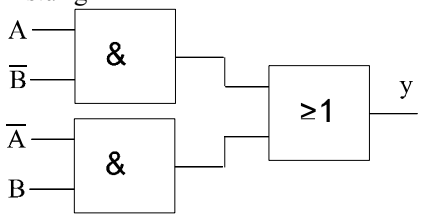
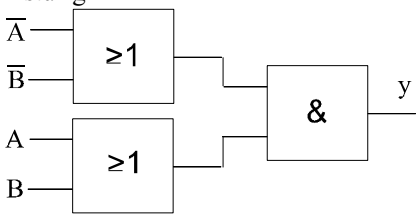
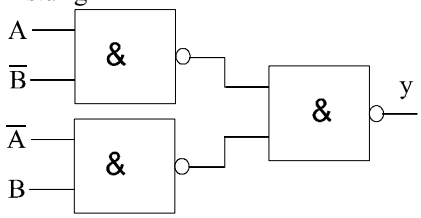
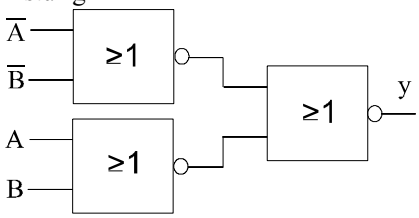
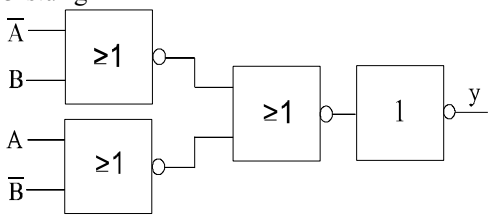
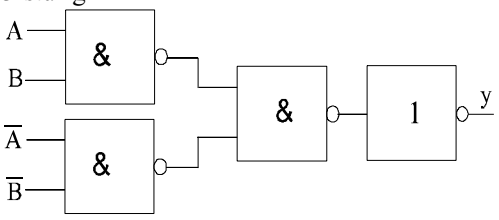
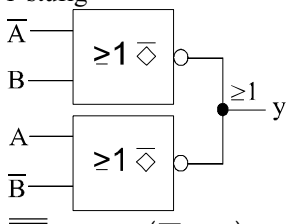
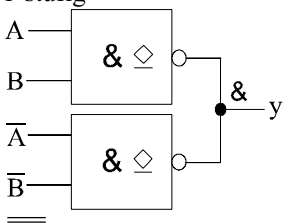
$\bar{y}_0 = y_1$: KMF: $y_0 = \overline{(a \wedge b)} \wedge \overline{(b \wedge c)} \wedge \overline{(a \wedge c)}$

$$y_0 = \bar{\bar{\bar{y}}}_0 = \overline{\overline{a \wedge b} \wedge \overline{b \wedge c} \wedge \overline{a \wedge c}}$$

Aufwand:

Typ	NAND (2 Eing.)
DMF	9 + 1 = 10 $(y_1 = \bar{y}_0)$
KMF	7 optimal!



DF Beispiel: $y = (A \wedge \overline{B}) \vee (\overline{A} \wedge B)$	KF Beispiel: $y = (\overline{A} \vee \overline{B}) \wedge (A \wedge B)$
UND/ODER 2-stufig  DF: $y = (A \wedge \overline{B}) \vee (\overline{A} \wedge B)$	ODER/UND 2-stufig  KF: $y = (\overline{A} \vee \overline{B}) \wedge (A \vee B)$
NAND/NAND 2-stufig  $\overline{\overline{DF}}$: $y = \neg(\neg(A \wedge \overline{B}) \wedge \neg(\overline{A} \wedge B))$	NOR/NOR 2-stufig  $\overline{\overline{KF}}$: $y = \neg(\neg(\overline{A} \vee \overline{B}) \vee \neg(A \vee B))$
NOR/NOR/INV 3-stufig  $\overline{\overline{\overline{DF}}}$: $y = \neg(\neg(\neg(\overline{A} \vee B) \vee \neg(A \vee \overline{B})))$	NAND/NAND/INV 3-stufig  $\overline{\overline{\overline{KF}}}$: $y = \neg(\neg(\neg(A \wedge B) \wedge \neg(\overline{A} \wedge \overline{B})))$
Wired-OR 1-stufig  $\overline{\overline{DF}}$: $y = \neg(\overline{A} \vee B) \vee \neg(A \vee \overline{B})$	Wired-AND 1-stufig  $\overline{\overline{KF}}$: $y = \neg(A \wedge B) \wedge \neg(\overline{A} \wedge \overline{B})$

8.2 Häufig benutzte Schaltnetze

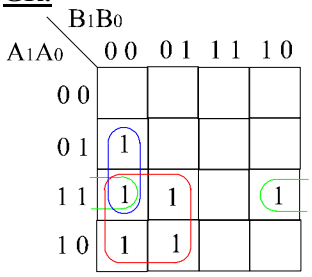
8.2.1 Vergleicher (Komparator)

Aufgabe: Vergleich zweier Dualzahlen auf größer, kleiner, gleich

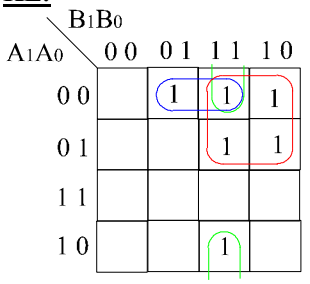
A > B : GR = 1
A < B : KL = 1
A = B : GL = 1

A_1	A_0	B_1	B_0	GR	KL	GL
0	0	0	0	0	0	1
0	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	0	1	0
0	1	0	0	1	0	0
0	1	0	1	0	0	1
0	1	1	0	0	1	0
0	1	1	1	0	1	0
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	0	1
1	0	1	1	0	1	0
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	0	1

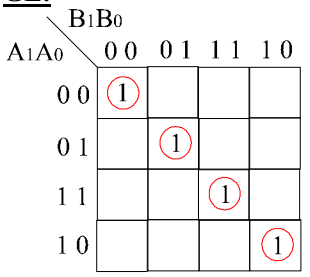
GR:



KL:



GL:

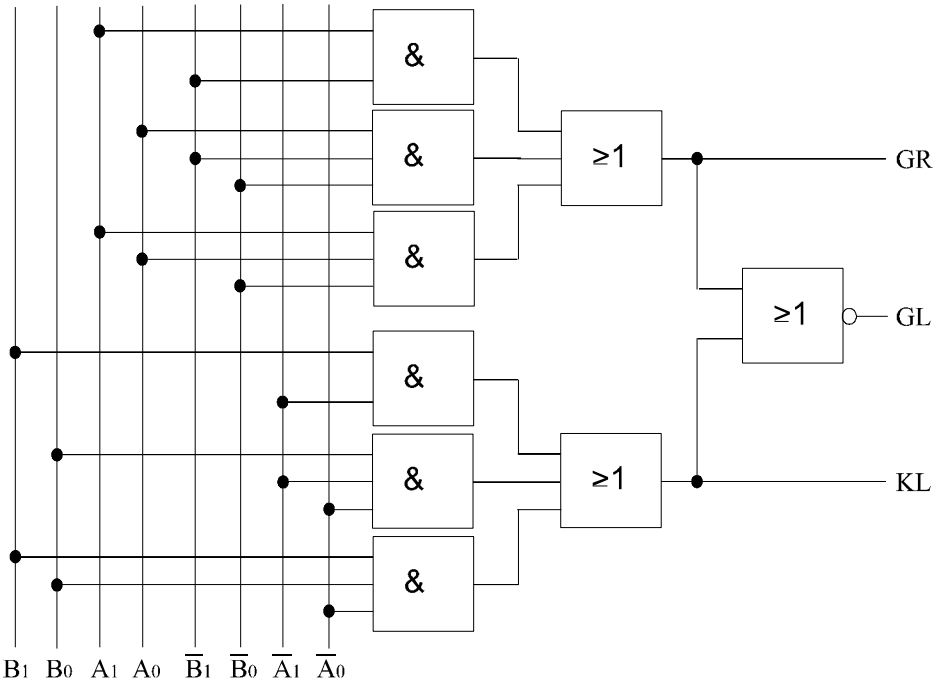


$$GR = \left(\overline{B_1} \wedge A_1\right) \vee \left(\overline{B_0} \wedge A_1 \wedge A_0\right) \vee \left(\overline{B_1} \wedge \overline{B_0} \wedge A_0\right)$$

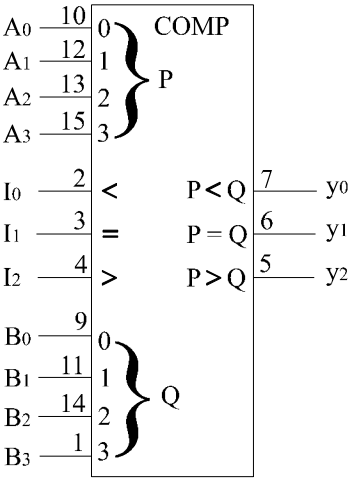
$$KL = \left(\overline{A_1} \wedge B_1\right) \vee \left(\overline{A_0} \wedge \overline{A_1} \wedge B_0\right) \vee \left(\overline{A_0} \wedge B_0 \wedge B_1\right)$$

$$GL = \left(\overline{A_1} \wedge \overline{A_0} \wedge \overline{B_1} \wedge \overline{B_0}\right) \vee \left(\overline{A_1} \wedge A_0 \wedge \overline{B_1} \wedge B_0\right) \vee \left(A_1 \wedge \overline{A_0} \wedge B_1 \wedge \overline{B_0}\right) \vee \left(A_1 \wedge A_0 \wedge B_1 \wedge B_0\right) =$$

$$= \overline{GR} \wedge \overline{KL} = \overline{GR \vee KL}$$



IC: SN 7485



A (A_0 - A_3)
B (B_0 - B_3)

y_0 = 1 wenn A < B
y_1 = 1 wenn A = B
y_2 = 1 wenn A > B

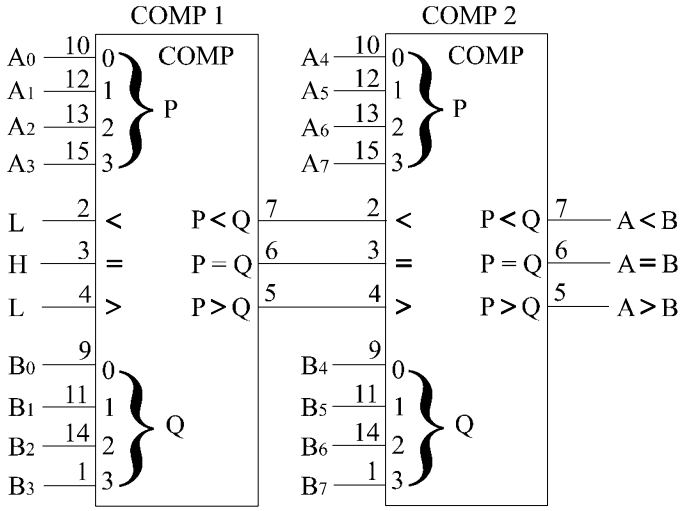
A_3, B_3	A_2, B_2	A_1, B_1	A_0, B_0	I_2	I_1	I_0	y_2	y_1	y_0
$A_3 > B_3$	x	x	x	x	x	x	H	L	L
$A_3 < B_3$	x	x	x	x	x	x	L	L	H
$A_3 = B_3$	$A_2 > B_2$	x	x	x	x	x	H	L	L
$A_3 = B_3$	$A_2 < B_2$	x	x	x	x	x	L	L	H
$A_3 = B_3$	$A_2 = B_2$	$A_1 > B_1$	x	x	x	x	H	L	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 < B_1$	x	x	x	x	L	L	H
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 > B_0$	H	L	L	H	L	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 < B_0$	L	L	H	L	L	H
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	L	H	L	L	H	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	x	H	x	L	H	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	H	L	H	L	L	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	L	L	L	H	L	H

		GR y_2	GL y_1	KL y_0	
$A_3 > B_3$	→	H	L	L	
:					
:					
$A_3 = B_3$	→ Vergleich A_2, B_2				1.Stufe: $I_0 = L$
$A_2 > B_2$	→	H	L	L	$I_1 = H$
:					$I_2 = L$
:					
$A_i = B_i$	$y_i = I_i$				

↑
vorhergehende
Vergleichsstufe

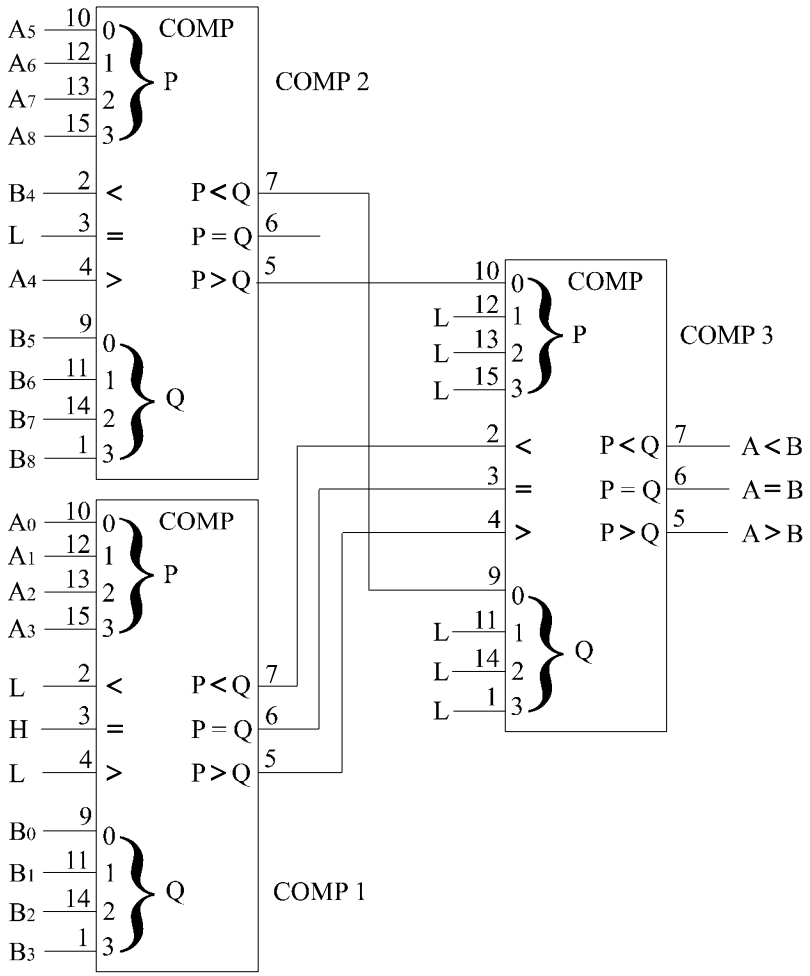
Vergleich 8-stelliger Dualzahlen:

- Serien-Erweiterung:



2 x SN7485

- Parallel-Erweiterung:

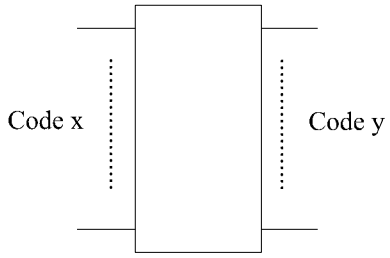


COMP 1:	COMP 2:
P = Q	$y_i = I_i$ (Resultat COMP 1)
P > Q	A > B
P < Q	A < B

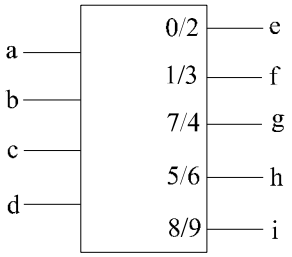
Vergleich von Zahlen mit mehr als 8 Bit: Bei Parallel-Erweiterung geringere Laufzeiten !

8.2.2 Code-Umsetzer

Aufgabe: Code x → Code y

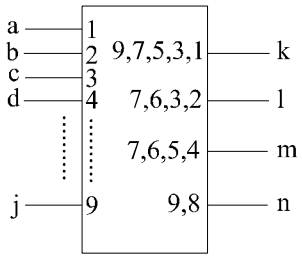


Beispiel: 8-4-2-1-Code (BCD) → Code y



n = 1 für BCD = 5, 6

Beispiel: Dezimal 1 aus 10 → BCD



Beispiel:

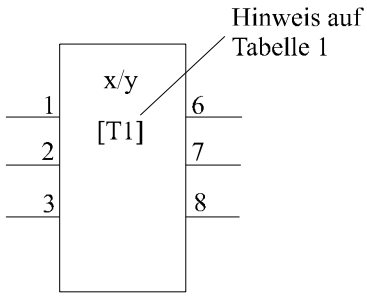
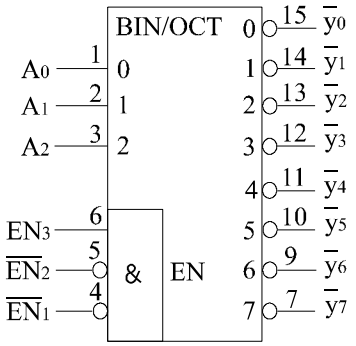


Tabelle 1:

Eingang			Ausgang		
3	2	1	8	7	6
0	0	0	1	0	1
0	0	1	0	0	1
0	1	0	0	1	1
	:			:	
	:			:	

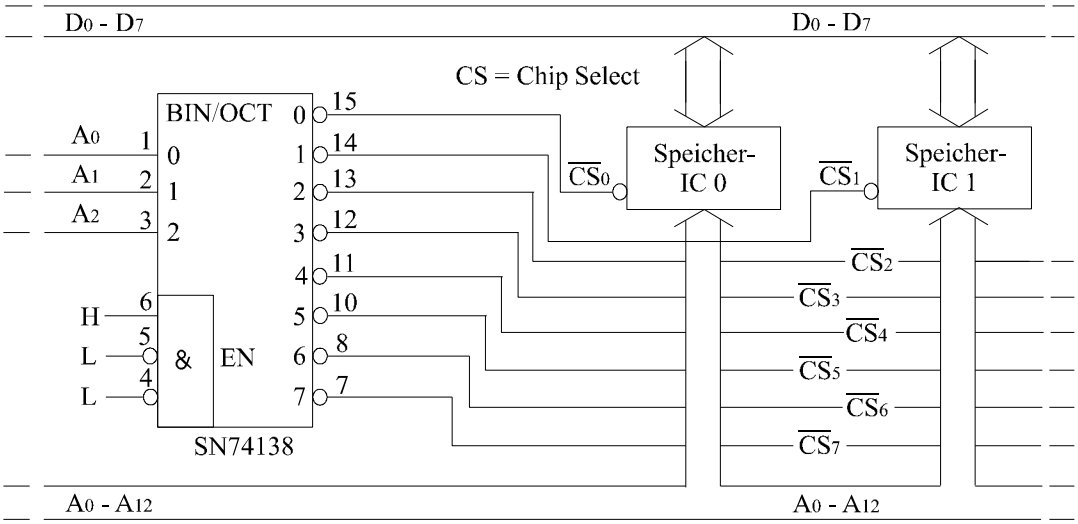
Beispiel: SN 74138 Dualzahl (3 Stellen) → 1 aus 8 - Code



Arbeitstabelle:

Eingänge						Ausgänge							
$\overline{EN_1}$	$\overline{EN_2}$	EN_3	A_0	A_1	A_2	$\overline{y_0}$	$\overline{y_1}$	$\overline{y_2}$	$\overline{y_3}$	$\overline{y_4}$	$\overline{y_5}$	$\overline{y_6}$	$\overline{y_7}$
H	x	x	x	x	x	H	H	H	H	H	H	H	H
x	H	x	x	x	x	H	H	H	H	H	H	H	H
x	x	L	x	x	x	H	H	H	H	H	H	H	H
L	L	H	L	L	L	L	H	H	H	H	H	H	H
L	L	H	H	L	L	H	L	H	H	H	H	H	H
L	L	H	L	H	L	H	H	L	H	H	H	H	H
L	L	H	H	H	L	H	H	H	L	H	H	H	H
L	L	H	L	L	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	L	H	H
L	L	H	L	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	L

Anwendung:

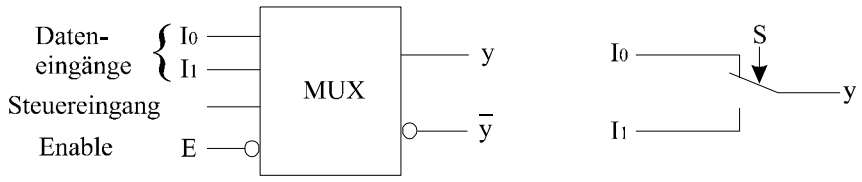


Auszug aus einem Mikroprozessor-System

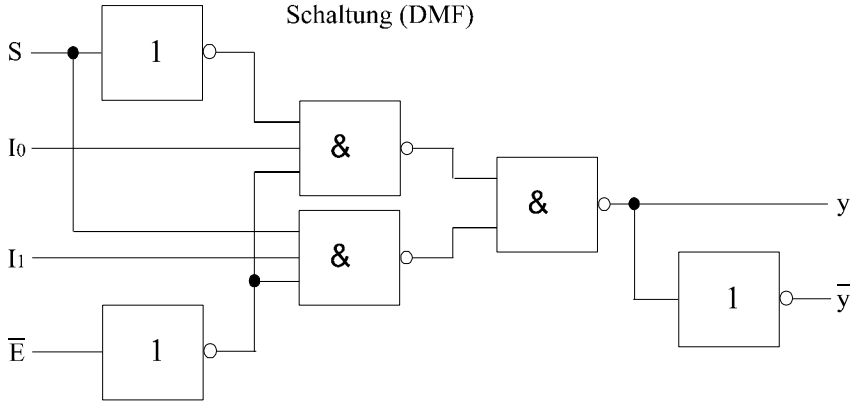
8.2.3 Multiplexer

DIN 44300: Funktionseinheit, die Nachrichten von Kanälen einer Anzahl an Kanäle einer anderen Anzahl übergibt.

Beispiel:

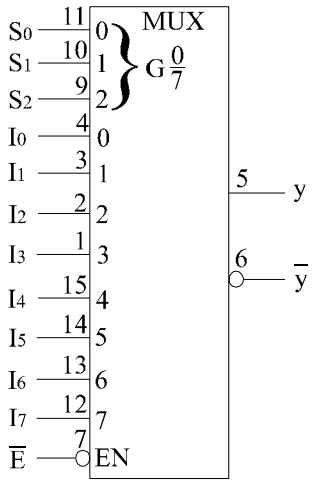


$\overline{E}$	S	I_0	I_1	y	$\overline{y}$	
1	x	x	x	0	1	
0	0	0	x	0	1	I_0 durchschalten
0	0	1	x	1	0	I_0 durchschalten
0	1	x	0	0	1	I_1 durchschalten
0	1	x	1	1	0	I_1 durchschalten



Entwurf eines Multiplexers mit zwei Eingängen

Beispiel: SN 74151 8 Dateneingänge



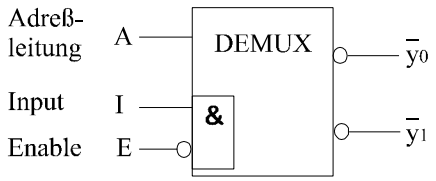
Eingänge												Ausgänge	
$\overline{E}$	S_2	S_1	S_0	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	y	$\overline{y}$
H	x	x	x	x	x	x	x	x	x	x	x	L	H
L	L	L	L	L	x	x	x	x	x	x	x	L	H
L	L	L	L	H	x	x	x	x	x	x	x	H	L
L	L	L	H	x	L	x	x	x	x	x	x	L	H
L	L	L	H	x	H	x	x	x	x	x	x	H	L
L	L	H	L	x	x	L	x	x	x	x	x	L	H
L	L	H	L	x	x	H	x	x	x	x	x	H	L
L	L	H	H	x	x	x	L	x	x	x	x	L	H
L	L	H	H	x	x	x	H	x	x	x	x	H	L
L	H	L	L	x	x	x	x	L	x	x	x	L	H
L	H	L	L	x	x	x	x	H	x	x	x	H	L
L	H	L	H	x	x	x	x	x	L	x	x	L	H
L	H	L	H	x	x	x	x	x	H	x	x	H	L
L	H	H	L	x	x	x	x	x	x	L	x	L	H
L	H	H	L	x	x	x	x	x	x	H	x	H	L
L	H	H	H	x	x	x	x	x	x	x	L	L	H
L	H	H	H	x	x	x	x	x	x	x	H	H	L

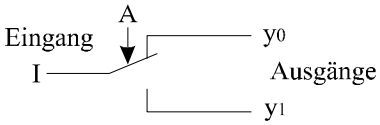
Auszug aus dem Datenblatt zum TTL-IC SN74151

8.2.4 Demultiplexer

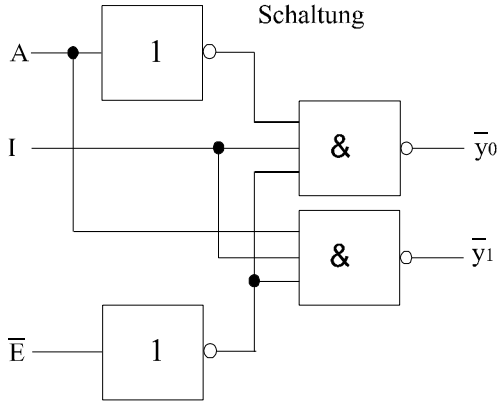
MUX: mehrere Eingänge → ein Ausgang
DEMUX: ein Eingang → mehrere Ausgänge

Beispiel:



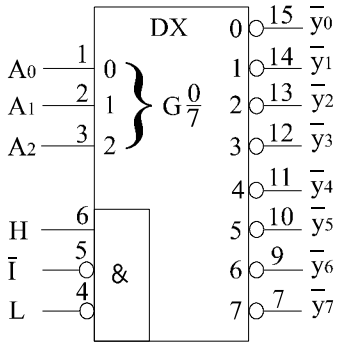


$\overline{E}$	A	I	$\overline{y}_0$	$\overline{y}_1$	
1	x	x	1	1	
0	0	0	1	1	$\overline{y_0} = \overline{I}$
0	0	1	0	1	
0	1	0	1	1	$\overline{y_1} = \overline{I}$
0	1	1	1	0	



Entwurf eines Demultiplexers mit zwei Ausgängen

Beispiel: SN 74138 (1 Eingang, 8 Ausgänge)

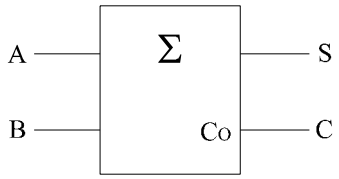


Eingänge				Ausgänge							
$\bar{I}$	A_0	A_1	A_2	$\overline{y_0}$	$\overline{y_1}$	$\overline{y_2}$	$\overline{y_3}$	$\overline{y_4}$	$\overline{y_5}$	$\overline{y_6}$	$\overline{y_7}$
H	x	x	x	H	H	H	H	H	H	H	H
L	L	L	L	L	H	H	H	H	H	H	H
L	H	L	L	H	L	H	H	H	H	H	H
L	L	H	L	H	H	L	H	H	H	H	H
L	H	H	L	H	H	H	L	H	H	H	H
L	L	L	H	H	H	H	H	L	H	H	H
L	H	L	H	H	H	H	H	H	L	H	H
L	L	H	H	H	H	H	H	H	H	L	H
L	H	H	H	H	H	H	H	H	H	H	L

TTL-IC SN74138 beim Einsatz als Demultiplexer

8.2.5 Rechenschaltungen

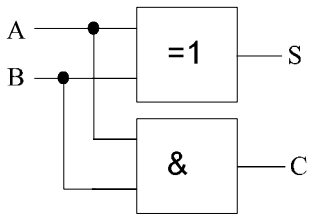
Halbaddierer: Kein Eingang für Übertrag von vorhergehender Stufe



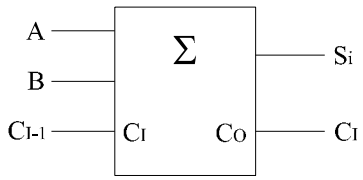
S: Summe
C: Carry (Übertrag)
C_O: Carry out

$$\begin{array}{r} A \\ +\ B \\ \hline C\ S \end{array}$$

B	A	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



Volladdierer: Mit Eingang für Übertrag von vorhergehender Stufe

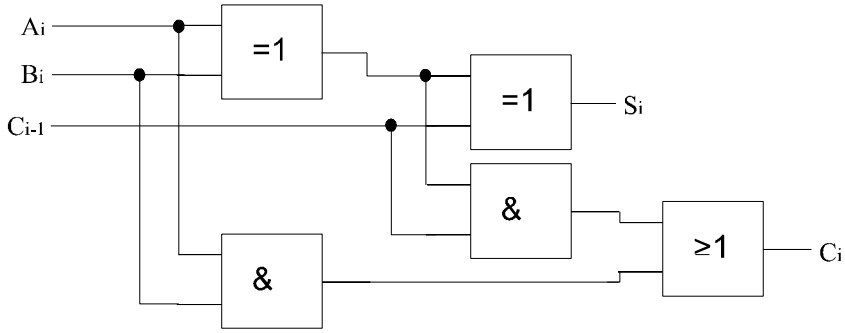


$$\begin{array}{r} A_i \\ B_i \\ +\ C_{i-1} \\ \hline C_i\ S_i \end{array}$$

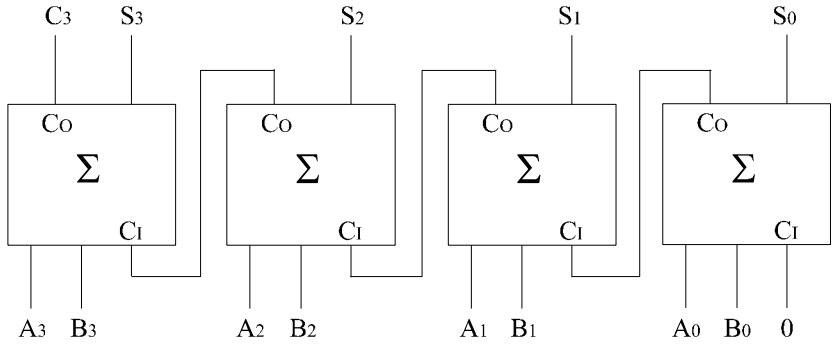
C_{i-1}	B_i	A_i	S_i	C_i
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$S_i = A_i \leftrightarrow B_i \leftrightarrow C_i$

$C_i = (A_i \wedge B_i) \vee (C_{i-1} \wedge (A_i \uparrow B_i))$



4 Bit-Volladdierer:



Serienübertrag ! Carry muß sukzessive alle Stufen durchlaufen !

$$t_{dc} = n \cdot t_{dcs}$$

t_{dc} : Time Delay Carry (Carry-Verzugszeit insgesamt)

t_{dcs} : Time Delay Carry Single (Carry-Verzugszeit pro Stufe)

3 Gatter zwischen C_{i-1} und C_i $\rightarrow$ $t_{dcs} \approx 30$ ns ($t_{pd} = 10$ ns)

$$t_{dc} = n \cdot 30ns$$

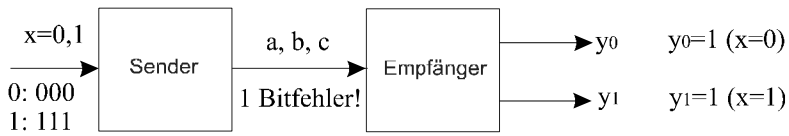
8.3 Normalformen

Bisher: Realisierung der minimalen Schaltfunktion

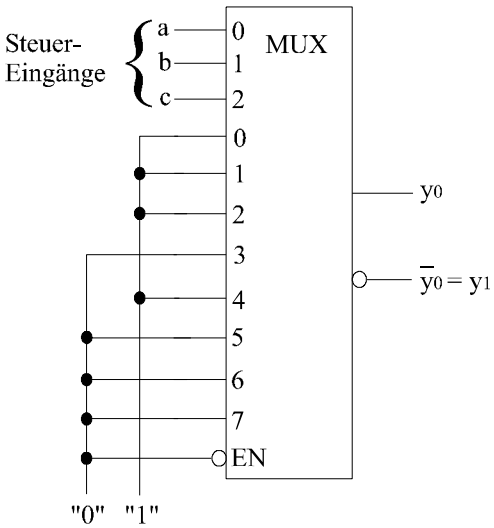
Jetzt: Realisierung ohne Minimierung !

8.3.1 Realisierung mit MUX

Beispiel:



Basis: MUX mit 8 Eingängen (3 Steuereingänge)



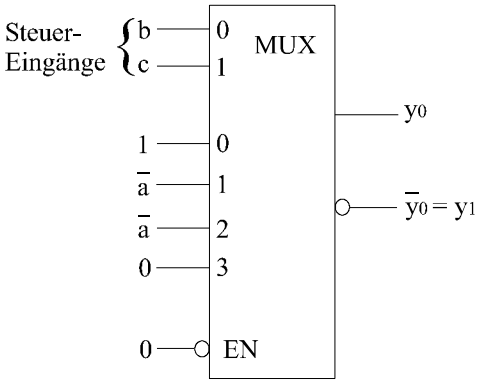
c	b	a	y_0	$y_1 = \overline{y_0}$
0	0	0	1	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

Anzahl der MUX-Steuereingänge = Anzahl der Eingangsvariablen
i-te Zeile der WT ≡ i-tem Eingang (Beschaltung mit dem in der WT geforderten Zustand)
(i = 0, ... 7)

Voraussetzung für dieses Vorgehen: Anzahl der MUX-Steuereingänge ≥ Anzahl der Eingangsvariablen

Beispiel: Nochmals: Sender - Empfänger (s.S.)

Basis: MUX mit 4 Eingängen (2 Steuereingänge)



c	b	a	y_0
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

pro Eingang gibt es vier verschiedene Fälle

a	y_0			
0	1	0	0	1
1	1	0	1	0
Eing.	1	0	a	$\overline{a}$

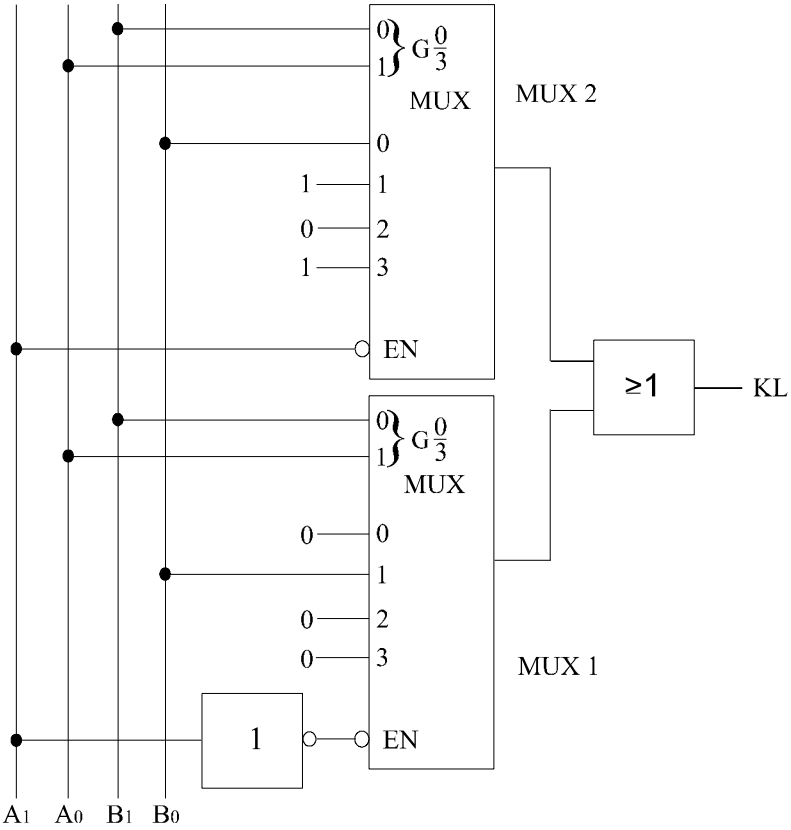
Satz: Mit einem MUX mit n Steuereingängen läßt sich jede Schaltfunktion mit n + 1 Eingangsvariablen realisieren !

Problem: Anzahl der Eingangsvariablen > Anzahl der Steuereingänge + 1

↓

Verwendung mehrerer MUX

Beispiel: Vergleicher für zwei zweistellige Dualzahlen
$A = (A_1, A_0)$
$B = (B_1, B_0)$
KL = 1, wenn A < B



Basis: 2 MUX mit je 2 Steuereingängen

Vorgehen: Obiges Verfahren + Ein-/Ausschalten der MUX mit Enable + Oder-Verknüpfung der MUX-Ausgänge

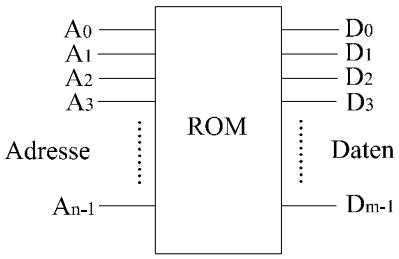
Satz: Mit 2 MUX mit je n Steuereingängen läßt sich jede Schaltfunktion mit n + 2 Eingangsvariablen realisieren, wenn die MUX voneinander unabhängige Enable-Eingänge besitzen.

Vorteil der MUX-Realisierung: Verwendung hochintegrierter Bausteine → hohe Zuverlässigkeit, Platzersparnis, wenige Bausteintypen !

8.3.2 Realisierung mit ROM

ROM = Read only Memory (nichtflüchtiger Speicher, Festwertspeicher)
PROM = Programmable ROM
EPROM = Eraseable PROM

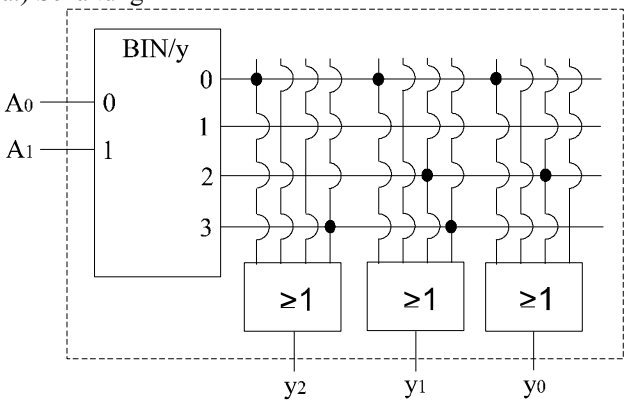
Beispiel: ROM-BIOS (IBM-PC)



2^n Adressen → 2^m Datenkombinationen mit je m Bit → Speicherinhalt: $2^n \cdot m$ Bit

Nachbildung eines ROM:

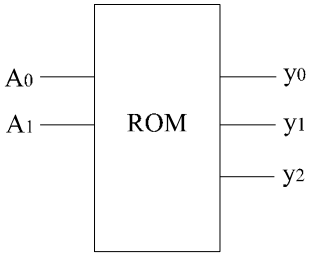
a.) Schaltung



b.) Wertetabelle

Nr.	A_1	A_0	y_2	y_1	y_0
0	0	0	1	1	1
1	0	1	0	0	0
2	1	0	0	1	1
3	1	1	1	1	0

Beispiel:

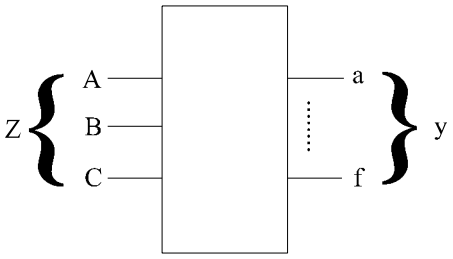


`Realisierung´ mit : Code-Umsetzer Dualcode → 1-aus-4-Code + Speichermatrix

Programmierung: Herstellen bzw. Löschen der Verbindungen an den Kreuzungspunkten.

Achtung !! Speicherinhalt bleibt nach Abschalten der Spannungsversorgung erhalten !!

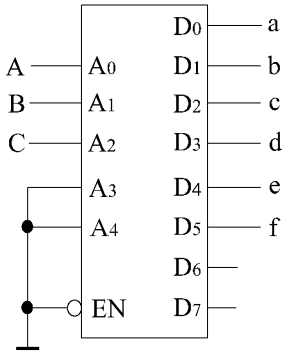
Beispiel:



Z : Dualzahl
$Z \leq 6 : y = Z^2 + 1$
$Z = 7 : y = Z^2 + 9$

	Z			f	e	d	c	b	a	
	c	b	a							
0	0	0	0	0	0	0	0	0	1	1
1	0	0	1	0	0	0	0	1	0	2
2	0	1	0	0	0	0	1	0	1	5
3	0	1	1	0	0	1	0	1	0	A_H
4	1	0	0	0	1	0	0	0	1	11_H
5	1	0	1	0	1	1	0	1	0	$1A_H$
6	1	1	0	1	0	0	1	0	1	25_H
7	1	1	1	1	1	1	0	1	0	$3A_H$
	Adresse			Data						

PROM: TBP18S030 (32x8 Bit)

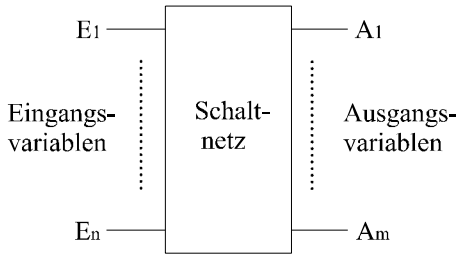


9. Schaltwerke

9.1 Einführung

9.1.1 Schaltnetze und Schaltwerke

Schaltnetz:



$$\underline{E} = \begin{bmatrix} E_1 \\ : \\ E_n \end{bmatrix} \underline{A} = f(\underline{E})\underline{A} = \begin{bmatrix} A_1 \\ : \\ A_m \end{bmatrix}$$

$$\underline{E} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \rightarrow \underline{A} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

Satz: Bei einem Schaltnetz hängt der Ausgangszustand nur vom gegenwärtigen Eingangszustand ab. Die Vorgeschichte ist ohne Bedeutung. Für jeden Eingangszustand gibt es einen Ausgangszustand.

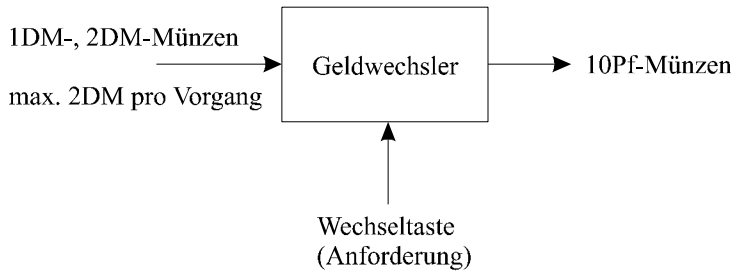
Schaltwerk: $\underline{A}_{(tn)} = f\left[\underline{E}_{(tn)}, \underline{E}_{(tn-1)}, \ldots, \underline{E}_{(t0)}\right] \quad t_i = \text{Zeitpunkt}$

Satz: Bei einem Schaltwerk hängt der Ausgangszustand von bisherigen Eingangszuständen ab. Die Vorgeschichte ist von Bedeutung. Das Schaltwerk hat ein Gedächtnis. Bei einem Schaltwerk ist für jeden Eingangszustand mehr als ein Ausgangszustand möglich.

Häufige Bezeichnung: Schaltnetz = kombinatorische Logik
Schaltwerk = sequenzielle Logik

9.1.2 Zustandsdiagramm und Zustandsfolgetabelle

Beispiel:



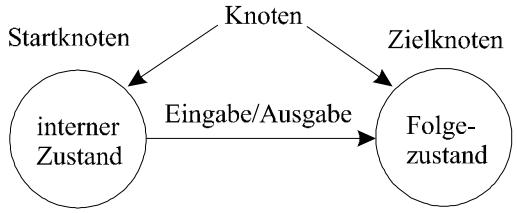
Automat muß sich die eingeworfenen Münzen merken (Vorgeschichte) !
→ interner Zustand

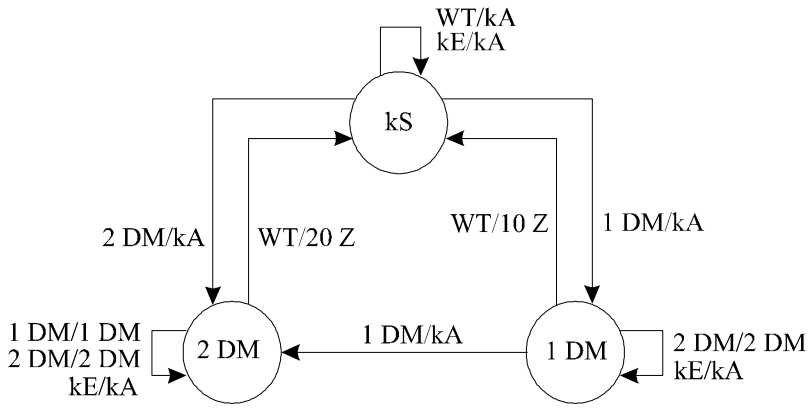
↓ Eingabe

Folgezustand

Eingaben:	1 DM 2 DM Wechseltaste (WT) keine Eingabe (kE)
Ausgaben:	10 Z (10x10 Pfg.) 20 Z (20x10 Pfg.) 1 DM (1 DM zurück) 2 DM (2 DM zurück) } wenn mehr als 2 DM eingeworfen wurden! Keine Ausgabe (kA)
Zustände:	1 DM (1 DM wurde eingegeben) 2 DM (2 DM wurden eingegeben) keine Schulden (kS)

Zustandsdiagramm:





Satz: Zur eindeutigen Beschreibung eines Automaten sind neben Eingaben und Ausgaben auch die internen Zustände erforderlich. Der Automat ist vollständig bestimmt, wenn Folgezustand und Ausgabe für jeden internen Zustand bei jeder möglichen Eingabe festliegt.

Zustandsfolgetabelle: Tabellarische Darstellung der Zustände

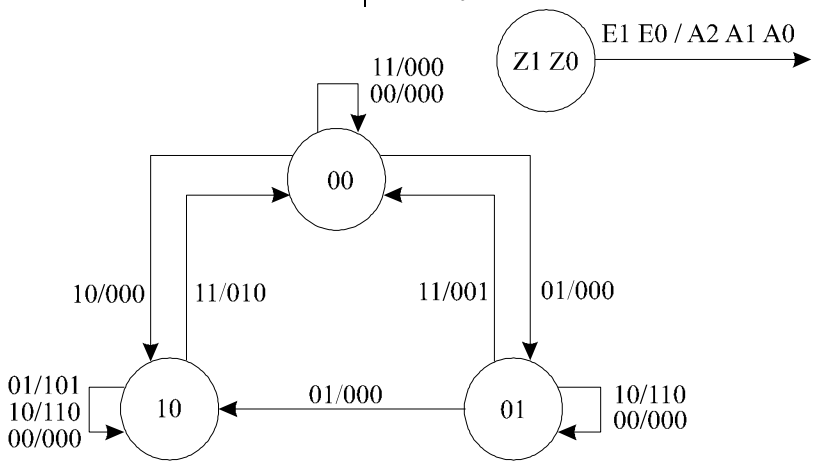
Eingabe	Zustand	Folgezustand	Ausgabe
kE	kS	kS	kA
kE	1 DM	1 DM	kA
kE	2 DM	2 DM	kA
1 DM	kS	1 DM	kA
1 DM	1 DM	2 DM	kA
1 DM	2 DM	2 DM	1 DM
2 DM	kS	2 DM	kA
2 DM	1 DM	1 DM	2 DM
2 DM	2 DM	2 DM	2 DM
WT	kS	kS	kA
WT	1 DM	kS	10 Z
WT	2 DM	kS	20 Z

Codierung: 4 Eingaben → 2 Bit
5 Ausgaben → 3 Bit
3 Zustände → 2 Bit

Eingabe	E_1	E_0
kE	0	0
1 DM	0	1
2 DM	1	0
WT	1	1

Ausgabe	A_2	A_1	A_0
kA	0	0	0
10 Z	0	0	1
20 Z	0	1	0
1 DM	1	0	1
2 DM	1	1	0

Int. Zustand	Z_1	Z_0
kS	0	0
1 DM	0	1
2 DM	1	0



Zustandsdiagramm zum Schaltwerk des Geldwechselautomaten

E_1	E_0	Z_1	Z_0	Z_1^+	Z_0^+	A_2	A_1	A_0
0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0
0	0	1	0	1	0	0	0	0
0	0	1	1	x	x	x	x	x
0	1	0	0	0	1	0	0	0
0	1	0	1	1	0	0	0	0
0	1	1	0	1	0	1	0	1
0	1	1	1	x	x	x	x	x
1	0	0	0	1	0	0	0	0
1	0	0	1	0	1	1	1	0
1	0	1	0	1	0	1	1	0
1	0	1	1	x	x	x	x	x
1	1	0	0	0	0	0	0	0
1	1	0	1	0	0	0	0	1
1	1	1	0	0	0	0	1	0
1	1	1	1	x	x	x	x	x

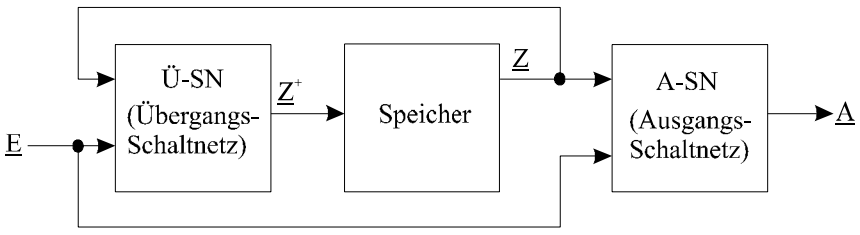
Zustandsfolgetabelle zum Schaltwerk des Geldwechselautomaten

Z_i : Zustand
Z_i^+: Folgezustand
Pseudozustand: Zustand kann nicht eintreten ! $\rightarrow$ x (Don't Care)

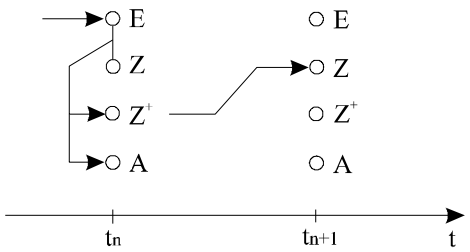
9.1.3 Struktur von Schaltwerken

Bestandteile eines Schaltwerkes:
- 1.Schaltnetz, das aus Eingangsvariablen und Momentanzustand den Folgezustand bestimmt. (Übergangs-Schaltnetz, Next State Decoder)
- 2.Schaltnetz, das die Ausgangsvariablen erzeugt (Ausgangs-Schaltnetz, Output Decoder)
- Speicher für Momentanzustand

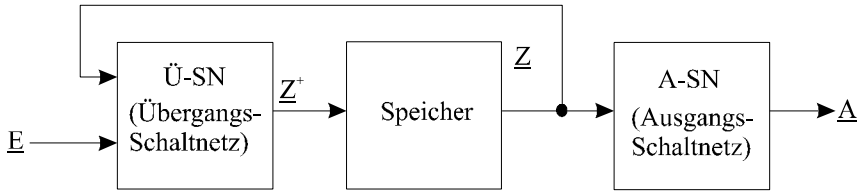
Mealy-Automat:



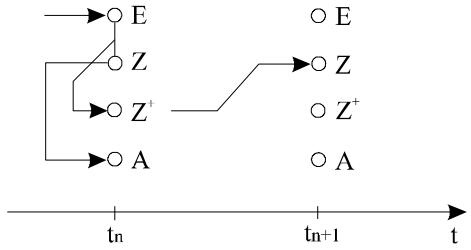
Zeitverhalten:



Moore-Automat:



Zeitverhalten:



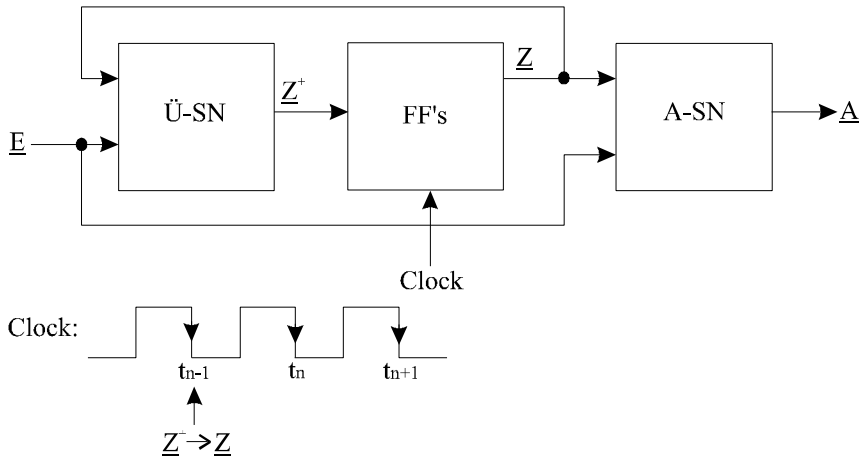
9.1.4 Asynchroner und synchroner Betrieb

asynchron: Änderung der Zustandsvariablen (und Ausgangsvariablen) asynchron (nicht gleichzeitig) bedingt durch Änderungen der Eingangsvariablen.

synchron: Änderung der Zustandsvariablen (und Ausgangsvariablen) synchron (gleichzeitig) in Abhängigkeit von den Eingangsvariablen. Auslösen der Änderung durch ein den Speicherelementen gemeinsames Taktsignal (Clock).

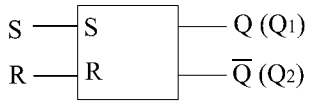
Speicherelement: Flipflop (2 stabile Zustände, 0 und 1)

Synchroner Mealy-Automat:



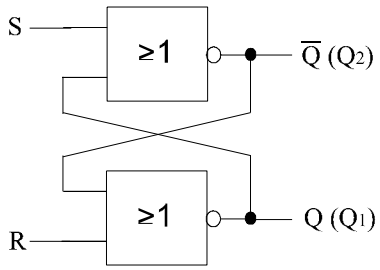
9.2 Flipflop's (FF's)

9.2.1 RS-Flipflop



S: Setz-Eingang (Set)
R: Rücksetz-Eingang (Reset)

Realisierung:



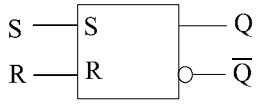
	R	S	Q_1	Q_1^+	Q_2^+	
Regulär $Q_2^+ = \overline{Q_1^+}$	0	0	0	0	1	Speichern
	0	0	1	1	0	$(Q^+ = Q)$
	0	1	0	1	0	Setzen
	0	1	1	1	0	$(Q^+ = 1)$
	1	0	0	0	1	Rücksetzen
	1	0	1	0	1	$(Q^+ = 0)$
	1	1	0	0	0	Irregulär
	1	1	1	0	0	$(Q_1^+ = Q_2^+)$

Regulärer Betrieb: $S \wedge R = 0$

Irregulärer Betrieb:

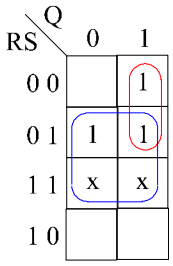
R	S	Q	Q^+
1	1	0	x
1	1	1	x

x = Don't Care
R = S = 1 wird ausgeschlossen !



	R	S	Q	Q^+
	0	0	0	0
	0	0	1	1
	0	1	0	1
	0	1	1	1
	1	0	0	0
	1	0	1	0
irreg.	1	1	0	x
	1	1	1	x

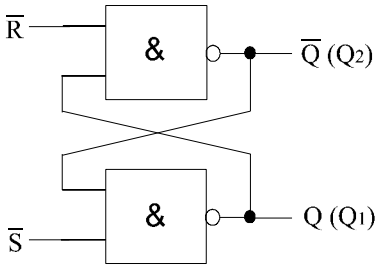
Q^+:



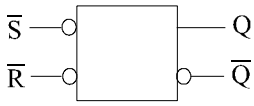
$$Q^+ = S \vee (Q \wedge \overline{R})$$

charakteristische Gleichung des FF

Alternative Realisierung:

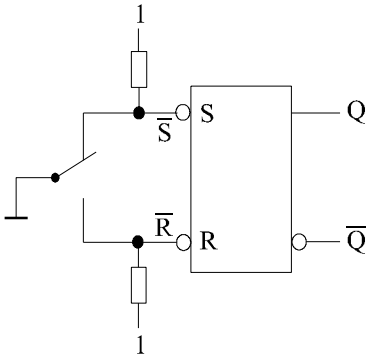


$\triangleq$



	$\overline{S}$	$\overline{R}$	Q Q_1	Q^+ Q_1^+	$\overline{Q^+}$ Q_2^+	
irreg. $(Q_1^+ = Q_2^+)$	0	0	0	1	1	
	0	0	1	1	1	
regulär $(Q_2^+ = \overline{Q_1^+})$	0	1	0	0	1	Löschen $(Q^+ = 0)$
	0	1	1	0	1	
	1	0	0	1	0	Setzen $(Q^+ = 1)$
	1	0	1	1	0	
	1	1	0	0	1	Speichern $(Q^+ = Q)$
	1	1	1	1	0	

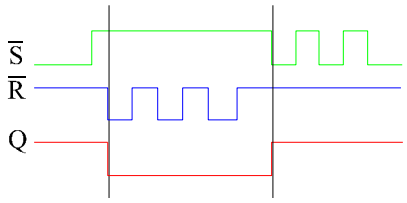
Anwendung: Entprellschaltung



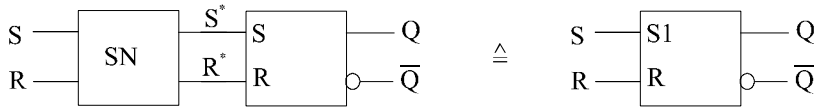
	$\overline{R}$	$\overline{S}$	Q^+
* →	0	0	x
schalten	0	1	0
Ruhe	1	0	1
Prellen	1	1	Q

* kann nicht eintreffen !

Timing:



Verhindern des irregulären Zustands mit Schaltnetz:



RS-FF mit Setzvorrang

R	S	R^*	S^*
0	0	0	0
0	1	0	1
1	0	1	0
1	1	0	1

$S^* = S$

$R^* = R \wedge \overline{S} = \overline{\overline{R} \vee S}$

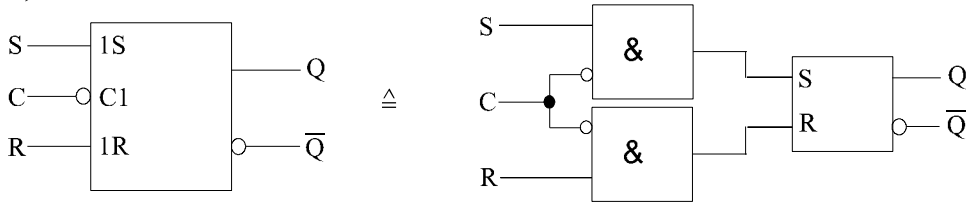
Setzvorrang !

Alternativ: Löschvorrang oder Signalerhalt

9.2.2 FF's mit Taktsteuerung

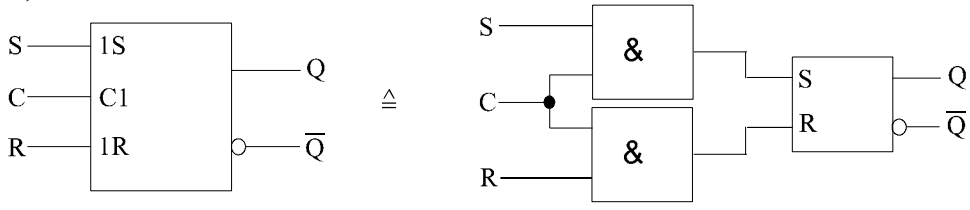
9.2.2.1 Taktzustandsgesteuerte Einspeicher-FF

a.)



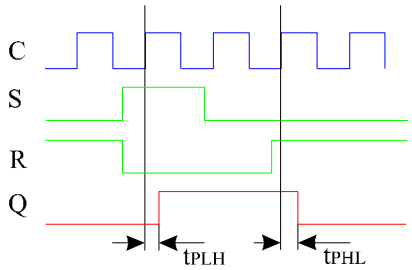
Passiver Taktzustand: C = 1 ($Q^+ = Q$)
Aktiver Taktzustand: C = 0
R, S : Vorbereitungseingänge
Nur regulärer Betrieb: $R \wedge S = 0$

b.)



Passiver Taktzustand: C = 0
Aktiver Taktzustand: C = 1

Timing:



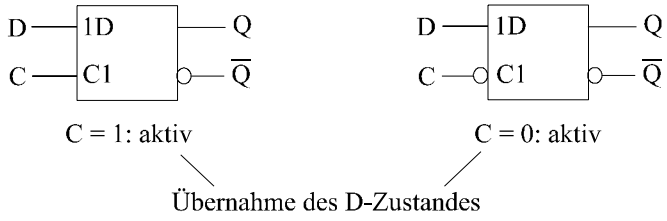
t_{PLH}: Verzögerungszeit $Q_L \rightarrow Q_H$ (Propergation Delay)
t_{PHL}: Verzögerungszeit $Q_H \rightarrow Q_L$

- Vorbereitungseingänge legen den Folgezustand fest
- Takt bestimmt den Zeitpunkt des Übergangs

↓

Signale an den Vorbereitungseingängen dürfen sich nur während des passiven Taktzustandes ändern !

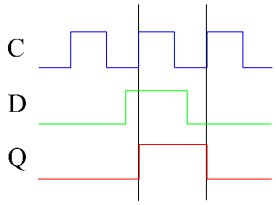
c.) D-FF



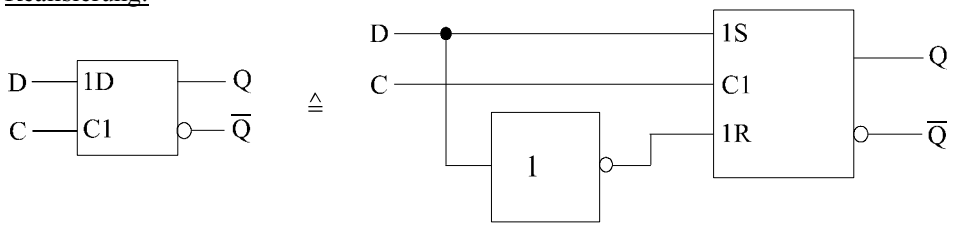
D	Q	Q^+
0	0	0
0	1	0
1	0	1
1	1	1

$\rightarrow$ $Q^+ = D$

C = 1 : aktiv:



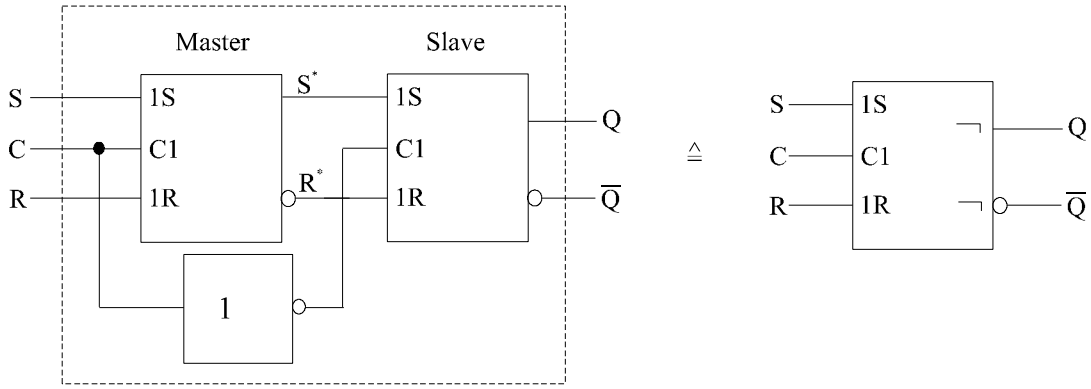
Realisierung:



$\Rightarrow$ $R \wedge S = 0!$ Stets regulärer Betrieb !

9.2.2.2 Taktzustandsgesteuerte Zweispeicher-FF

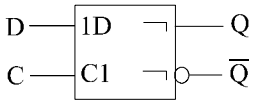
a.) Zweispeicher-RS-FF



Zwei Einspeicher-FF's
¬ retardierter Ausgang
C = 1: Set/Reset des Master-FF
C = 0: Set/Reset des Slave-FF
Änderung der Eingänge → verzögerte Änderung der Ausgänge (Retardierung)
Achtung ! Irregulärer Betrieb möglich (R = S = 1) !

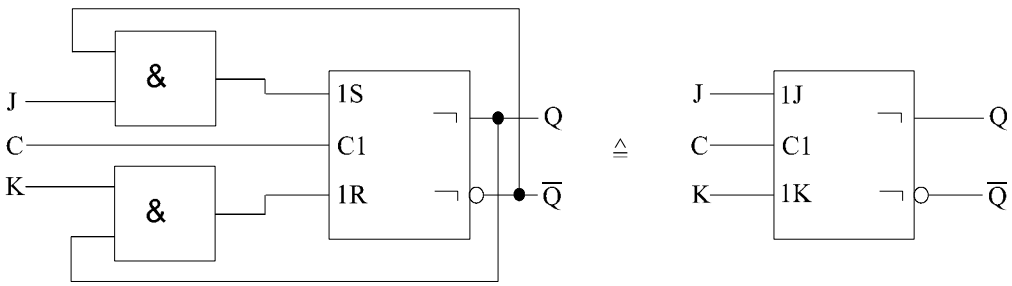
Andere Bezeichnung: Master-Slave-FF

b.) Zweispeicher-D-FF



Realisierung wie a.) !

c.) JK-FF



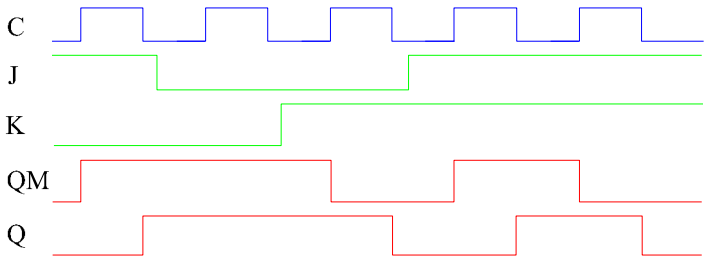
J	K	Q	R	S	Q^+	
0	0	0	0	0	0	Speichern
0	0	1	0	0	1	
0	1	0	0	0	0	
0	1	1	1	0	0	Reset
1	0	0	0	1	1	Set
1	0	1	0	0	1	Speichern
1	1	0	0	1	1	Set
1	1	1	1	0	0	Reset

→

J	K	Q^+	
0	0	Q	Speichern
0	1	0	Reset
1	0	1	Set
1	1	$\overline{Q}$	Wechseln

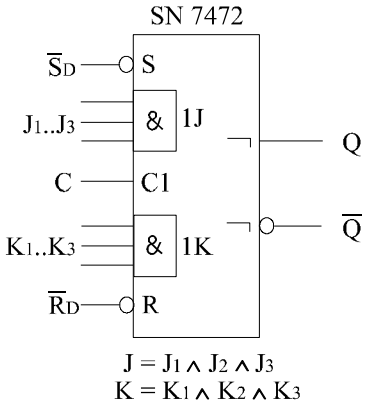
Kein irregulärer Zustand !

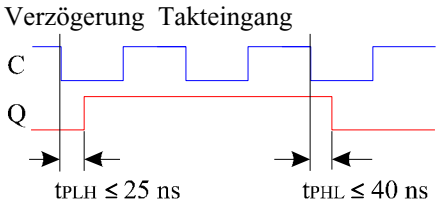
Timing-Diagramm:



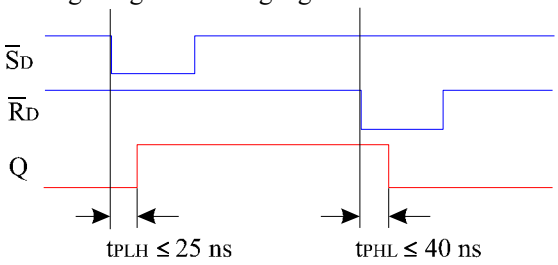
Anfangszustand: QM = Q = 0

Beispiel: SN 7472





Verzögerung Direkteingänge



Mehrfachvorbereitungseingänge (3x J, 3x K)

$\overline{S_D}$: Set Direct
$\overline{R_D}$: Re set Direct
} unabhängig von J, K, C !

9.2.2.3 Taktflankengesteuerte FF's

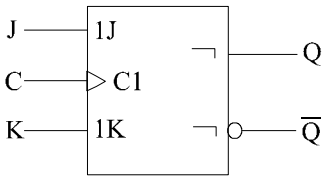
Wechsel 0/1 : positive Flanke ↑

1/0 : negative Flanke ↓

Aktive Flanke: ↑
Passive Flanke: ↓

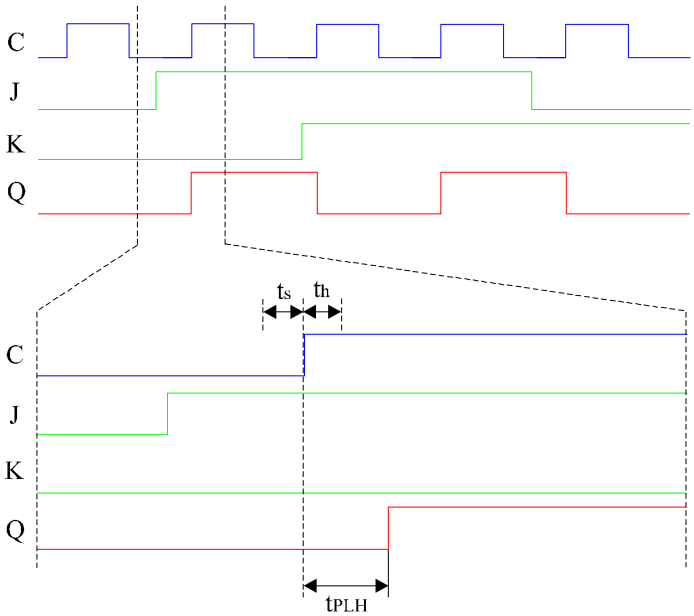
Aktive Flanke: ↓
Passive Flanke: ↑

a.) JK-FF

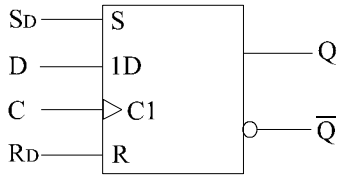


J	K	Q^+
0	0	Q
0	1	0
1	0	1
1	1	$\overline{Q}$

Impuls-Diagramm:

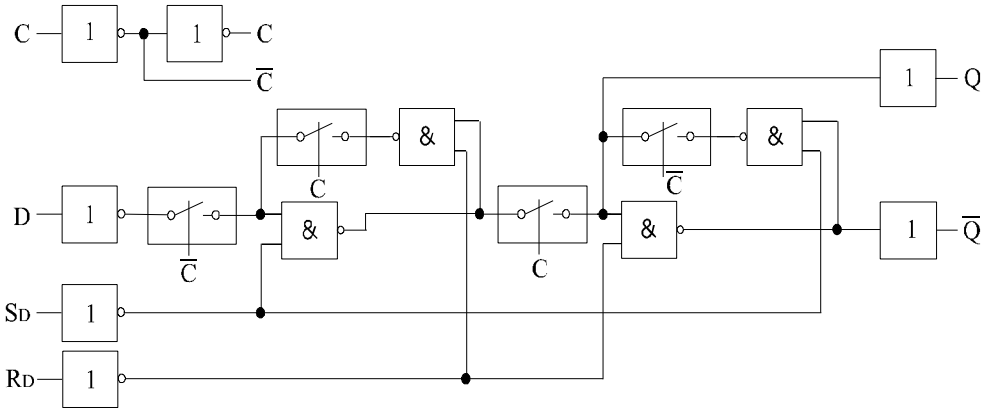


b.) D-FF

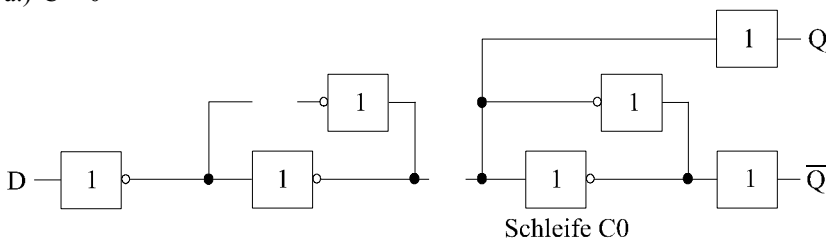


D	S_D	R_D	Q^+
0	0	0	0
1	0	0	1
x	1	0	1
x	0	1	0

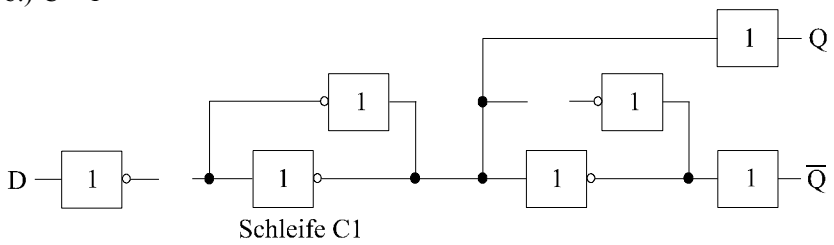
Schaltung:



a.) C = 0



b.) C = 1



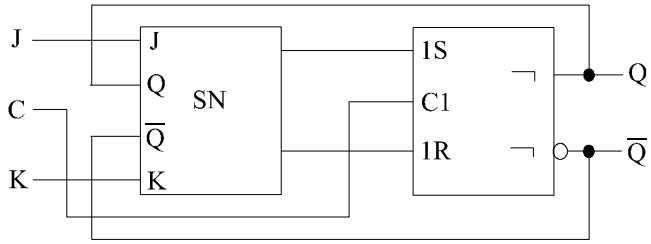
	RS-FF	JK-FF	D-FF
Zustands-folgetabelle (Kurzform)	S R | Q^+ 0 0 | Q 0 1 | 0 1 0 | 1 1 1 | x	J K | Q 0 0 | Q 0 1 | 0 1 0 | 1 1 1 | $\overline{Q}$	D | Q^+ 0 | 0 1 | 1
Charakter-istische Gleichung	$Q^+ = S \vee (\overline{R} \wedge Q)$ $S \wedge R = 0$	$Q^+ = (J \wedge \overline{Q}) \vee (\overline{K} \wedge Q)$	$Q^+ = D$
Synthese-tabelle (Kurzform)	Q Q^+ | S R 0 0 | 0 x 0 1 | 1 0 1 0 | 0 1 1 1 | x 0	Q Q^+ | J K 0 0 | 0 x 0 1 | 1 x 1 0 | x 1 1 1 | x 0	Q Q^+ | D 0 0 | 0 0 1 | 1 1 0 | 0 1 1 | 1

Übergangsverhalten bistabiler Flipflops

9.2.3 Konvertierung von FF's

Problem: Realisierung von FF-Typen mit anderen FF-Typen

Beispiel: Realisierung eines JK-FF mit RS-FF



J	K	Q	Q^+	R	S
0	0	0	0	x	0
0	0	1	1	0	x
0	1	0	0	x	0
0	1	1	0	1	0
1	0	0	1	0	1
1	0	1	1	0	x
1	1	0	1	0	1
1	1	1	0	1	0

Keine irregulären Zustände !

R:

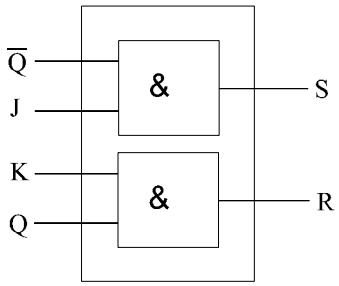
JK \ Q	0	1
0 0	x	
0 1	x	1
1 1		1
1 0		

$R = Q \wedge K$

S:

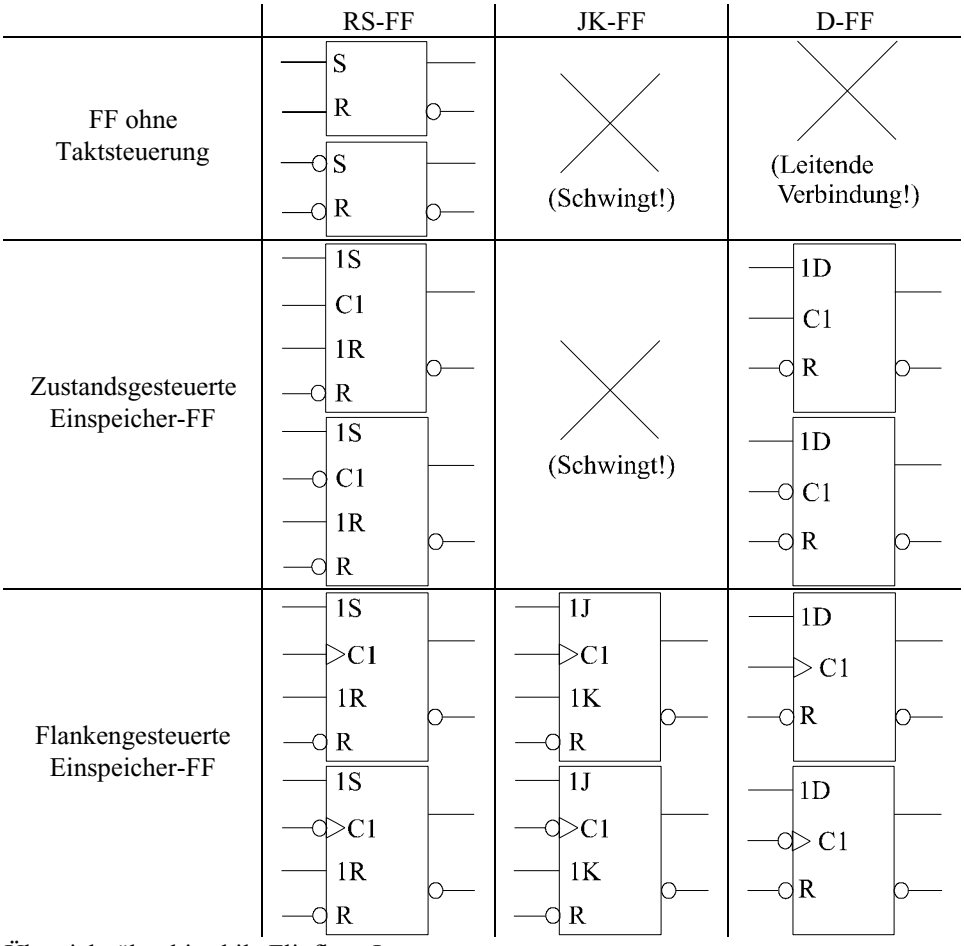
JK \ Q	0	1
0 0		x
0 1		
1 1	1	
1 0	1	x

$S = \overline{Q} \wedge J$

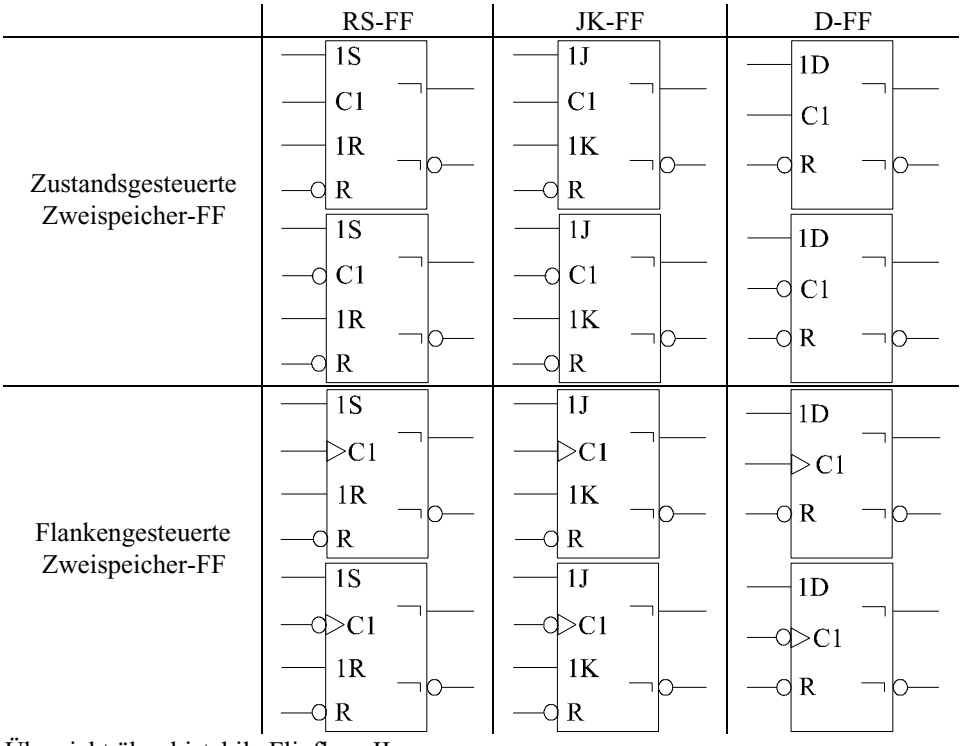


Schaltnetz

9.2.4 Überblick



Übersicht über bistabile Flipflops I



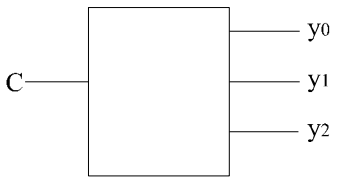
Übersicht über bistabile Flipflops II

9.3 Zähler

Zählertypen: synchron - asynchron
Einrichtungs- , Zweirichtungszähler (vorwärts / rückwärts)
programmierbare Zähler (Laden des Zählerwertes)

9.3.1 Synchrone Zähler

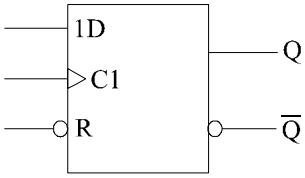
Beispiel: mod-5-Vorwärtszähler (0, 1, 2, 3, 4, 0, 1, 2, ...) mit D-FF



Dez. Zählerstand	y_2	y_1	y_0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0

5 Zustände ! $\rightarrow$ 3 FF's

$y_0 = Q_0$
$y_1 = Q_1$
$y_2 = Q_2$



Dez. Zust.	Q_2	Q_1	Q_0	$D_2 = Q_2^+$	$D_1 = Q_1^+$	$D_0 = Q_0^+$
0	0	0	0	0	0	1
1	0	0	1	0	1	0
2	0	1	0	0	1	1
3	0	1	1	1	0	0
4	1	0	0	0	0	0
5	1	0	1	x 0	x 1	x 0
6	1	1	0	x 0	x 1	x 0
7	1	1	1	x 1	x 0	x 0

D_0:

Q_2Q_1 \ Q_0	0	1
0 0	1	
0 1	1	
1 1	x	x
1 0		x

$D_0 = \overline{Q}_0 \wedge \overline{Q}_2$

D_1:

Q_2Q_1 \ Q_0	0	1
0 0		1
0 1	1	
1 1	x	x
1 0		x

$D_1 = (\overline{Q}_0 \wedge Q_1) \vee (Q_0 \wedge \overline{Q}_1)$

D_2:

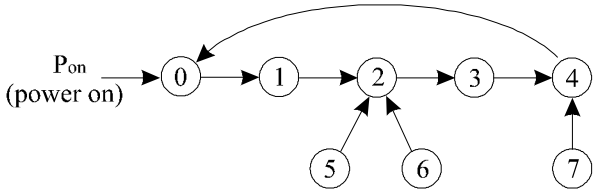
Q_2Q_1 \ Q_0	0	1
0 0		
0 1		1
1 1	x	x
1 0		x

$D_2 = Q_0 \wedge Q_1$

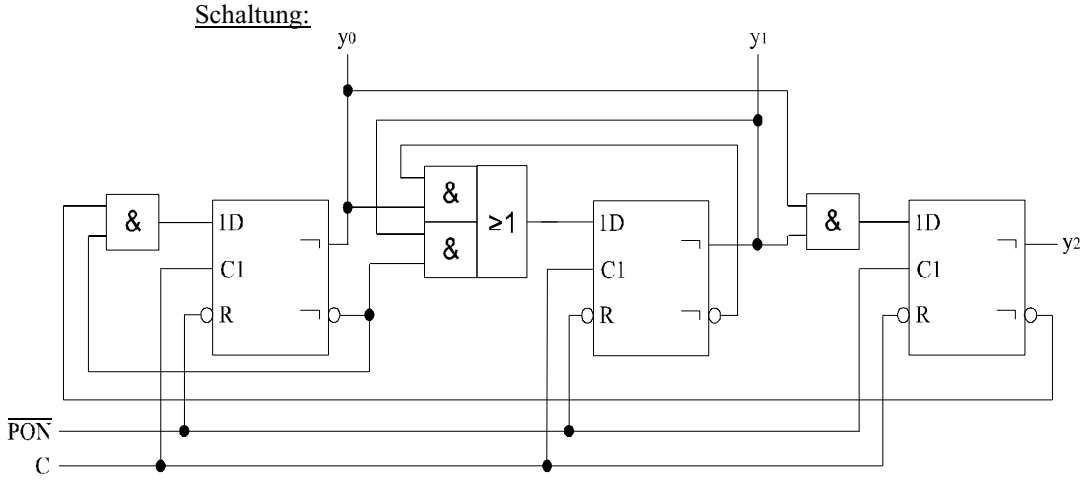
→ Zähler fängt sich selbst nach Störungen !

Satz: Pseudozustände treten bei Normalfunktion nicht auf. Störungen können jedoch Pseudozustände bewirken. Der Zähler muß selbständig Pseudozustände verlassen und in den normalen Zyklus zurückfinden. Oft wird ein bestimmter Folgezustand für Pseudozustände gefordert (z.B. 0 .... 0).

Zustandsfolgediagramm:

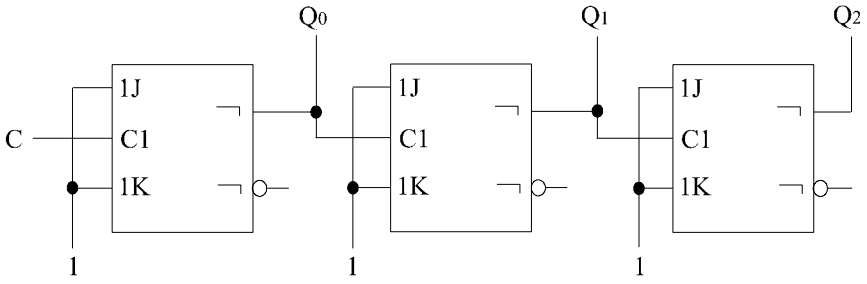


Schaltung:



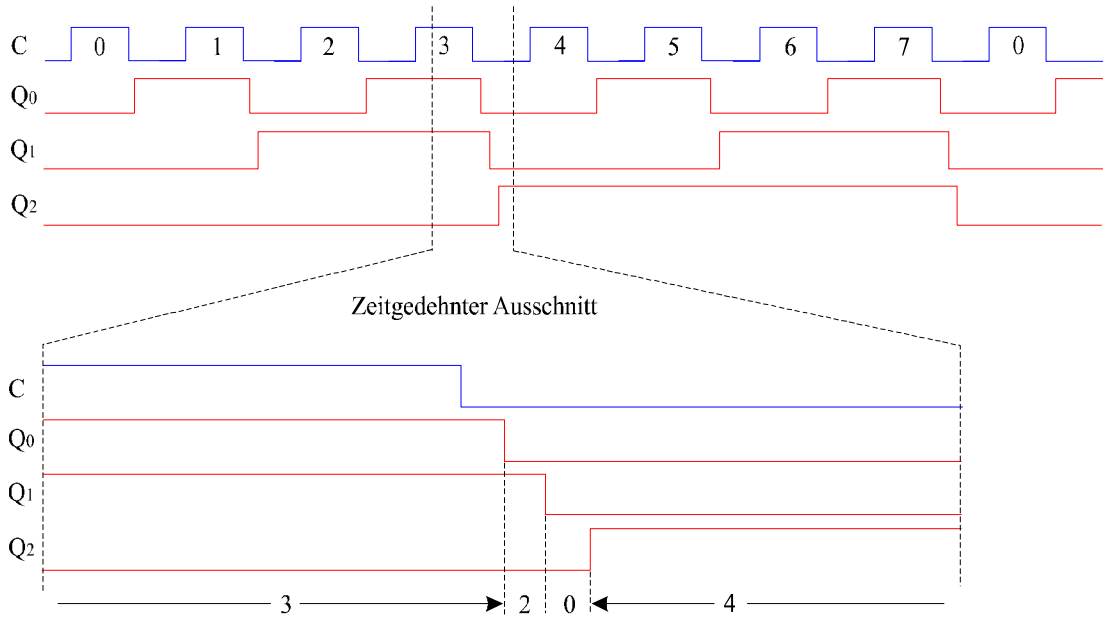
9.3.2 Asynchrone Zähler

Beispiel: mod-8-Vorwärtszähler mit JK-FF's



JK = 1 → $Q^+ = \overline{Q}$ (Wechseln) → Halbieren der Taktfrequenz

Problem: Zählen von 3 auf 4 → vorübergehend falsche Zustände 0, 2.



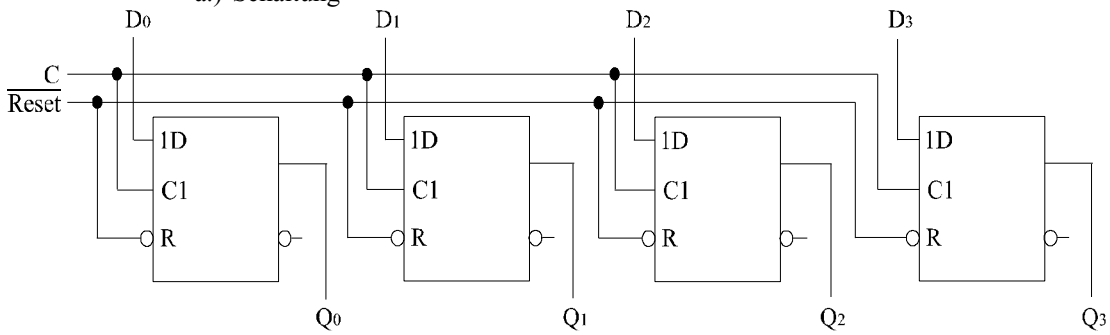
9.4 Register

9.4.1 Auffangregister (Latch)

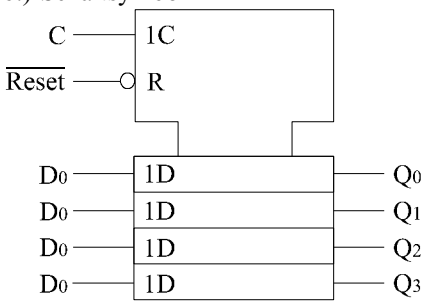
Aufgabe: Speichern von n Bit

Beispiel: 4-Bit-Latch

a.) Schaltung



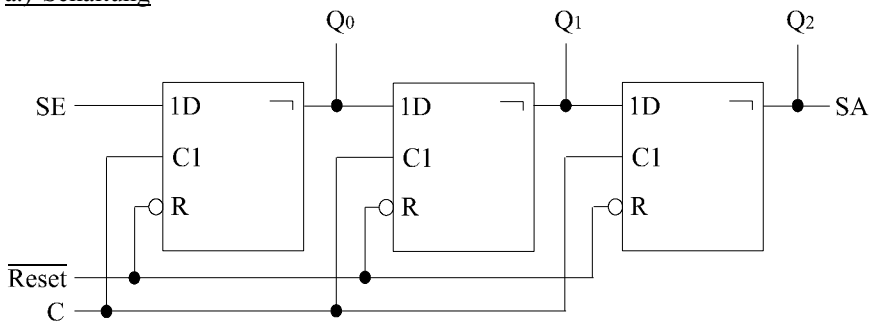
b.) Schaltsymbol



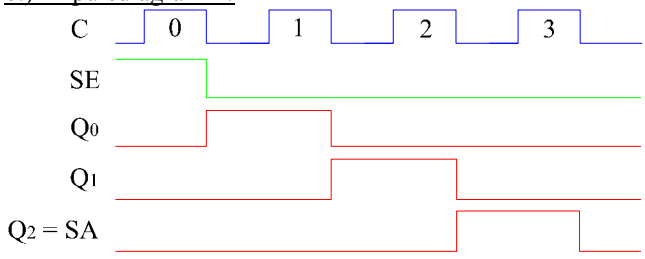
9.4.2 Schieberegister

Beispiel: 3 Bit Schieberegister

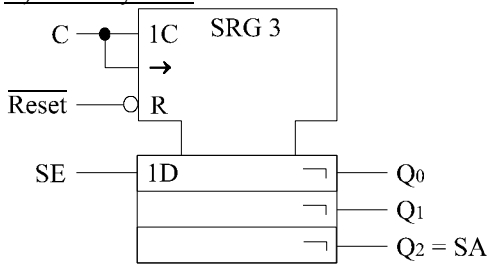
a.) Schaltung



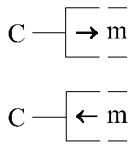
b.) Impulsdiagramm:



c.) Schaltsymbol:

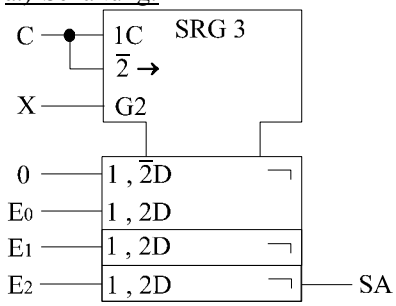


d.) Schiebeeingang:

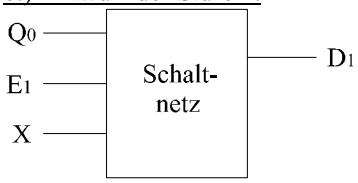


Beispiel: Parallel / Seriell-Umsetzer für 3 Bit mit D-FF

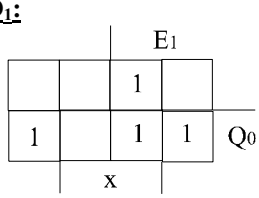
a.) Schaltung:



b.) Entwurf der Stufe 1:



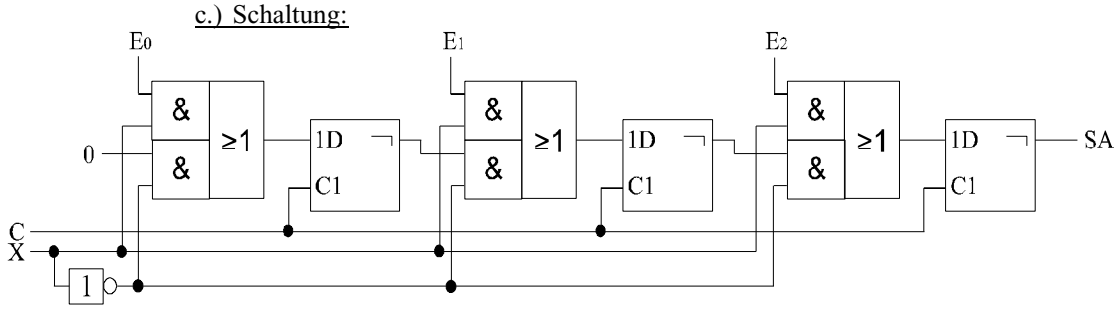
D1:



x	Q_0	E_1	D_1
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

$$D_1 = \left(X \wedge E_1\right) \vee \left(\overline{X} \wedge Q_0\right)$$

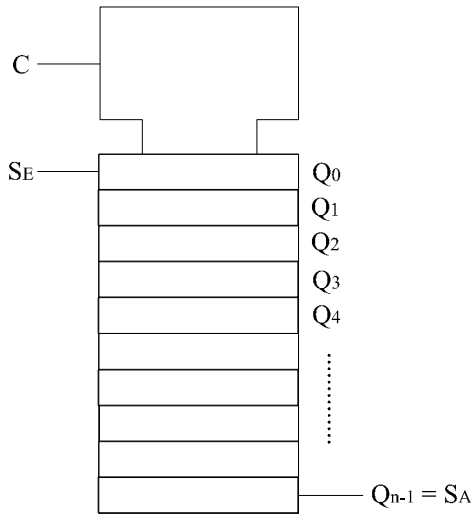
c.) Schaltung:



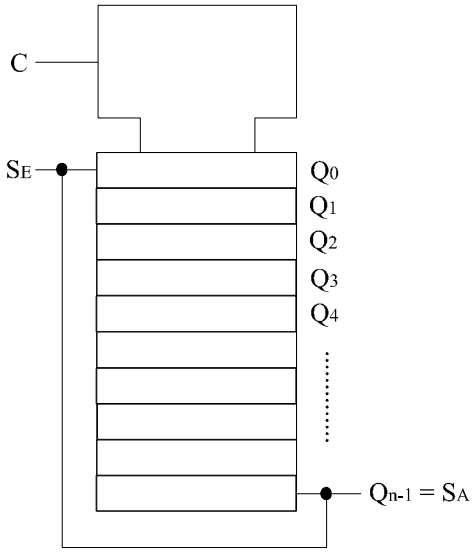
X = 0 : Dateneingänge E_0, E_1, E_2 gesperrt. Weitergabe der FF-Zustände.
X = 1 : Dateneingänge E_0, E_1, E_2 freigegeben.

9.4.3 Schieberegister mit Rückführung

Schieberegister mit seriellem Eingang / Ausgang



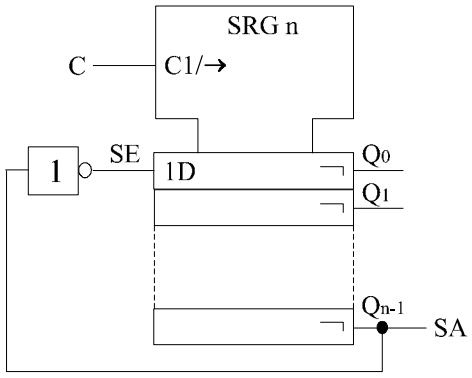
1.) Umlaufregister:



Ausgangszustand nach n Schritten !
Ausgangszustand ≠ 0 ... 0, 1 ... 1

2.) Invertierte Rückführung:

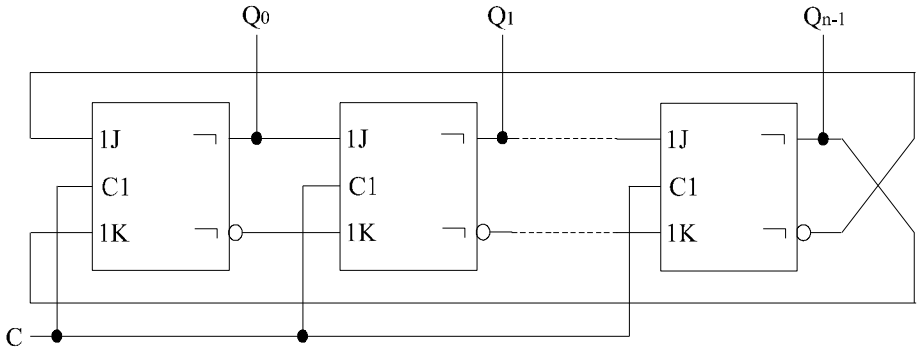
a.) Johnson-Zähler



SE = serieller Eingang

SA = serieller Ausgang

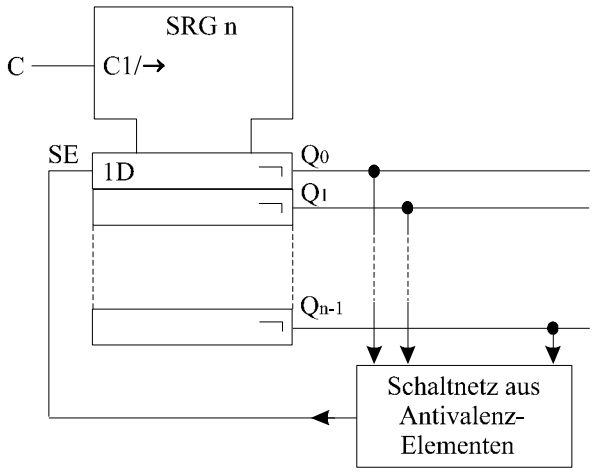
b.) Switched-Tail-Ring-Counter



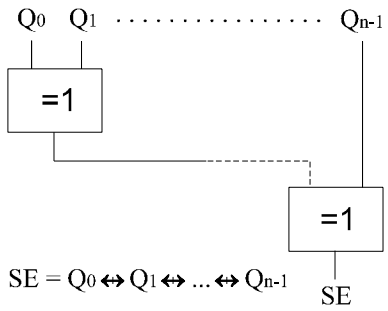
nach n Schritten: Invertierter Ausgangszustand
nach 2n Schritten: Ausgangszustand

3.) Rückführung mit EXOR-Schaltnetz:

a.) Rückgekoppeltes Schieberegister



b.) Schaltnetz aus Antivalenz-Elementen



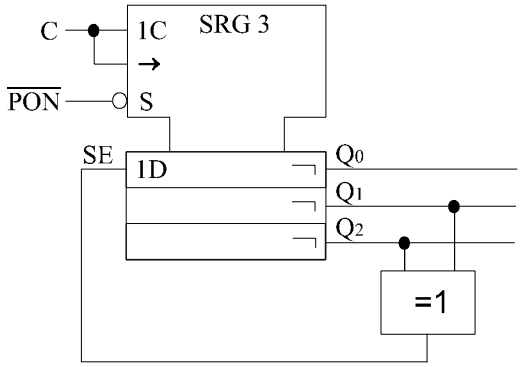
$$SE = \left(\sum_i Q_i \right)_{mod\,2}$$

p_{ON} : Setzen des Anfangszustandes

Pseudo-Zufallsfolge: Zyklische Wiederholung einer ‚Zufallsfolge“
(Hier: Zufallsfolge aus 7 Zuständen)

Satz: Mit einem n-Bit-Schieberegister läßt sich eine Pseudozufallsfolge mit der Periodenlänge 2^n-1 erzeugen (Zustand 0 ... 0 entfällt).

a.) Schaltung:



b.) Zustandsfolge:

	Q_0	Q_1	Q_2
0	1	1	1
1	0	1	1
2	0	0	1
3	1	0	0
4	0	1	0
5	1	0	1
6	1	1	0
0	1	1	1
:		:	

c.) Tabelle:

n	Ausg.-Indizes
3	1 2
4	2 3
5	2 4
6	4 5
7	3 6
8	2 4 6 7
9	4 8
10	6 9
11	8 10
12	5 7 10 11

ANHANG A

3 Bit Schieberegister 92
4 Bit-Volladdierer 69
4-Bit-Latch 91

A
Abhängigkeitsnotation 56
Addition 6
alte Symbole 29
analog 1
analoge Signale 1
Antivalenz 28
Äquivalenz 28
ASCII 18
asynchron 78
Aufzugsteuerung 32

B
Basisverknüpfungen 29
Bewertbarer Code 10
binär 1
Binärcode 10
Block Check Charakter 21
Byte 2

C
Code (DIN 44 300) 10
Codierung mehrerer Ziffern 10
Complementary MOS 46
CRC (Cyclic Redundancy Check) 21

D
Demultiplexer 67
depletion load 46
D-FF 82
digital 1
digitale Messung 1
Disjunktion 28
Disjunktive Normalform 33
Distanz D 10
Division 7
Don't care-Zustand 39
Dualcode → Gray-Code 13

E
EGB 53
Einser-Komplement 8
Electrostatic Sensitive Device 53
Elektrostatisch gefährdetes Bauelement 53
Entprellschaltung 80
Entwurf eines Demultiplexers 67
Entwurf eines Multiplexers 66
EPROM 72
ESD 53
Exklusiv-NOR 28
Exklusiv-ODER 28
EXOR 28

F
Fehlererkennung 19
Fehlerkorrektur 19

G
Gatterlaufzeit 44
Gegentaktausgang 48
Gewicht G 10

H
Halbaddierer 68
Hamming-Code 22

I
Implikation 29
Inverter 35
invertierte DMF 40
invertierte DNF 40

J
JK-FF 83
Johnson-Zähler 94

K
Karnaugh-Veith-Diagramm 38
Kollektor-Widerstand 47
Konjunktion 28
konjunktive Normalform 34
Kontroll-Bits 21
Konvertierung von FF's 87
KV-Diagramm 38

L
Logikpegel 35
Logische Zustände 35

M
m aus n 31
Majorität 31
Maximal-Distanz D_{max} 11
Mealy-Automat 77
Minimal-Distanz D_{min} 11
mod-5-Vorwärtszähler 89
mod-8-Vorwärtszähler 90
Moore-Automat 78
Multiplexer 65
Multiplikation 6

N
Nachbildung eines ROM 72
Nachkommastellen 4
NAND 28
Noise Margins 52
NOR 28
Nummernüberblick Digital-ICs 49

O
ODER 28
Open Collector 47
Open Drain 47

P
Parallel / Seriel-Umsetzer 92
Parallel-Erweiterung 63
Paritätsbit 20
Parity even 20
Parity odd 20
Pegelsymbol → Logiksymbol 37
Performance Comparison 50
PROM 72
Pseudozustände 90

R
Realisierung eines JK-FF mit RS-FF 87
Realisierung ohne Minimierung 69
Redundanz 11
ROM 72

S
Schaltelemente mit einem Eingang 26
Schaltelemente mit zwei Eingängen 27
Schaltfunktion 24
Schaltnetz 74
Schaltsymbole (DIN 40900, Teil 12) 29
Schaltsymbole für die Ausgangsstufen 48
Schaltsymbole für Wired-Verknüpfungen 48
Schaltvariable 24
Schaltwerk 74
Schiebeeingang (Symbol) 92
Schieberegister mit seriellem Ein- / Ausgang 93
Schmitt-Trigger 51
Schottky-TTL 44
Schutzmaßnahmen 53
Schwellwert 30
Serien-Erweiterung 62
Stellenschreibweise 3
Stetiger Code 11
Subtraktion 6
Switched-Tail-Ring-Counter 94
synchron 78
Synchroner Mealy-Automat 78

T
Tetrade 2
Tristate-Ausgang 49

U
Übersicht über bistabile Flipflops I 88
Übersicht über bistabile Flipflops II 88
Umformung von Schaltsymbolen 37
Umlaufregister 94
UND 28
Ungerade 31

V
Vereinfachung (Minimieren) 38
Verhindern des irregulären Zustands 81
Volladdierer 68
Vollständiger Code 11
Vorzeichen 7

W
Wichtige Potenzen 4
Wired-Technik 47
Worst-Case Values 52

Z
Zahlensysteme 3
Zählertypen 89
Zustandsdiagramm 75
Zustandsfolgetabelle 76
Zweispeicher-D-FF 83
Zweispeicher-RS-FF 83