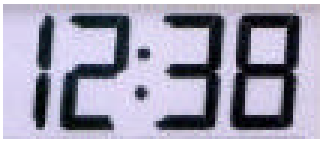
Digitale Darstellung von Größen

Eine Meßgröße ist digital, wenn sie in ihrem Wertebereich nur eine endliche Anzahl von Werten annehmen kann, also "abzählbar" ist.
Digital kommt dabei von digitus (lat.: der Finger). Eine Zahl kann z.B. durch eine Anzahl von Fingern dargestellt werden.
Ein einfacher Digitalrechner ist der Rechenrahmen, bei dem Zahlen durch eine entsprechende Anzahl von Kugeln dargestellt werden. Der Genauigkeit ist dabei im Prinzip keine Grenze gesetzt, denn die Anzahl der Kugeln kann beliebig erhöht werden.

Bei elektronischen Digitalrechnern verwendet man statt der Kugeln elektrische Impulse. Jede Zahl kann durch eine Anzahl von Impulsen dargestellt werden.

Da digitale Größen durch abzählbar viele Elemente dargestellt werden können, verwendet man die Veranschaulichung durch Zahlen, die aus Ziffern zusammengesetzt sind.



(Hier: 12 Uhr und 38 Minuten)

Verwendet man zur Darstellung nur binäre Elemente (bis lat.: zweimal) so spricht man von binärer Digitaltechnik.

Da zur Zeit im technischen Bereich nur die binäre Darstellung verwendet wird, kann hier der Zusatz binär entfallen und man spricht nur von Digitaltechnik.

Die in der Digitaltechnik üblichen beiden binären Zustände werden auch digitale Zustände genannt. Beispiele:

Erster binärer Zustand	Zweiter binärer Zustand
Schalter offen	Schalter geschlossen
Diode leitend	Diode gesperrt
Spannung hoch	Spannung niedrig
Werkstoff magnetisiert	Werkstoff nicht magnetisiert
Lampe leuchtet	Lampe leuchtet nicht
Stromstärke hoch	Stromstärke niedrig
Sensor aktiviert	Sensor nicht aktiviert

Wir arbeiten mit elektronischen Bauteilen, bei denen Spannungszustände als binäre Zustände verwendet werden. Bei den von uns verwendeten TTL-Bausteinen (Transistor-Transistor-Logik) der 74xxx-Familie haben wir die beiden Zustände

0	(low)	entspricht	0V (Masse)
1	(high)	entspricht	+5V

Dabei gelten die folgenden Toleranzen:

2,5V –5,5 V high

low	:	0V bis	0,8V
high	:	2,5V bis	5,5V

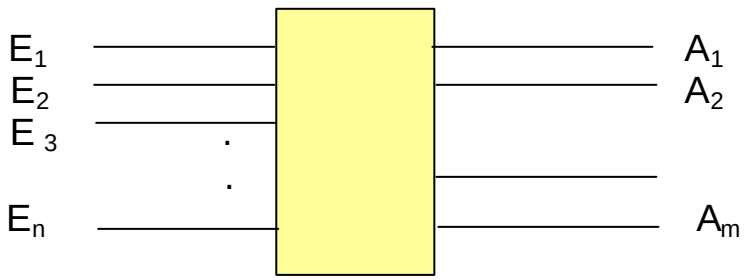
Der Zusammenhang mit der zweiwertigen Logik wird dabei durch folgende Festlegung hergestellt:

0	bzw. low	entspricht dem Wahrheitswert	false
1	bzw. high	entspricht dem Wahrheitswert	true

Man nennt diese Festlegung „positive Logik“

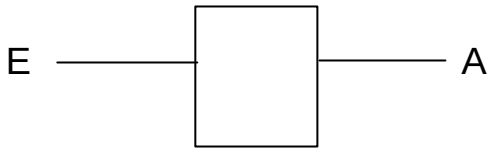
Digitale Bausteine

Unter einem digitalen Baustein versteht man eine elektronische Schaltung mit n Eingängen und m Ausgängen, in der die unterschiedlichen Belegungen der Eingangsleitungen so verarbeitet werden, dass jede Ausgangsleitung eine eindeutig definierten Ausgangswert besitzt.



Wir untersuchen vorerst nur Schaltungen mit einer Ausgangsleitung:

Die einfachste Digitalschaltung besitzt einen Eingang und einen Ausgang



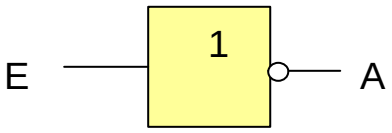
Für die Funktion sind dann folgende vier Möglichkeiten denkbar:

	NULL	IDENTISCH	INVERTER	EINS
E	A	A	A	A
0	0	0	1	1
1	0	1	0	1

Der NULL-Baustein ist überflüssig, denn er gibt unabhängig von der Belegung des Eingangs immer eine 0 aus. Diese Null kann ich auch direkt am Minuspol der Spannungsquelle abgreifen. Genauso verhält es sich mit dem EINS-Baustein der immer eine 1 ausgibt, die ich wieder am Pluspol der Spannungsquelle abgreifen kann. Der IDENTISCH-Baustein ist ebenfalls überflüssig , denn er bewirkt keine Änderung der Eingangsbelegung.

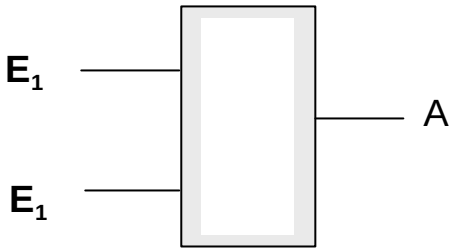
Als einzig sinnvoller Baustein bleibt also der Inverter übrig, der die Belegung des Eingangs gerade invertiert.

NOT



E	A
0	1
1	0

Digitalschaltungen mit 2 Eingängen und einem Ausgang:



Es gibt genau 16 Bausteine dieses Typs:

		CONST 0	AND	Inhibition E_2	Identität E_1	Inhibition E_1	Identität E_2	XOR	OR	NOR	Äquivalenz	Negation E_2	Implikation E_2	Negation E_1	Implikation E_1	NAND	CONST 1
E_1	E_2	A_0	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	A_{11}	A_{12}	A_{13}	A_{14}	A_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Inhibition : Verbot, Hemmung

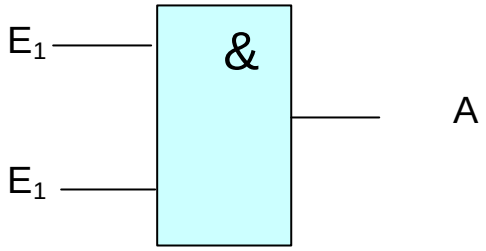
Merke: Jeder Baustein dieser Tabelle kann ausschließlich mit NAND-Bausteinen aufgebaut werden. Dieser Bausteine bildet eine Basis für alle anderen Bausteine.

Aufgabe 1 Baue einen NOT (AND, OR, NOR) nur mit NAND-Bausteinen auf.

Von technischer Bedeutung sind nur die Bausteine NOT, AND, NAND, OR, NOR und XOR

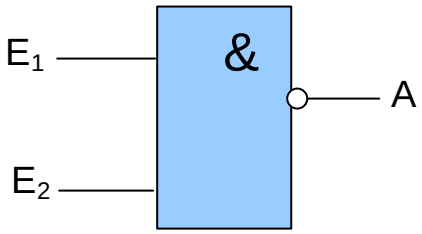
Die Bausteine mit ihren DIN-Schaltzeichen:

AND



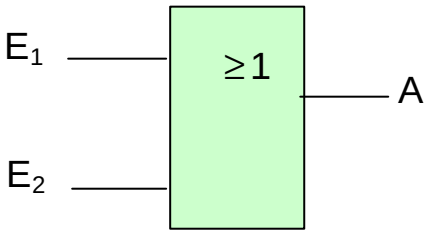
E_1	E_2	$A = E_1 \wedge E_2$
0	0	0
0	1	0
1	0	0
1	1	1

NAND



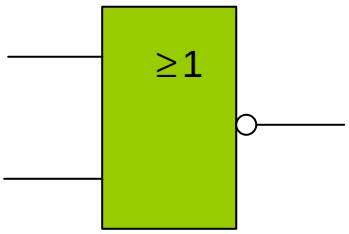
E_1	E_2	$A = \overline{E_1 \wedge E_2}$
0	0	1
0	1	1
1	0	1
1	1	0

OR



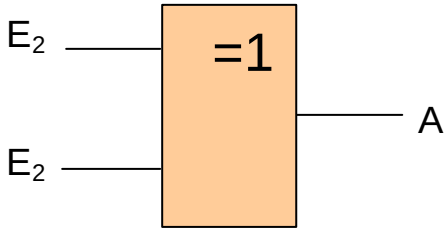
E_1	E_2	$A = E_1 \vee E_2$
0	0	0
0	1	1
1	0	1
1	1	1

NOR



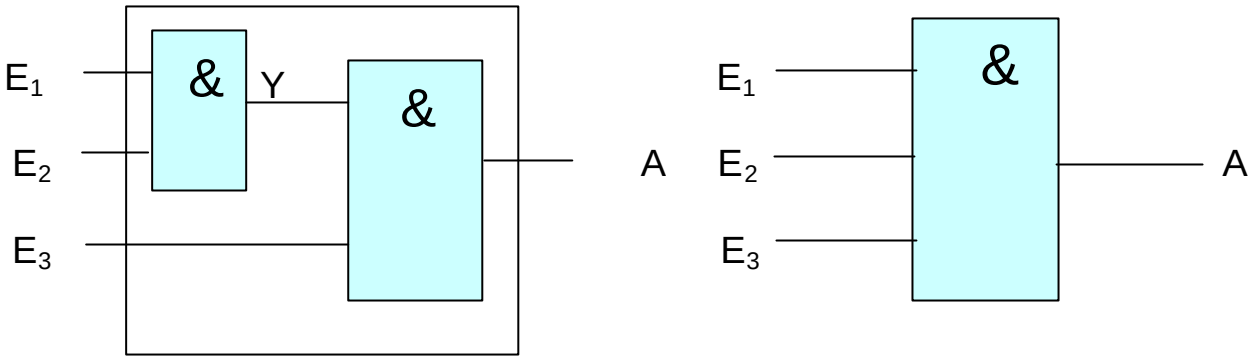
E_1	E_2	$A = \overline{E_1 \vee E_2}$
0	0	1
0	1	0
1	0	0
1	1	0

XOR



E_1	E_2	$A = E_1 \dot{\vee} E_2$
0	0	0
0	1	1
1	0	1
1	1	0

Bausteine mit 3 und mehr Eingängen lassen sich aus Bausteinen mit zwei Eingängen erstellen:



Wertetabelle:

E_1	E_2	E_3	Y	A
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Merke : Ein AND-Baustein mit n Eingängen hat genau dann den Ausgangswert A=1, wenn alle n Eingänge mit 1 belegt sind.

Aufgabe 2 Formuliere eine entsprechende Aussage auch für OR-Bausteine mit n Eingängen:

Aufgabe 3: Zeige, dass für die Verknüpfungen $\wedge$ und $\vee$ jeweils das Assoziativgesetz gilt, indem du zeigst, dass beide Verknüpfungstabellen zu äquivalenten Ergebnissen führen:

a) $(E_1 \wedge E_2) \wedge E_3 \quad = \quad E_1 \wedge (E_2 \wedge E_3)$

E_1	E_1	E_1	$E_1 \wedge E_2$	$(E_1 \wedge E_2) \wedge E_3$	$E_2 \wedge E_3$	$E_1 \wedge (E_2 \wedge E_3)$
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

Zeichne zu beiden Termen die entsprechenden Schaltungen

b) $(E_1 \vee E_2) \vee E_3 \quad = \quad E_1 \vee (E_2 \vee E_3)$

E_1	E_2	E_3	$E_1 \vee E_2$	$(E_1 \vee E_2) \vee E_3$	$E_2 \vee E_3$	$E_1 \vee (E_2 \vee E_3)$
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

c) Zeichne zu beiden Termen die entsprechenden Schaltungen:

Aufgabe 4: Zeige auf die gleiche Weise die Gültigkeit der beiden Distributiv-gesetze:

a) $(E_1 \vee E_2) \wedge E_3 \quad = \quad (E_1 \wedge E_3) \vee (E_2 \wedge E_3)$

E_1	E_2	E_3	$E_1 \vee E_2$	$(E_1 \vee E_2) \wedge E_3$	$E_1 \wedge E_3$	$E_2 \wedge E_3$	$(E_1 \wedge E_3) \vee (E_2 \wedge E_3)$
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

Zeichne für jeden Term die entsprechende Schaltung:

b) $(E_1 \vee E_2) \wedge E_3 \quad = \quad (E_1 \wedge E_3) \vee (E_2 \wedge E_3)$

E_1	E_2	E_3	$E_1 \vee E_2$	$(E_1 \vee E_2) \wedge E_3$	$E_1 \wedge E_3$	$E_2 \wedge E_3$	$(E_1 \wedge E_3) \vee (E_2 \wedge E_3)$
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

Zeichne für jeden Term die entsprechende Schaltung:

Aufgabe 5)

Zeige, dass die beiden Schaltfunktionen Z_1 und Z_2 äquivalent sind und zeichne die entsprechenden Schaltungen

$$Z_1 = (\overline{A} \wedge B \wedge C) \vee (\overline{A} \wedge B \wedge \overline{C}) \quad \text{und} \quad Z_2 = \overline{A} \wedge B$$

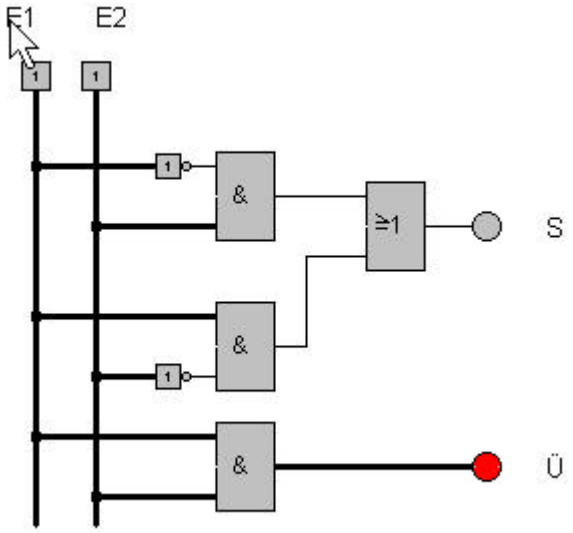
Rechenschaltungen

Unser Ziel: Wir wollen eine Schaltung entwickeln mit der man die Summe von zwei (und mehr) Dualzahlen berechnen kann.
Beispiel:

	1	0	1	1
+	1	0	0	1
1	0	1	1	
1	0	1	0	0

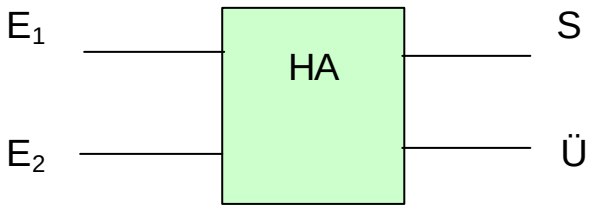
Beim ersten Schritt muss man zwei einstellige Dualzahlen addieren und den Übertrag ermitteln. Dabei können vier Fälle auftreten:
0+0=0 Übertrag =0 ; 0+1=1 Übertrag =0 ; 1+0=0 Übertrag =0 und
1+1=0 Übertrag =1

Der Halbaddierer ist eine Schaltung die genau dies leistet:



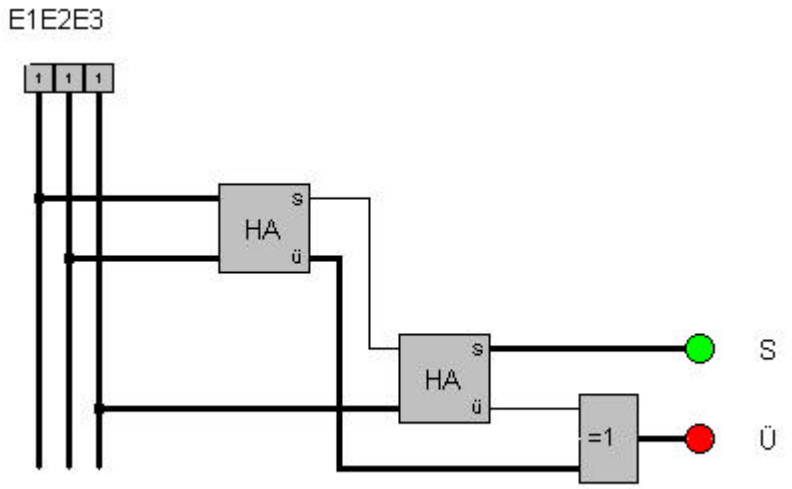
E_1	E_2	S	Ü
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Schaltsymbol:



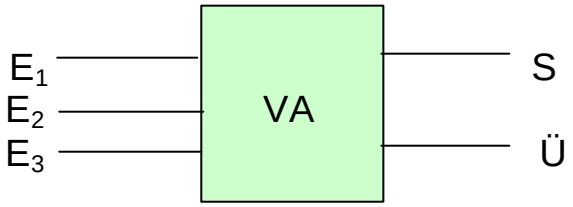
Bei den weiteren Schritten muss man dann jeweils drei einstellige Dualzahlen addieren und den Übertrag ermitteln. Hier können acht Fälle auftreten:
0+0+0=0 Übertrag =0; ..... 1+1+1=1 Übertrag =1

Der Volladdierer (aufgebaut aus zwei Halbaddierern):

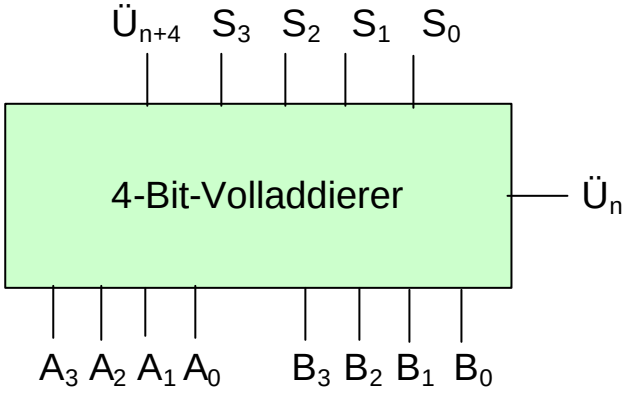


E_1	E_2	E_3	S	Ü
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Schlaltsymbol:

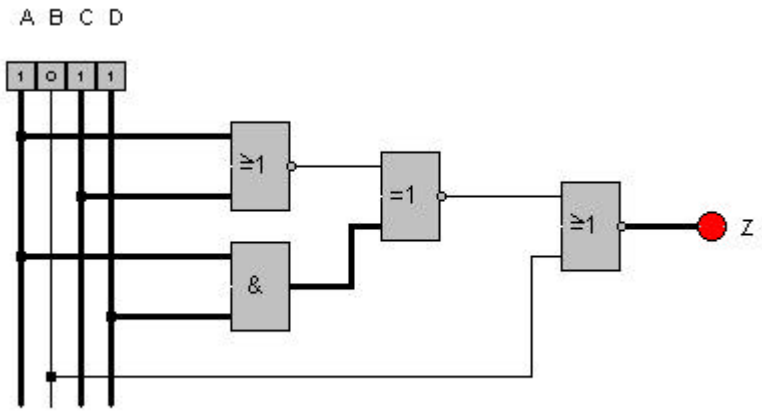


Aufgabe 6: Baue die Innenschaltung eines 4-Bit Volladdierers mit normalen Volladdierern auf:



Lösung:

Aufgabe 7: Stelle für folgende Schaltung die zugehörige Schaltfunktion auf und gib die Tabelle mit den zugehörigen Funktionswerten an:



Z=_____

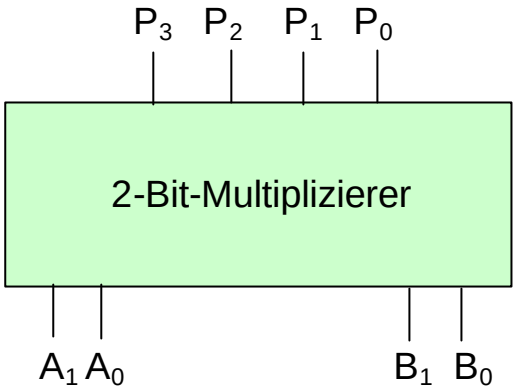
A	B	C	D				Z
0	0	0	0				
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	0				
0	1	1	0				
0	1	1	1				
1	0	0	0				
1	0	0	1				
1	0	1	0				
1	0	1	1				
1	1	0	0				
1	1	0	1				
1	1	1	0				
1	1	1	1				

Aufgabe 8: Zeichne zu folgender Schaltfunktion das zugehörige Schaltbild:

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

Zeige, dass Z unabhängig von der Belegung von A,B,C immer den Wert 0 annimmt.

Aufgabe 9 Entwickle eine 2-Bit Multiplizierschaltung



Beispiele: 10*11=0110 11*11=1001

A_1	A_0	B_1	B_0	P_3	P_2	P_1	P_0
0	0	0	0				
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	0				
0	1	1	0				
0	1	1	1				
1	0	0	0				
1	0	0	1				
1	0	1	0				
1	0	1	1	0	1	1	0
1	1	0	0				
1	1	0	1				
1	1	1	0				
1	1	1	1	1	0	0	1

Zur Herleitung der Innenschaltung eine Vorüberlegung:

```
 10 *11
  10    1. Summand
 _10_   2. Summand
 110    Summe und gleichzeitig Wert des Produkts
```

Zeichne die Innenschaltung des 2-Bit-Multiplizierers:

a) Entwickle eine Schaltung, die dir den 1. Summanden und den 2. Summanden erzeugt.

b) Jetzt ist es leicht, mit einer geeigneten Summenschaltung den Wert des Produkts zu erzeugen:

Entwickle analog die Schaltung für einen 4-Bit-Multiplizierer
Beispiel: 1010*1101 (Verwende dazu 4-Bit-Volladdierer)

```
    1010
   0000
  1010
 1010
  1 1 1
-----------
 10000010
```

Rechengesetze, Rechenregeln und Kürzungsregeln der Booleschen Schaltalgebra

$A \wedge B = B \wedge A$	$A \vee B = B \vee A$	K-Gesetze
$(A \wedge B) \wedge C = A \wedge (B \wedge C)$	$(A \vee B) \vee C = A \vee (B \vee C)$	A-Gesetze
$A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$	$A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$	D-Gesetze
$\overline{A \wedge B} = \overline{A} \vee \overline{B}$	$\overline{A \vee B} = \overline{A} \wedge \overline{B}$	Inversionsgesetze
$A \wedge A = A$	$A \vee A = A$	R1
$A \wedge 1 = A$	$A \vee 1 = 1$	R2
$A \wedge 0 = 0$	$A \vee 0 = A$	R3
$A \vee (A \wedge B) = A$	$A \wedge (A \vee B) = A$	K1
$A \vee (\overline{A} \wedge B) = A \vee B$	$A \wedge (\overline{A} \vee B) = A \wedge B$	K2
$(A \wedge B) \vee (A \wedge \overline{B}) = A$	$(A \vee B) \wedge (A \vee \overline{B}) = A$	K3

Beweis zu K1:

$$\begin{aligned} A \vee (A \wedge B) &= (A \wedge 1) \vee (A \wedge B) && \text{R3} \\ &= A \wedge (1 \vee B) && \text{DG} \\ &= A \wedge 1 && \text{R3} \\ &= A && \text{R3} \end{aligned}$$

Aufgabe 11 : Beweise analog die Kürzungsregeln K2 und K3

Aufgabe 12: Vereinfache den Term aus Aufgabe 8 zu Z=0

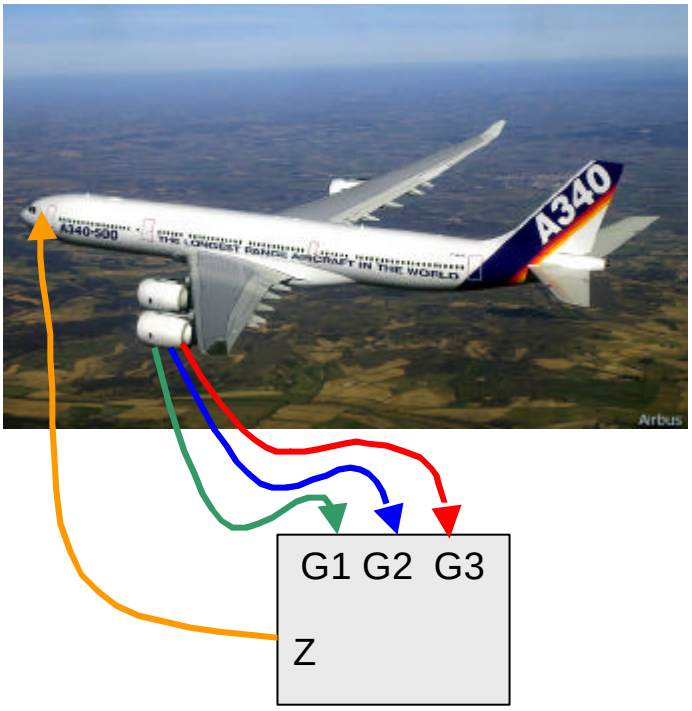
Aufgabe 13

Schaltfunktionen zu einer vorgegebenen Schalttabelle

Häufig wissen wir, wie eine Schaltung funktionieren soll und müssen eine Schaltung entwickeln, die diese Funktionalität besitzt.

Beispiel:

Bei einem Flugzeug sind an allen hochsensiblen Stellen drei Gefahrenmelder G1,G2, G3 angebracht. Alarm im Cockpit wird nur dann ausgelöst, wenn mindestens zwei dieser drei Melder Alarm anzeigen.



G1	G2	G3	Z	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	←
1	0	0	0	
1	0	1	1	←
1	1	0	1	←
1	1	1	1	←

Lösungsidee:
Ich konzentriere mich auf die Fälle, in denen Z den Wert 1 hat und nutze folgenden Sachverhalt aus:

Ein ODER-Baustein zeigt am Ausgang eine 1, wenn mindestens an Eingang eine einem 1 anliegt !

$Z = (\quad) \vee (\quad) \vee (\quad) \vee (\quad)$

Jetzt muss ich nur noch dafür sorgen, dass jede Klammer genau für den entsprechenden Fall gerade eine 1 erzeugt:
Ein UND-Baustein zeigt genau dann am Ausgang eine 1, wenn an allen Eingängen eine 1 anliegt !

1. Fall G1=0 G2=1 G3=1 $(\overline{G1} \wedge G2 \wedge G3) = 1$
2. Fall G1=1 G2=0 G3=1 $(G1 \wedge \overline{G2} \wedge G3) = 1$
3. Fall G1=1 G2=1 G3=0 $(G1 \wedge G2 \wedge \overline{G3}) = 1$
4. Fall G1=1 G2=1 G3=1 $(G1 \wedge G2 \wedge G3) = 1$

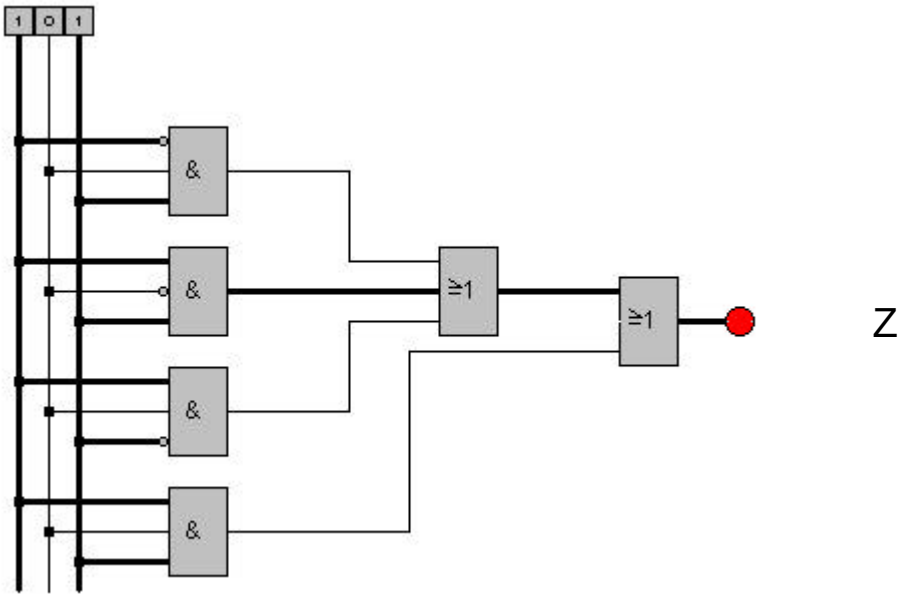
Also ist die Schaltfunktion

$$Z = (\overline{G1} \wedge G2 \wedge G3) \vee (G1 \wedge \overline{G2} \wedge G3) \vee (G1 \wedge G2 \wedge \overline{G3}) \vee (G1 \wedge G2 \wedge G3)$$

eine Lösung.

Man nennt sie auch **DISJUNKTIVE Normalform (DNF),** weil die ODER-Verknüpfung auch Disjunktion heisst.

Die zugehörige Schaltung sieht dann so aus:



G1G2G3

Die konjunktive Normalform

Konzentriert man sich auf die Nullen bei Z, so gelangen wir zur konjunktiven Normalform (KNF):

G1	G2	G3	Z	
0	0	0	0	←
0	0	1	0	←
0	1	0	0	←
0	1	1	1	
1	0	0	0	←
1	0	1	1	
1	1	0	1	
1	1	1	1	

Ein UND-Baustein zeigt am Ausgang eine 0, wenn mindestens an Eingang eine einem 0 anliegt !

$Z = (\quad) \wedge (\quad) \wedge (\quad) \wedge (\quad)$

Jetzt muss ich nur noch dafür sorgen, dass jede Klammer genau für den entsprechenden Fall gerade eine 0 erzeugt:
Ein ODER-Baustein zeigt genau dann am Ausgang eine 0, wenn an allen Eingängen eine 0 anliegt !

1. Fall G1=0 G2=0 G3=0 $(G1 \vee G2 \vee G3) = 0$
2. Fall G1=0 G2=0 G3=1 $(G1 \vee G2 \vee \overline{G3}) = 0$
3. Fall G1=1 G2=1 G3=0 $(G1 \vee \overline{G2} \vee G3) = 0$
4. Fall G1=1 G2=1 G3=1 $(\overline{G1} \vee G2 \vee G3) = 0$

$$Z = (G1 \vee G2 \vee G3) \wedge (G1 \wedge G2 \wedge \overline{G3}) \wedge (G1 \wedge \overline{G2} \wedge G3) \wedge (\overline{G1} \wedge G2 \wedge G3)$$

ist also ebenfalls eine Lösung, die sogenannte konjunktive Normalform.

Die UND-Verknüpfung wird auch Konjunktion genannt.