Boolesche Algebra

- Boolesche Algebra
  - Grundlage bei der Entwicklung von digitaler Elektronik.
  - Ebenfalls Grundlage aller logischer Verknüpfungen in Programmiersprachen.
  - Wird in allen modernen Programmiersprachen zur Verfügung gestellt.
  - Spezielle Form der Booleschen Algebra nennt sich Schaltalgebra und dient als Hilfsmittel zur Berechnung binärer Schaltnetze und Schaltwerke.

  - Boolesche Algebra = Rechnen mit Wahrheitswerten:
    - 0 = „false“ = falsch
    - 1 = „true“ = wahr

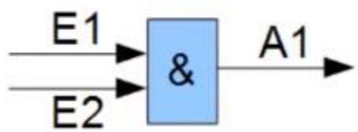
  - Operationen werden mit Hilfe von Wahrheitswerten definiert.

Boolesche Algebra - Logische Funktionen

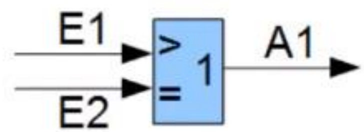
Bezeichnung	UND-Funktion (Konjunktion)	ODER-Funktion (Disjunktion)	NICHT-Funktion (Negation)

Schaltsymbol (DIN EN 60617)

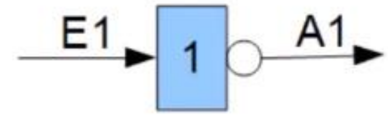
UND-Funktion:



ODER-Funktion:



NICHT-Funktion:



Funktionstabelle

UND-Funktion:

E1	0	0	1	1
E2	0	1	0	1
A1	0	0	0	1

ODER-Funktion:

E1	0	0	1	1
E2	0	1	0	1
A1	0	1	1	1

NICHT-Funktion:

E1	0	1
A1	1	0

Schaltalgebra

UND-Funktion: $A1 = E1 \wedge E2$

ODER-Funktion: $A1 = E1 \vee E2$

NICHT-Funktion: $A1 = \overline{E1}$

Boolesche Algebra - Axiome:

Kommutativgesetze	(1) $a \wedge b = b \wedge a$	(1') $a \vee b = b \vee a$
Assoziativgesetze	(2) $(a \wedge b) \wedge c = a \wedge (b \wedge c)$	(2') $(a \vee b) \vee c = a \vee (b \vee c)$
Idempotenzgesetze	(3) $a \wedge a = a$	(3') $a \vee a = a$
Distributivgesetze	(4) $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$	(4') $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$
Neutralitätsgesetze	(5) $a \wedge 1 = a$	(5') $a \vee 0 = a$
Extremalgesetze	(6) $a \wedge 0 = 0$	(6') $a \vee 1 = 1$
Doppelnegationsgesetz	(7) $\neg(\neg a) = a$	
De Morgansche Gesetze	(8) $\neg(a \wedge b) = \neg a \vee \neg b$	(8') $\neg(a \vee b) = \neg a \wedge \neg b$
Komplementärgesetze	(9) $a \wedge \neg a = 0$	(9') $a \vee \neg a = 1$
Dualitätsgesetze	(10) $\neg 0 = 1$	(10') $\neg 1 = 0$
Absorptionsgesetze	(11) $a \vee (a \wedge b) = a$	(11') $a \wedge (a \vee b) = a$

- Boolesche Algebra - Axiome:

Kürzungsregeln:

$$R1 : x1 \vee (x1 \wedge x2) = x1$$

$$R2 : x1 \wedge (x1 \vee x2) = x1$$

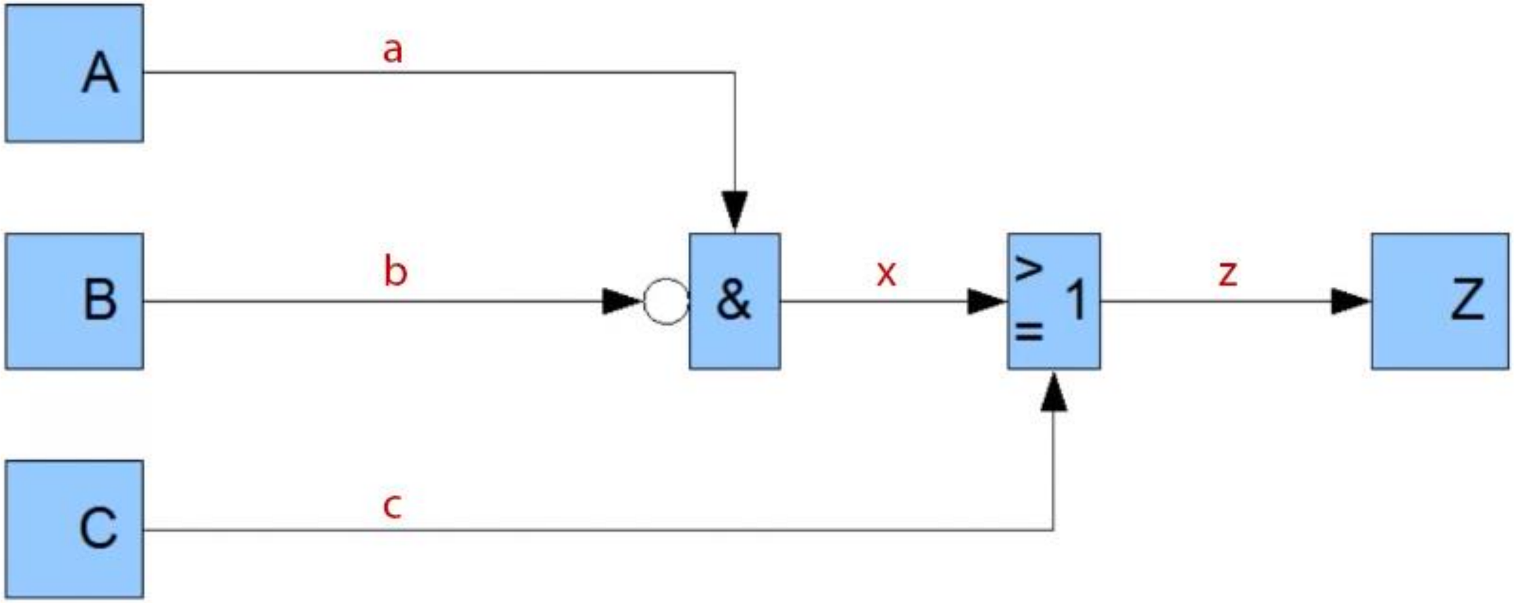
$$R3 : x1 \vee (\overline{x1} \wedge x2) = x1 \vee x2$$

$$R4 : x1 \wedge (\overline{x1} \vee x2) = x1 \wedge x2$$

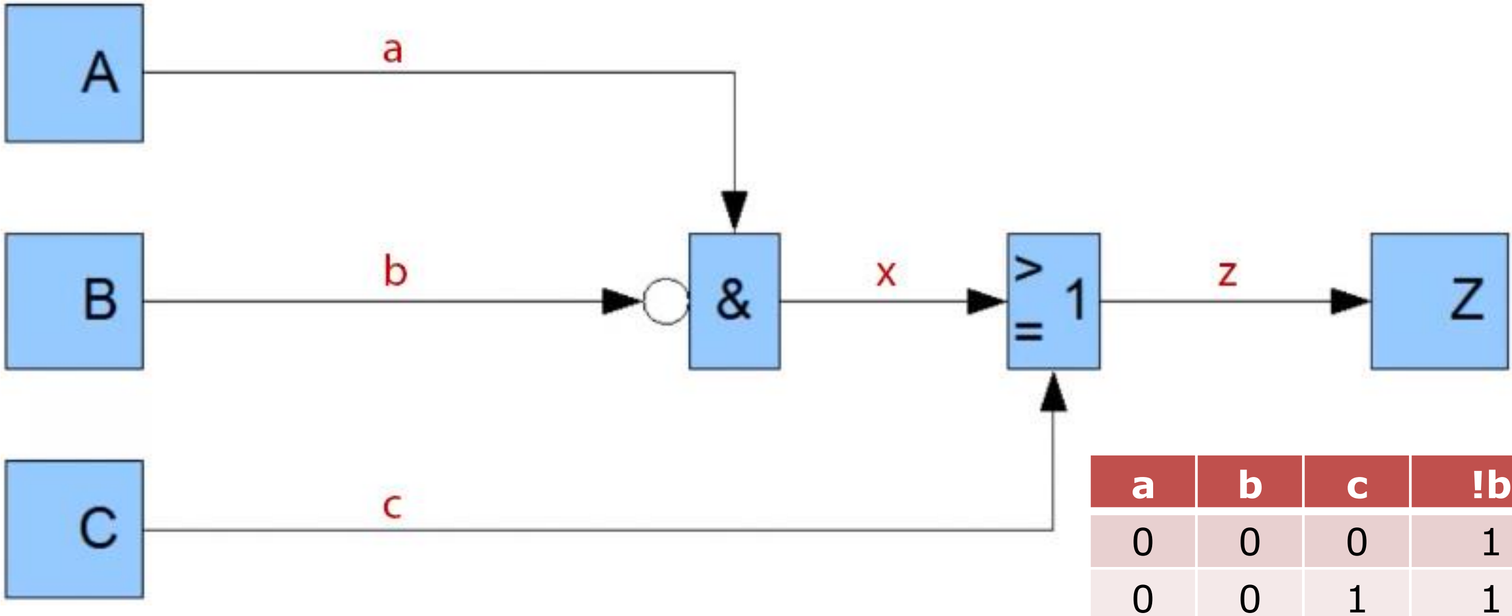
$$R5 : (x1 \wedge x2) \vee (x1 \wedge \overline{x2}) = x1$$

$$R6 : (x1 \vee x2) \wedge (x1 \vee \overline{x2}) = x1$$

► Boolesche Algebra - Schaltungsanalyse:

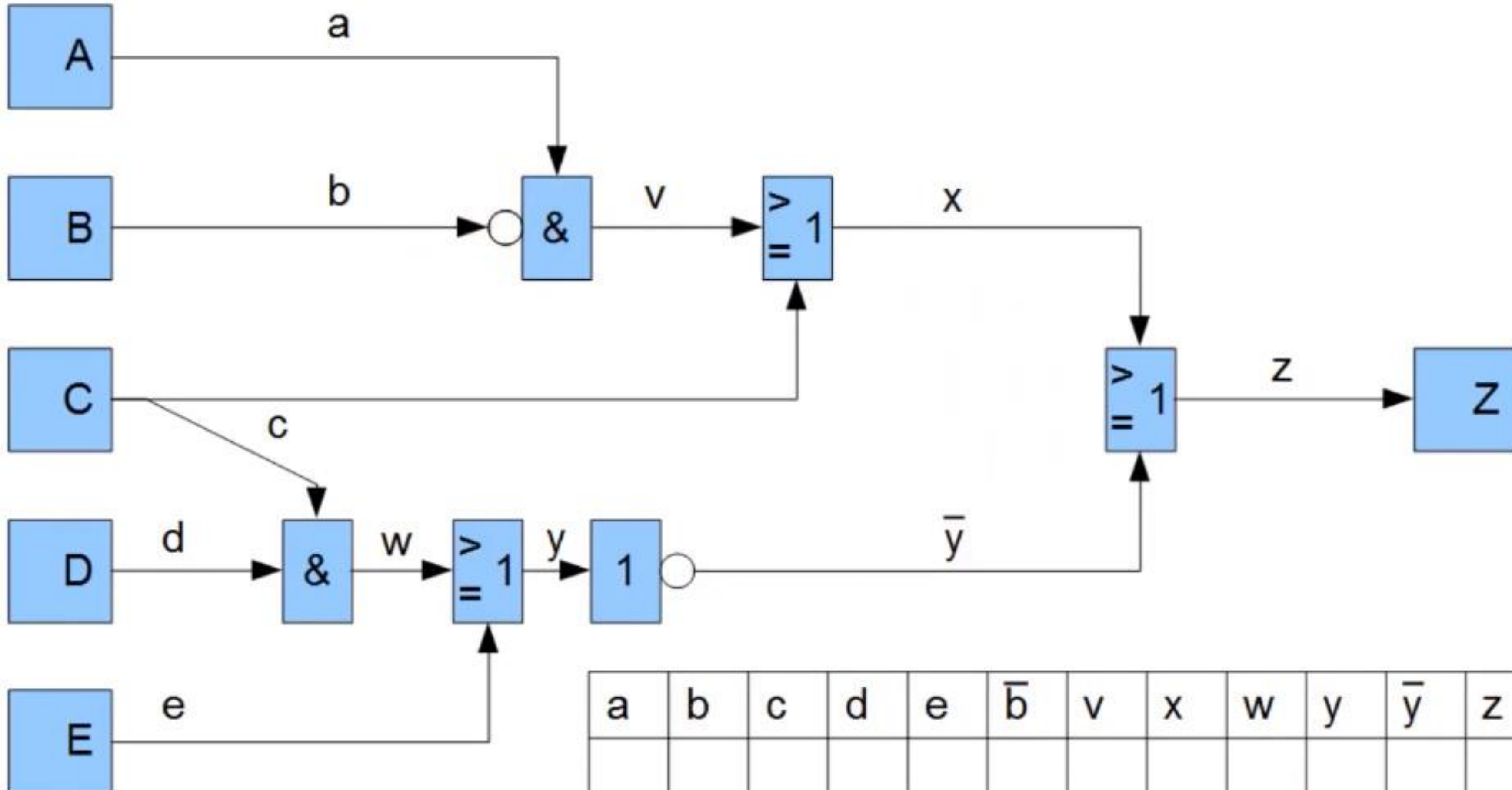


- Boolesche Algebra - Schaltungsanalyse:



a	b	c	!b	x	z
0	0	0	1	0	0
0	0	1	1	0	1
0	1	0	0	0	0
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	0	0	1

▶ Boolesche Algebra – Schaltungsanalyse – Übungsbeispiel:



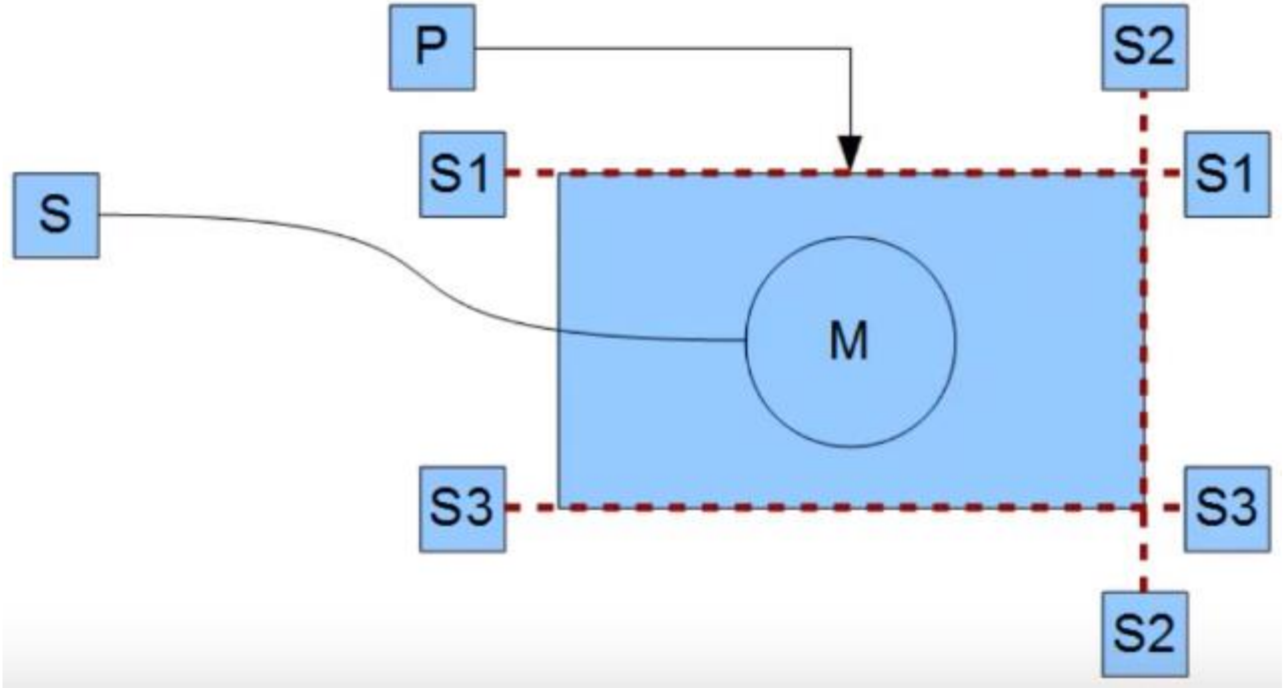
a	b	c	d	e	$\bar{b}$	v	x	w	y	$\bar{y}$	z

Boolesche Algebra – Schaltungsanalyse – Übungsbeispiel:

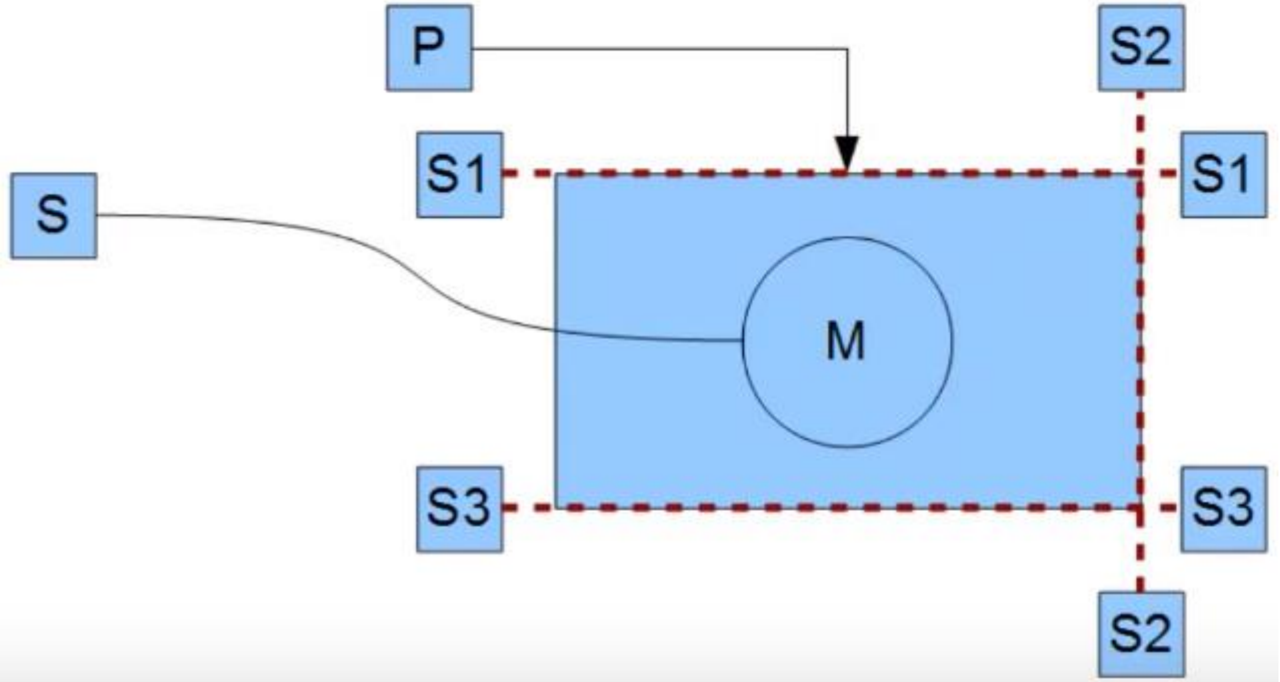
a	b	c	d	e	!b	v	x	w	y	!y	z
0	0	0	0	0	1	0	0	0	0	1	1
0	0	0	0	1	1	0	0	0	1	0	0
0	0	0	1	0	1	0	0	0	0	1	1
0	0	0	1	1	1	0	0	0	1	0	0
0	0	1	0	0	1	0	1	0	0	1	1
0	0	1	0	1	1	0	1	0	1	0	1
0	0	1	1	0	1	0	1	1	1	0	1
0	0	1	1	1	1	0	1	1	1	0	1
0	1	0	0	0	0	0	0	0	0	1	1
0	1	0	0	1	0	0	0	0	1	0	0
0	1	0	1	0	0	0	0	0	0	1	1
0	1	0	1	1	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	1	1
0	1	1	0	1	0	0	1	0	1	0	1
0	1	1	1	0	0	0	1	1	1	0	1
0	1	1	1	1	0	0	1	1	1	0	1

a	b	c	d	e	!b	v	x	w	y	!y	z
1	0	0	0	0	1	1	1	0	0	1	1
1	0	0	0	1	1	1	1	0	1	0	1
1	0	0	1	0	1	1	1	0	0	1	1
1	0	0	1	1	1	1	1	0	1	0	1
1	0	1	0	0	1	1	1	0	0	1	1
1	0	1	0	1	1	1	1	0	1	0	1
1	0	1	1	0	1	1	1	1	1	0	1
1	0	1	1	1	1	1	1	1	1	0	1
1	1	0	0	0	0	0	0	0	0	1	1
1	1	0	0	1	0	0	0	0	1	0	0
1	1	0	1	0	0	0	0	0	0	1	1
1	1	0	1	1	0	0	0	0	1	0	0
1	1	1	0	0	0	0	1	0	0	1	1
1	1	1	0	1	0	0	1	0	1	0	1
1	1	1	1	0	0	0	1	1	1	0	1
1	1	1	1	1	0	0	1	1	1	0	1

- Boolesche Algebra – Übung 1 – Teil 1: Wahrheitstabelle:
  - In einer Maschine soll abhängig von einem Schaltsignal ein Antrieb ein- bzw. ausgeschaltet werden.
  - Zusätzlich soll der Antrieb abschalten, wenn mindestens zwei von drei Lichtschranken ausgelöst werden.



Boolesche Algebra – Übung 1 – Teil 1: Wahrheitstabelle



P	S	S1	S2	S3	M
1	0	0	0	0	0
1	0	0	0	1	0
1	0	0	1	0	0
1	0	0	1	1	0
1	0	1	0	0	0
1	0	1	0	1	0
1	0	1	1	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	0	0	1	1
1	1	0	1	0	1
1	1	0	1	1	0
1	1	1	0	0	1
1	1	1	0	1	0
1	1	1	1	0	0
1	1	1	1	1	0

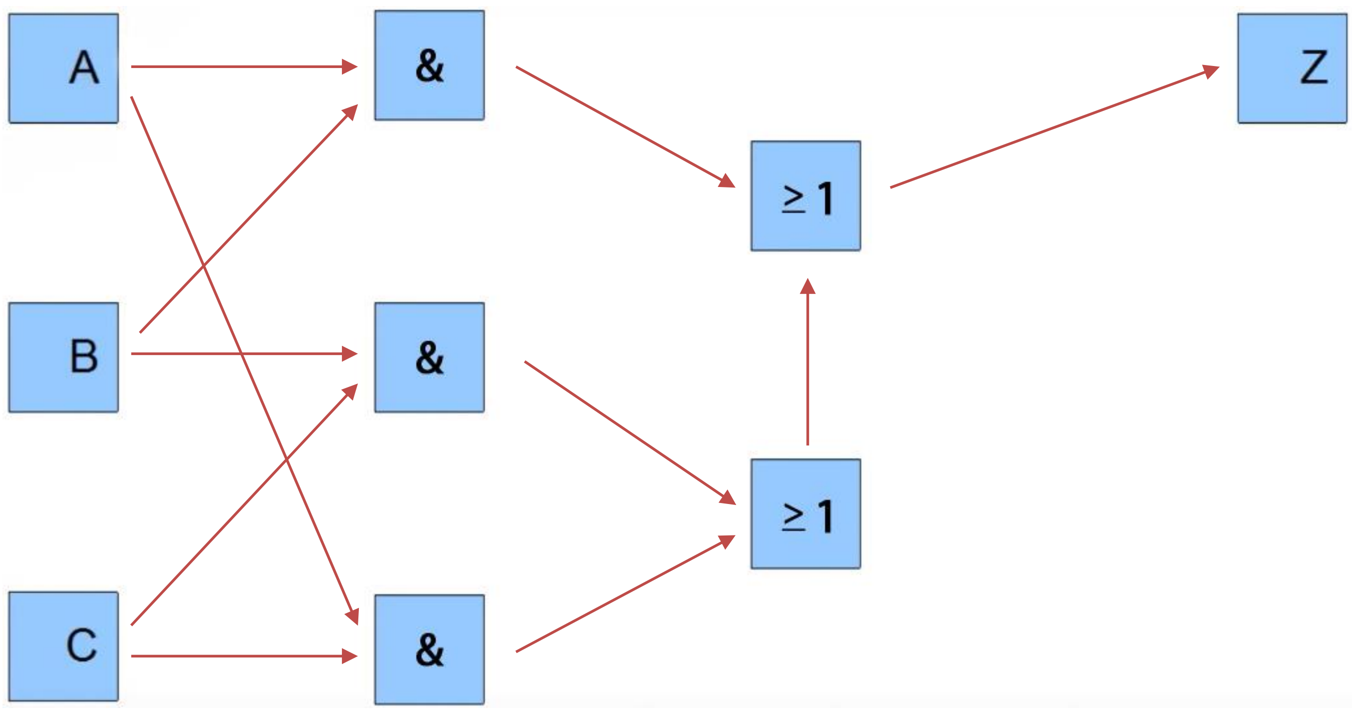
Boolesche Algebra – Übung 1 – Teil 2: Schaltungsanalyse

- Entwickeln Sie das zur ermittelten Wahrheitstabelle gehörige Schaltungsbild nur für die Eingänge **S1**, **S2**, **S3** sowie den Ausgang **Z**.

P	S	S1	S2	S3	M
1	0	0	0	0	0
1	0	0	0	1	0
1	0	0	1	0	0
1	0	0	1	1	0
1	0	1	0	0	0
1	0	1	0	1	0
1	0	1	1	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	0	0	1	1
1	1	0	1	0	1
1	1	0	1	1	0
1	1	1	0	0	1
1	1	1	0	1	0
1	1	1	1	0	0
1	1	1	1	1	0

Boolesche Algebra – Übung 1 – Teil 2: Schaltungsanalyse

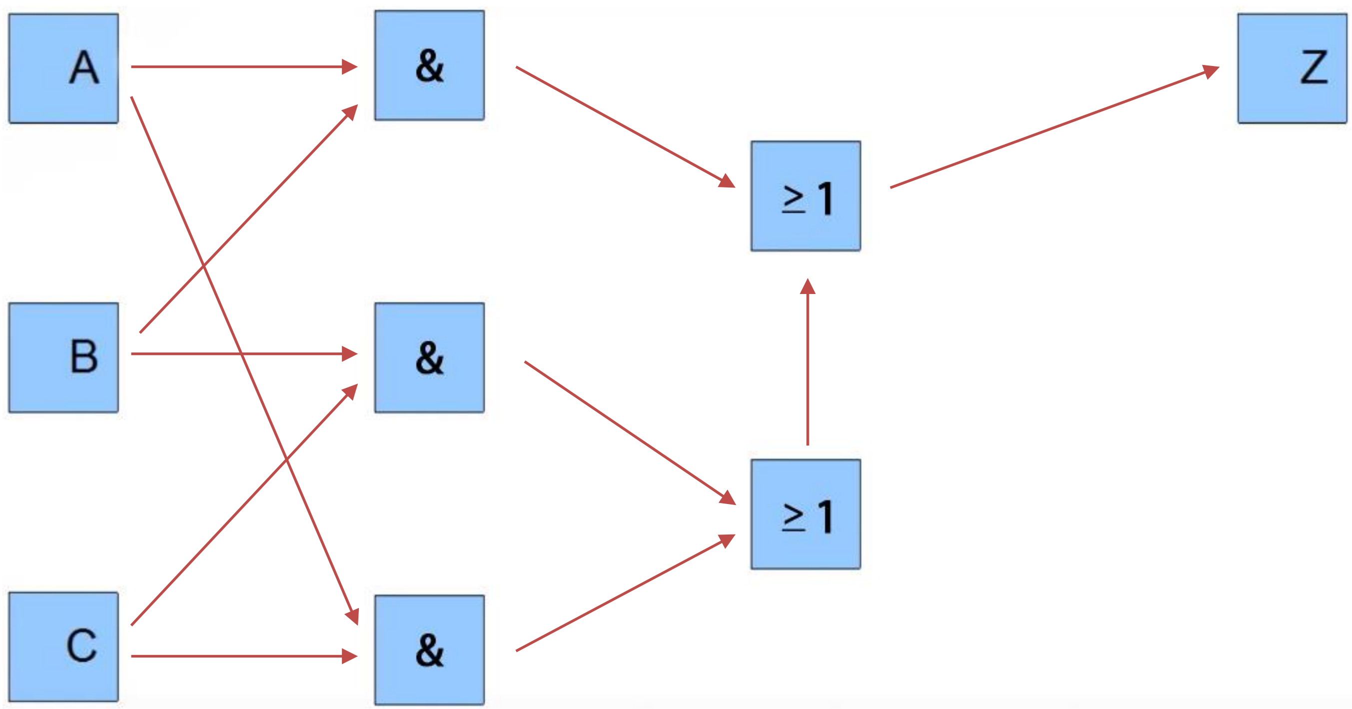
- Entwickeln Sie das zur ermittelten Wahrheitstabelle gehörige Schaltungsbild nur für die Eingänge **S1**, **S2**, **S3** sowie den Ausgang **Z**.



P	S	S1	S2	S3	M
1	0	0	0	0	0
1	0	0	0	1	0
1	0	0	1	0	0
1	0	0	1	1	0
1	0	1	0	0	0
1	0	1	0	1	0
1	0	1	1	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	0	0	1	1
1	1	0	1	0	1
1	1	0	1	1	0
1	1	1	0	0	1
1	1	1	0	1	0
1	1	1	1	0	0
1	1	1	1	1	0

Boolesche Algebra – Übung 1 – Teil 3: Schaltalgebra

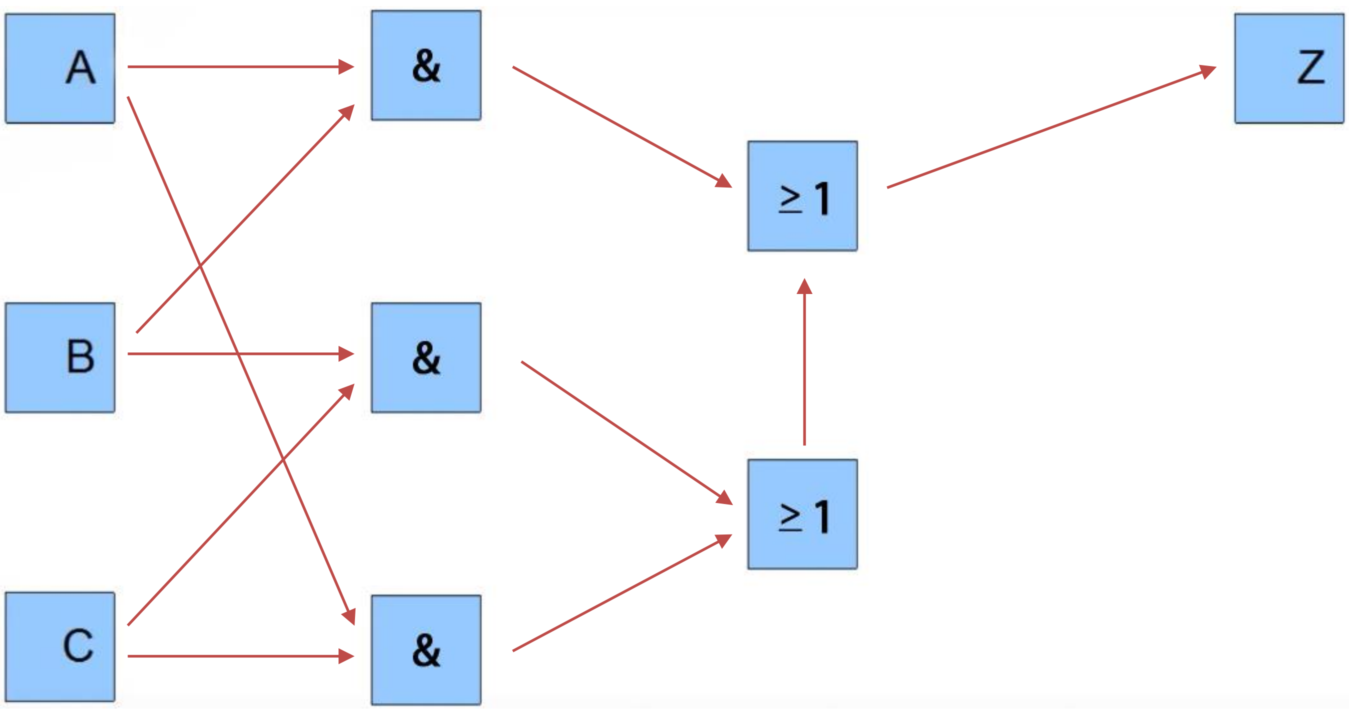
- Entwickeln Sie die zum erstellten Schaltungsbild gehörige Schaltalgebra.



P	S	S1	S2	S3	M
1	0	0	0	0	0
1	0	0	0	1	0
1	0	0	1	0	0
1	0	0	1	1	0
1	0	1	0	0	0
1	0	1	0	1	0
1	0	1	1	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	0	0	1	1
1	1	0	1	0	1
1	1	0	1	1	0
1	1	1	0	0	1
1	1	1	0	1	0
1	1	1	1	0	0
1	1	1	1	1	0

- Z = …

- Boolesche Algebra – Übung 1 – Teil 3: Schaltalgebra
  - Entwickeln Sie die zum erstellten Schaltungsbild gehörige Schaltalgebra.



P	S	S1	S2	S3	M
1	0	0	0	0	0
1	0	0	0	1	0
1	0	0	1	0	0
1	0	0	1	1	0
1	0	1	0	0	0
1	0	1	0	1	0
1	0	1	1	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	0	0	1	1
1	1	0	1	0	1
1	1	0	1	1	0
1	1	1	0	0	1
1	1	1	0	1	0
1	1	1	1	0	0
1	1	1	1	1	0

- Z = (A ^ B) v ((B ^ C) v (A ^ C))

- Boolesche Algebra – Übung 1 – Teil 4: Vereinfachung der Schaltalgebra
  - Vereinfachen Sie die im vorherigen Schritt entwickelte Gleichung nach den Grundsätzen der Booleschen Algebra.

  - Z = (A ^ B) v ((B ^ C) v (A ^ C))

(1)	$a \wedge b = b \wedge a$	(1')	$a \vee b = b \vee a$
(2)	$(a \wedge b) \wedge c = a \wedge (b \wedge c)$	(2')	$(a \vee b) \vee c = a \vee (b \vee c)$
(3)	$a \wedge a = a$	(3')	$a \vee a = a$
(4)	$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$	(4')	$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$
(5)	$a \wedge 1 = a$	(5')	$a \vee 0 = a$
(6)	$a \wedge 0 = 0$	(6')	$a \vee 1 = 1$
(7)	$\neg(\neg a) = a$		
(8)	$\neg(a \wedge b) = \neg a \vee \neg b$	(8')	$\neg(a \vee b) = \neg a \wedge \neg b$
(9)	$a \wedge \neg a = 0$	(9')	$a \vee \neg a = 1$
(10)	$\neg 0 = 1$	(10')	$\neg 1 = 0$
(11)	$a \vee (a \wedge b) = a$	(11')	$a \wedge (a \vee b) = a$

Boolesche Algebra – Übung 1 – Teil 4: Vereinfachung der Schaltalgebra

- Vereinfachen Sie die im vorherigen Schritt entwickelte Gleichung nach den Grundsätzen der Booleschen Algebra.

- Z = (A ^ B) v ((B ^ C) v (A ^ C))
- Z = (A ^ B) v (B ^ C) v (A ^ C)
- Z = (A ^ B) v (A ^ C) v (B ^ C)
- **Z = (A ^ (B v C)) v (B ^ C)**

(1) $a \wedge b = b \wedge a$	(1') $a \vee b = b \vee a$
(2) $(a \wedge b) \wedge c = a \wedge (b \wedge c)$	(2') $(a \vee b) \vee c = a \vee (b \vee c)$
(3) $a \wedge a = a$	(3') $a \vee a = a$
(4) $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$	(4') $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$
(5) $a \wedge 1 = a$	(5') $a \vee 0 = a$
(6) $a \wedge 0 = 0$	(6') $a \vee 1 = 1$
(7) $\neg(\neg a) = a$	
(8) $\neg(a \wedge b) = \neg a \vee \neg b$	(8') $\neg(a \vee b) = \neg a \wedge \neg b$
(9) $a \wedge \neg a = 0$	(9') $a \vee \neg a = 1$
(10) $\neg 0 = 1$	(10') $\neg 1 = 0$
(11) $a \vee (a \wedge b) = a$	(11') $a \wedge (a \vee b) = a$

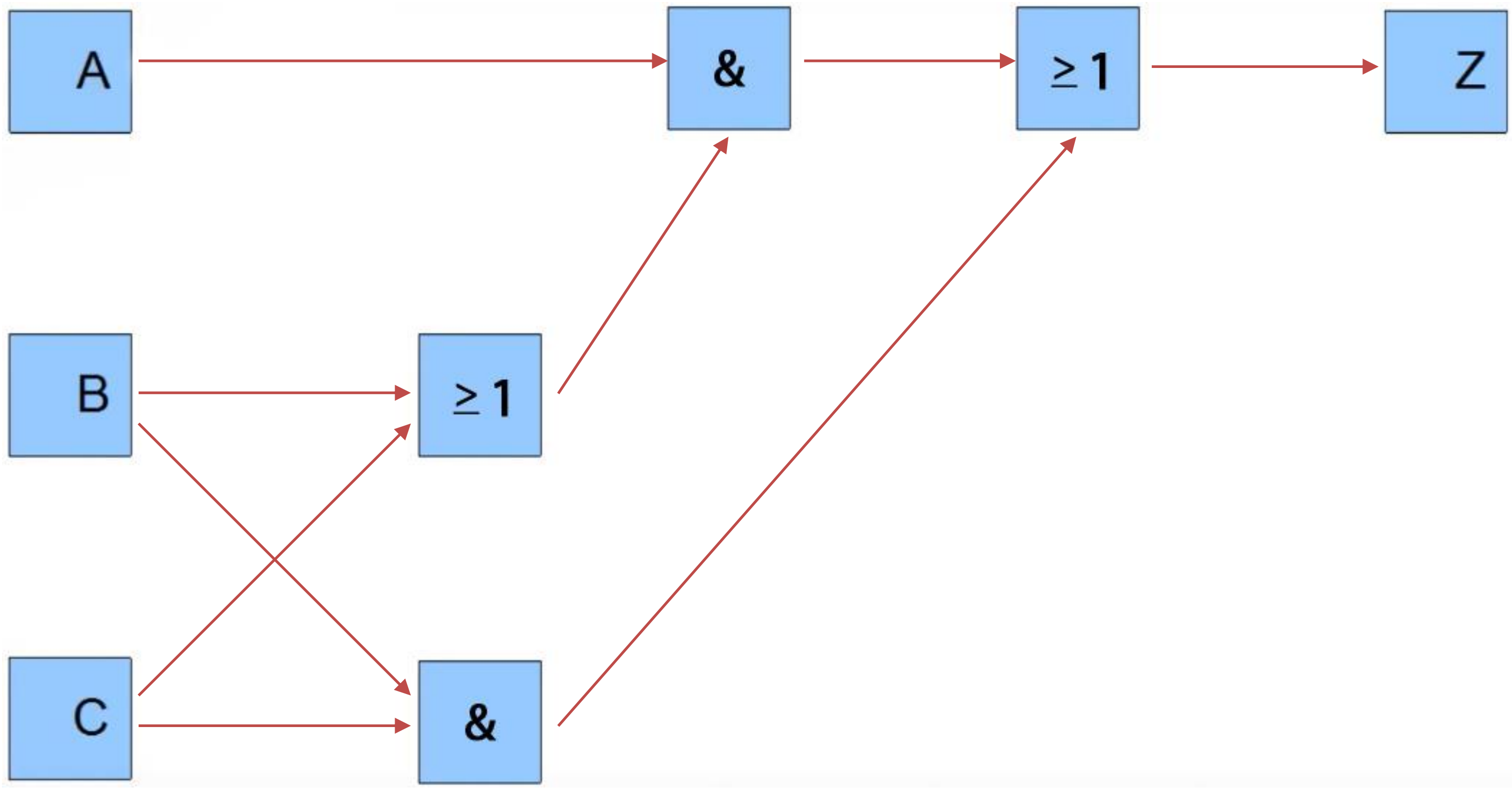
- Boolesche Algebra – Übung 1 – Teil 5: Schaltung vereinfachen
  - Erstellen Sie die zugehörige Schaltung zur vereinfachten Schaltungsgleichung.

  - **Z = (A ^ (B v C)) v (B ^ C)**

Boolesche Algebra – Übung 1 – Teil 5: Schaltung vereinfachen

- Erstellen Sie die zugehörige Schaltung zur vereinfachten Schaltungsgleichung.

- **Z = (A ^ (B v C)) v (B ^ C)**

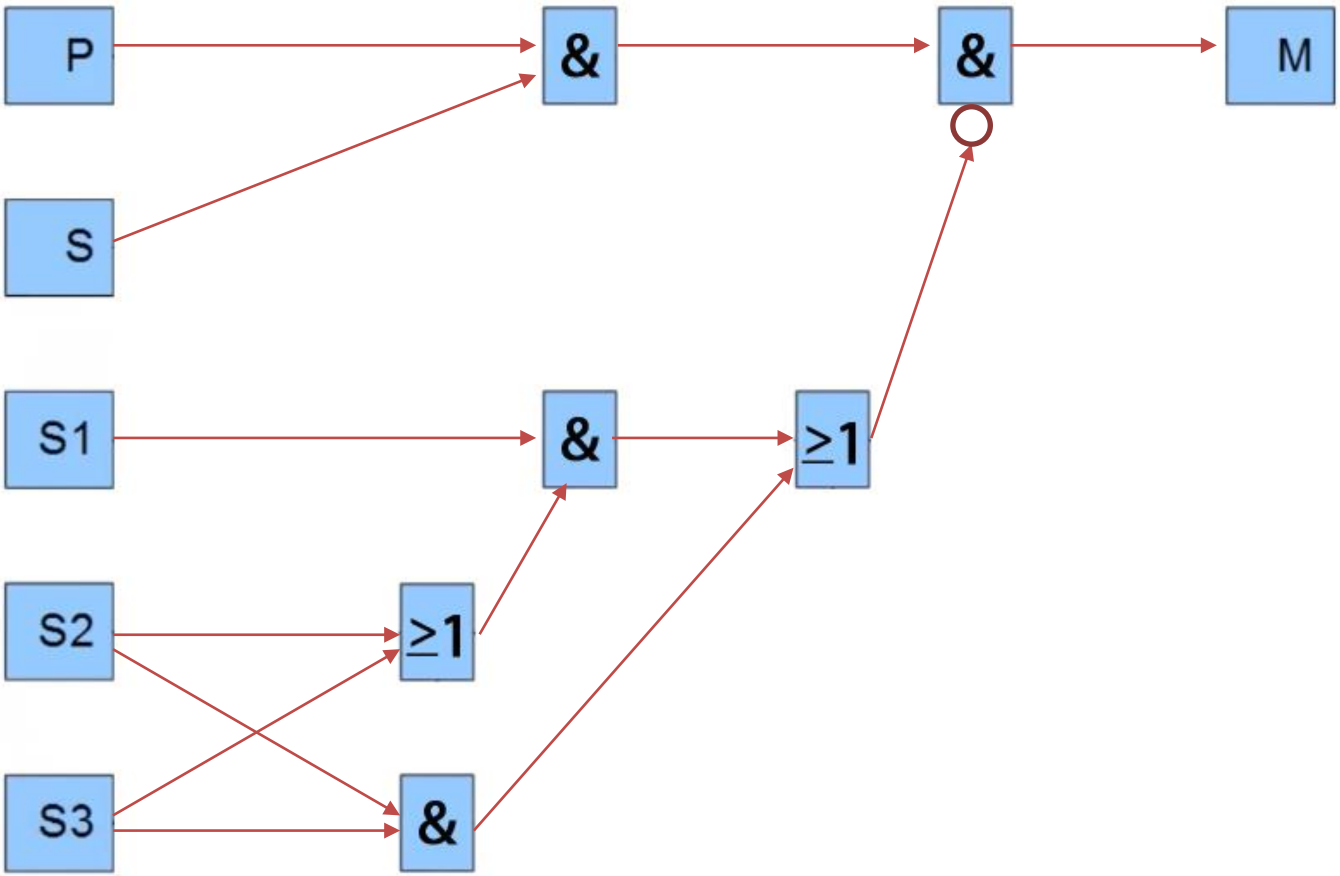


Boolesche Algebra – Übung 1 – Teil 6: Schaltung vervollständigen

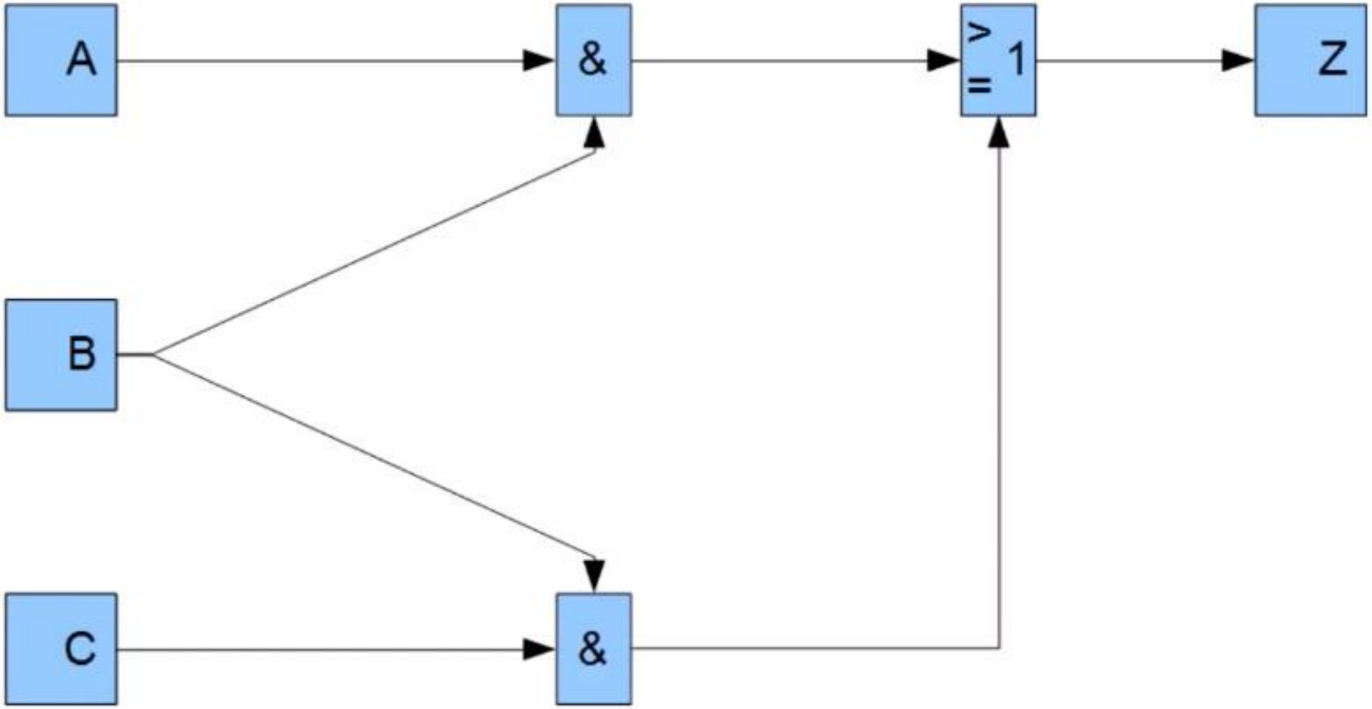
- Vervollständigen Sie die vereinfachte Schaltung um den kompletten Funktionsumfang der Maschine, wie in der Wahrheitstabelle dargestellt.

P	S	S1	S2	S3	M
1	0	0	0	0	0
1	0	0	0	1	0
1	0	0	1	0	0
1	0	0	1	1	0
1	0	1	0	0	0
1	0	1	0	1	0
1	0	1	1	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	0	0	1	1
1	1	0	1	0	1
1	1	0	1	1	0
1	1	1	0	0	1
1	1	1	0	1	0
1	1	1	1	0	0
1	1	1	1	1	0

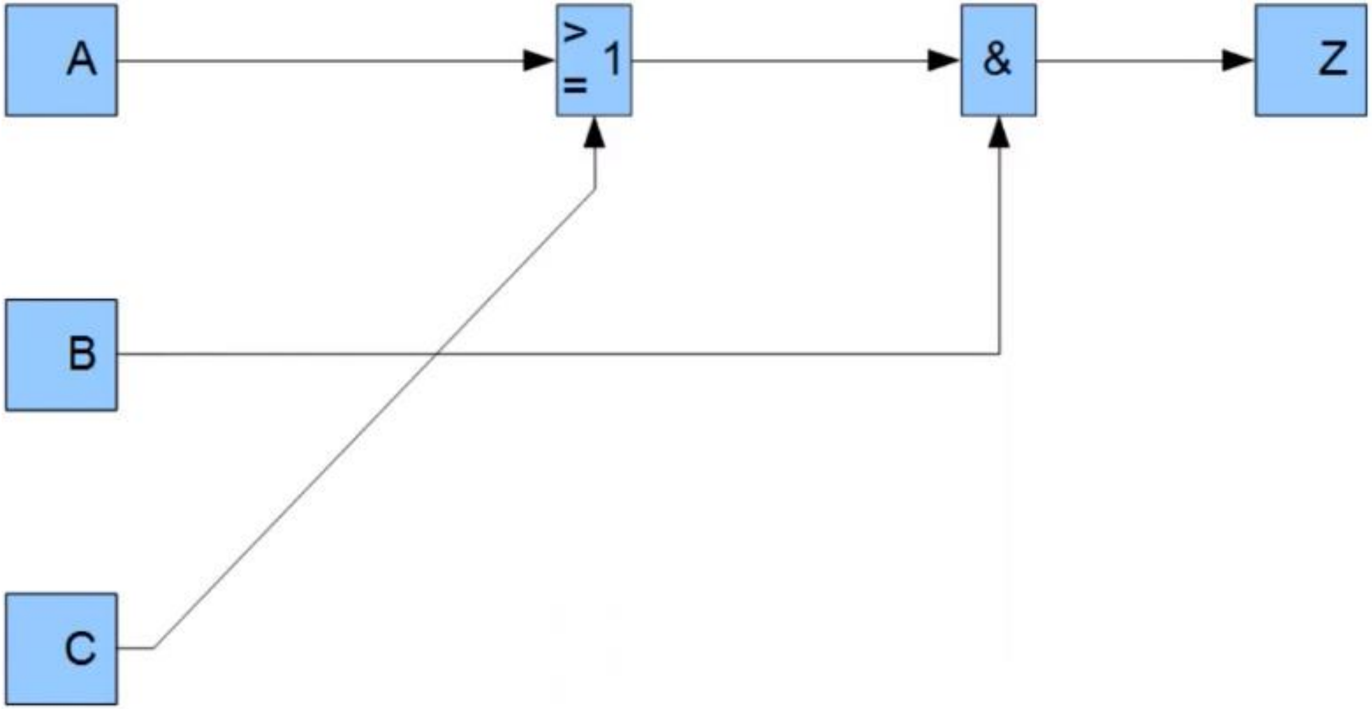
- Boolesche Algebra – Übung 1 – Teil 6: Schaltung vervollständigen
  - Vervollständigen Sie die vereinfachte Schaltung um den kompletten Funktionsumfang der Maschine, wie in der Wahrheitstabelle dargestellt.



- Boolesche Algebra – Übung 2
  - Entwickeln sie die Schaltalgebra zur dargestellten Schaltung, vereinfachen Sie diese & erstellen Sie daraus das ebenfalls vereinfachte Schaltungsbild.

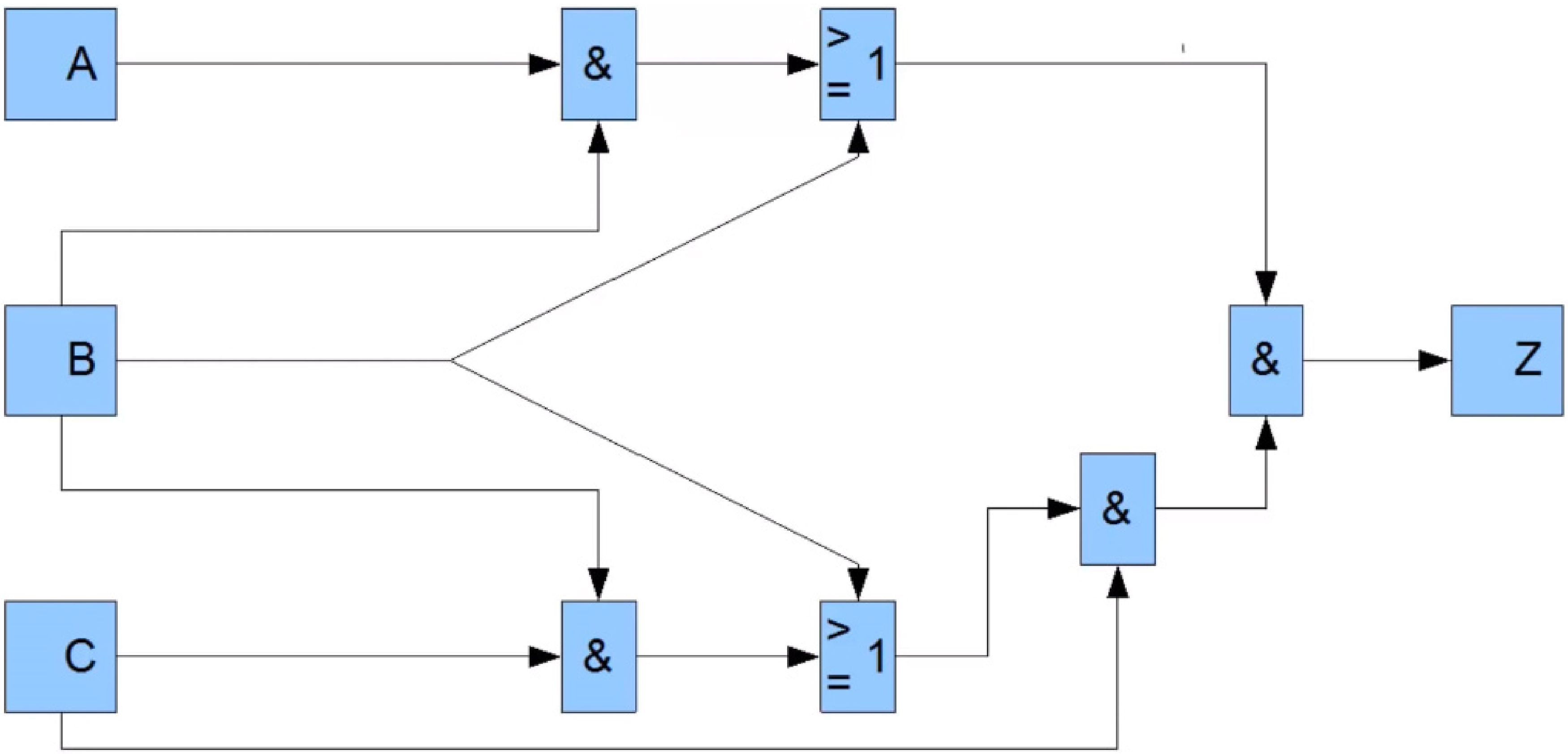


- Boolesche Algebra – Übung 2
  - Entwickeln sie die Schaltalgebra zur dargestellten Schaltung, vereinfachen Sie diese & erstellen Sie daraus das ebenfalls vereinfachte Schaltungsbild.



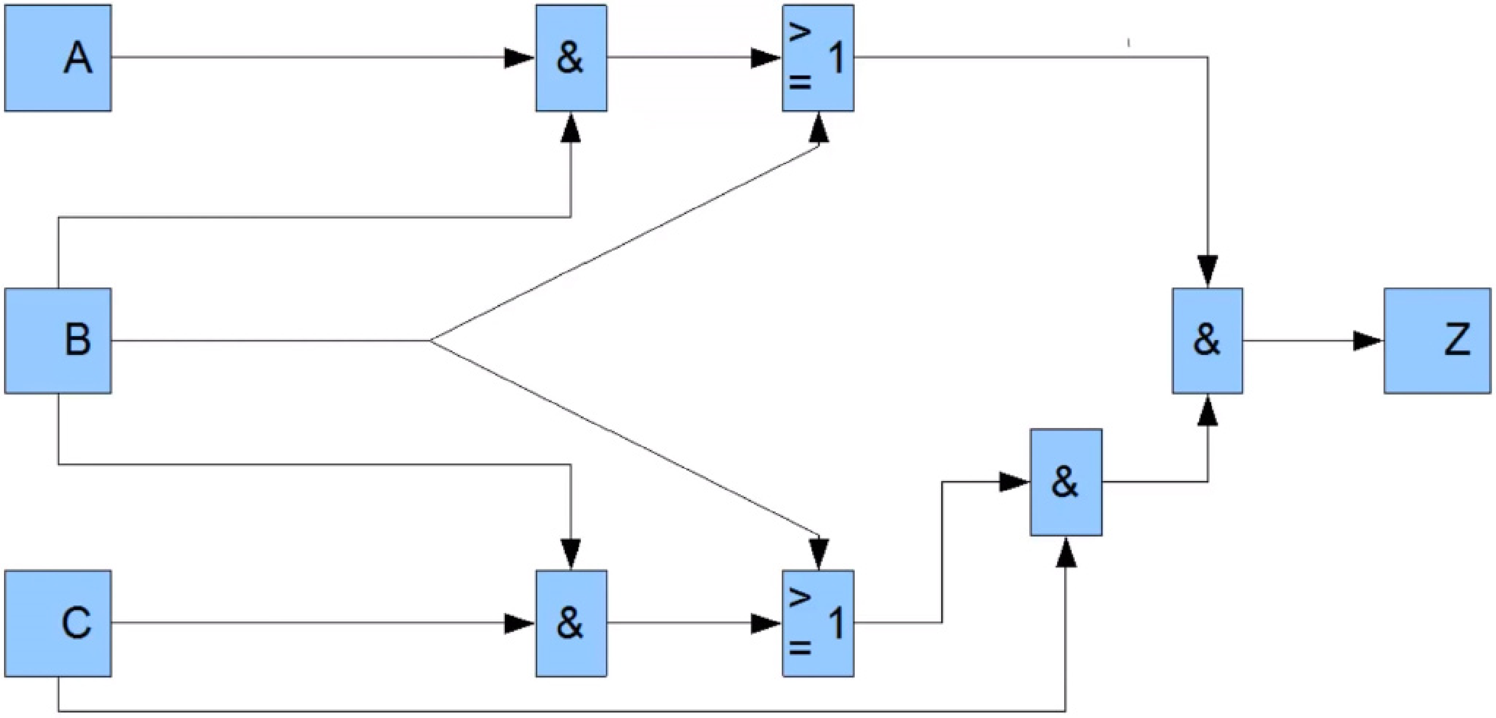
- Z = (A ^ B) v (B ^ C)
  <u>Z = B ^ (A v C)</u>

- Boolesche Algebra – Übung 3
  - Entwickeln sie die Schaltalgebra zur dargestellten Schaltung, vereinfachen Sie diese & erstellen Sie daraus das ebenfalls vereinfachte Schaltungsbild.



Boolesche Algebra – Übung 3

- Entwickeln sie die Schaltalgebra zur dargestellten Schaltung, vereinfachen Sie diese & erstellen Sie daraus das ebenfalls vereinfachte Schaltungsbild.



- **((A ^ B) v B) ^ (((B ^ C) v B) ^ C)**

Boolesche Algebra – Übung 3

- Entwickeln sie die Schaltalgebra zur dargestellten Schaltung, vereinfachen Sie diese & erstellen Sie daraus das ebenfalls vereinfachte Schaltungsbild.

- ((A ^ B) v B) ^ (((B ^ C) v B) ^ C)

Kürzungsregeln:

$R1 : x1 \vee (x1 \wedge x2) = x1$

$R2 : x1 \wedge (x1 \vee x2) = x1$

$R3 : x1 \vee (\overline{x1} \wedge x2) = x1 \vee x2$

$R4 : x1 \wedge (\overline{x1} \vee x2) = x1 \wedge x2$

$R5 : (x1 \wedge x2) \vee (x1 \wedge \overline{x2}) = x1$

$R6 : (x1 \vee x2) \wedge (x1 \vee \overline{x2}) = x1$

(1)	$a \wedge b = b \wedge a$	(1')	$a \vee b = b \vee a$
(2)	$(a \wedge b) \wedge c = a \wedge (b \wedge c)$	(2')	$(a \vee b) \vee c = a \vee (b \vee c)$
(3)	$a \wedge a = a$	(3')	$a \vee a = a$
(4)	$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$	(4')	$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$
(5)	$a \wedge 1 = a$	(5')	$a \vee 0 = a$
(6)	$a \wedge 0 = 0$	(6')	$a \vee 1 = 1$
(7)	$\neg(\neg a) = a$		
(8)	$\neg(a \wedge b) = \neg a \vee \neg b$	(8')	$\neg(a \vee b) = \neg a \wedge \neg b$
(9)	$a \wedge \neg a = 0$	(9')	$a \vee \neg a = 1$
(10)	$\neg 0 = 1$	(10')	$\neg 1 = 0$
(11)	$a \vee (a \wedge b) = a$	(11')	$a \wedge (a \vee b) = a$

- Boolesche Algebra – Übung 3
  - Entwickeln sie die Schaltalgebra zur dargestellten Schaltung, vereinfachen Sie diese & erstellen Sie daraus das ebenfalls vereinfachte Schaltungsbild.

  - ((A ^ B) v B) ^ (((B ^ C) v B) ^ C)

    B ^ (((B ^ C) v B) ^ C)
    B ^ (B ^ C)
    B ^ B ^ C

  - **B ^ C**

Boolesche Algebra – Übung 3

- Entwickeln sie die Schaltalgebra zur dargestellten Schaltung, vereinfachen Sie diese & erstellen Sie daraus das ebenfalls vereinfachte Schaltungsbild.

Boolesche Algebra – Hausaufgabe „Die fünf Weisen“

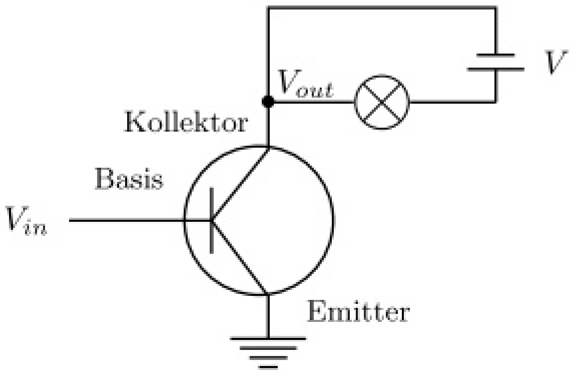
- Zur Vereinfachung von Abstimmungsverfahren haben die fünf Weisen auf ein elektronisches Verfahren umgestellt. Bei einfachen Ja/Nein-Entscheidungen kann jeder für Ja einen Knopf drücken und für Nein drückt er nicht. Als Ergebnis leuchtet eine Lampe (Ja) oder eben nicht (Nein). Zwei der Weisen sind besonders weise, wenn sie beide für bzw. gegen eine Entscheidung stimmen, überstimmen sie alle anderen. Sind diese beiden sich nicht einig, entscheidet die Mehrheit.

- Modellieren Sie dieses Problem als Schaltung, schreiben Sie den Term auf, vereinfachen diesen und geben die vereinfachte Schaltung an.

Digitale Schaltungen

- Gatter (engl.: Gate):
  - Elementarer Baustein einer digitalen Schaltung für zweiwertige Signale.
  - Mit Hilfe von Gattern werden in Rechnern grundlegende logische Funktionen implementiert (AND-Gatter, OR-Gatter, …).
  - Basierend auf der Transistortechnologie.

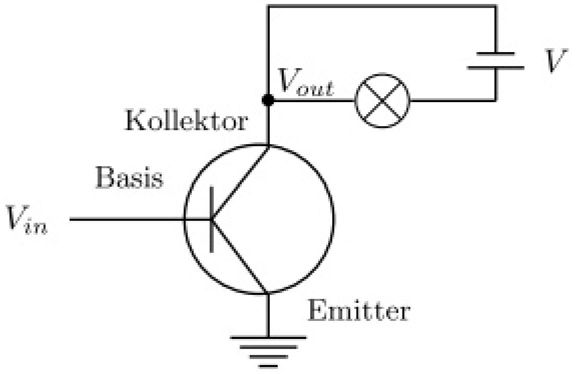
  - Grundprinzip einer Transistorschaltung:



    - Ein Transistor verfügt über 3 Ein-, bzw. Ausgänge: Kollektor, Basis & Emitter.
    - Liegt die Eingangsspannung V_{in} unter einem kritischen Wert, schaltet sich der Transistor aus und stellt de facto einen unendlichen Wiederstand dar.
    - In diesem Zustand nimmt die Ausgangsspannung V_{out} den Wert V an, der Stromkreis oben rechts ist geschlossen → die Lampe brennt.
    - Überschreitet V_{in} einen kritischen Wert, schaltet sich der Transistor ein, der Stromkreis wird geerdet, d.h. V_{out} wird auf Masse (~ 0 Volt) geschaltet → die Lampe ist aus.

Gatter (engl.: Gate) – Beispiel 1:

- Diese Tatsache impliziert folgende Schaltungslogik:
  - Wenn V_{in} niedrig → V_{out} hoch
  - Wenn V_{in} hoch → V_{out} niedrig

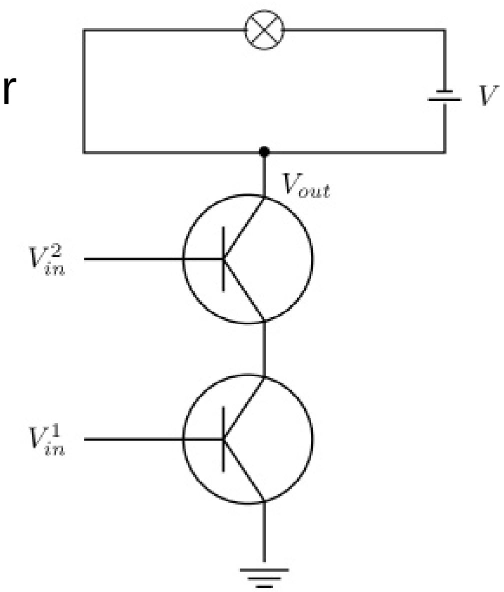


- Daraus folgt, dass ein Transistor nichts anderes als einen Inverter darstellt, der eine logische 0 in eine 1 transformiert und eine logische 1 in eine 0.

- Bezeichnen wir das Signal der Eingangsspannung V_{in} als Boolesche Variable *a* und die Ausgangsspannung V_{out} als Boolesche Variable *b*, ergibt sich die folgende Schaltungslogik:

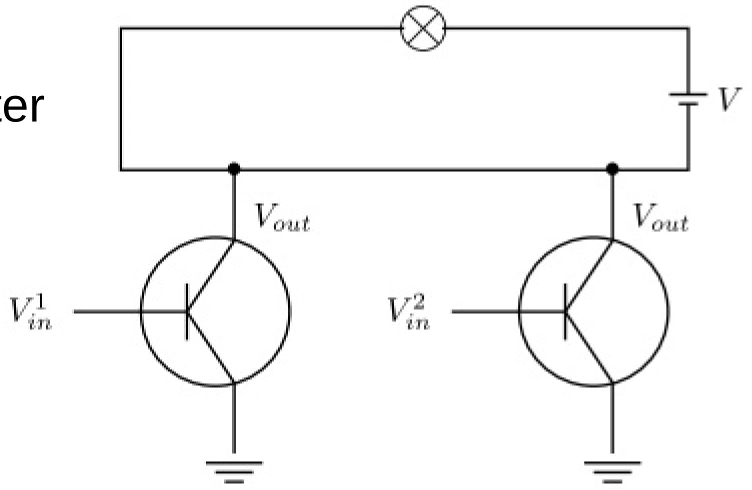
a	***b***
0	1
1	0

- Gatter (engl.: Gate) – Beispiel 2:
  - Rechts ein Beispiel zweier in Reihe (seriell) geschalteter Transistoren:
  - Hier gilt folgende Schaltungslogik:
    - Ist V_{in}^1 ***und*** V_{in}^2 hoch → V_{out} niedrig
    - Ist V_{in}^1 hoch, V_{in}^2 niedrig → V_{out} hoch
    - Ist V_{in}^1 niedrig, V_{in}^2 hoch → V_{out} hoch
    - Ist V_{in}^1 ***und*** V_{in}^2 niedrig → V_{out} hoch
  - Setzen wir wieder a ← V_{in}^1, b ← V_{in}^2, und c ← V_{out}, dann lässt sich die Reihenschaltung zweier Transistoren durch folgende Schaltungslogik beschreiben:
  - Dies ist die Schalttabelle der Booleschen NAND–Funktion. Zwei in Reihe geschaltete Transistoren realisieren daher ein NAND–Gatter.



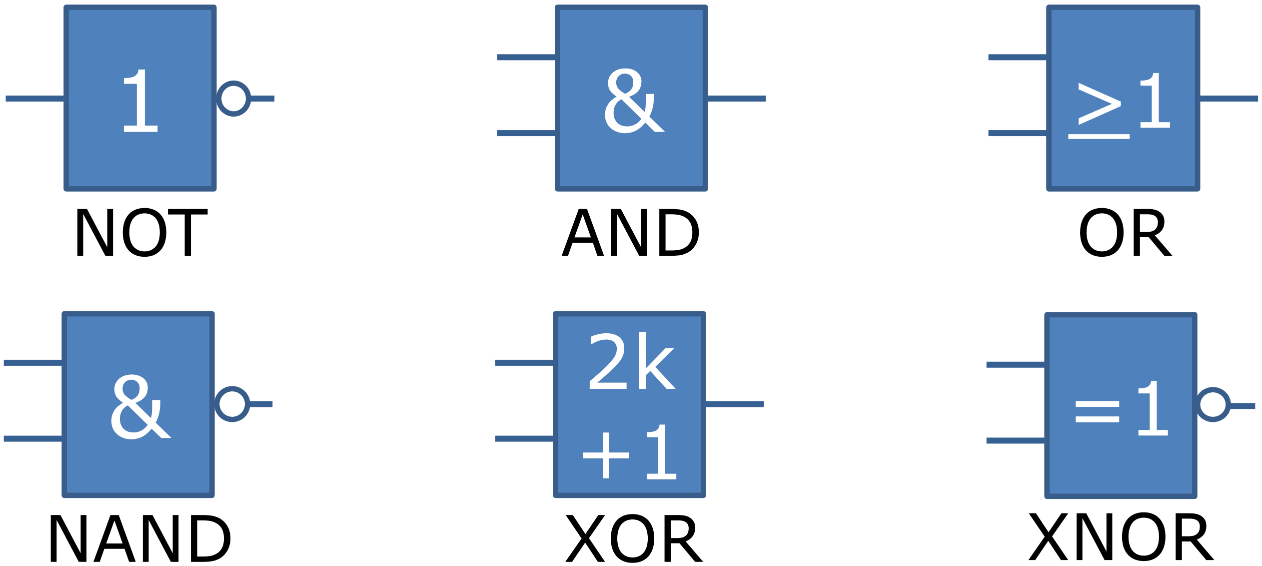
a	b	c
0	0	1
0	1	1
1	0	1
1	1	0

- Gatter (engl.: Gate) – Beispiel 3:
  - Rechts ein Beispiel zweier parallel geschalteter Transistoren.
  - Hier gilt folgende Schaltungslogik:
    - Ist V_{in}^1 ***und*** V_{in}^2 hoch → V_{out} niedrig
    - Ist V_{in}^1 hoch, V_{in}^2 niedrig → V_{out} niedrig
    - Ist V_{in}^1 niedrig, V_{in}^2 hoch → V_{out} niedrig
    - Ist V_{in}^1 ***und*** V_{in}^2 niedrig → V_{out} hoch
  - Setzen wir wieder a ← V_{in}^1, b ← V_{in}^2, und c ← V_{out}, dann lässt sich die Parallelschaltung zweier Transistoren durch folgende Schaltungslogik beschreiben:
  - Diese drei Schaltungen bilden die einfachsten Gatter. Sie heißen NOT-, NAND- bzw. NOR-Gatter.



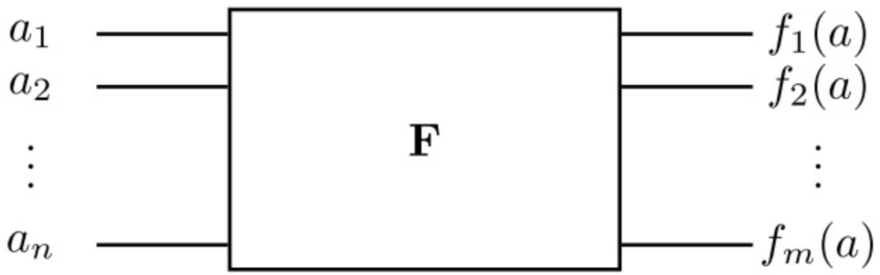
a	b	c
0	0	1
0	1	0
1	0	0
1	1	0

- Überblick Schaltgatter:



a	b	NOT(a)	OR	AND	NOR	NAND	XOR	XNOR
0	0	1	0	0	1	1	0	1
0	1	1	1	0	0	1	1	0
1	0	0	1	0	0	1	1	0
1	1	0	1	1	0	0	0	1

- Schaltnetze
  - Mit einem Schaltgatter allein können viele Operationen – wie zum Beispiel die Addition – nicht realisiert werden. Dies ist nur dann möglich, wenn mehrere Gatter zu einer größeren Einheit zusammengefasst werden.
  - In Form eines Symbols kann man sich ein Schaltnetz so vorstellen:



    Wobei:
    - a_i = Schaltzustände an den Eingängen von F (n Eingänge)
    - f_j = Schaltfunktionen
    - $f_j(a)$ = Schaltzustände an den Ausgängen von F (m Ausgänge)

Halbaddierer

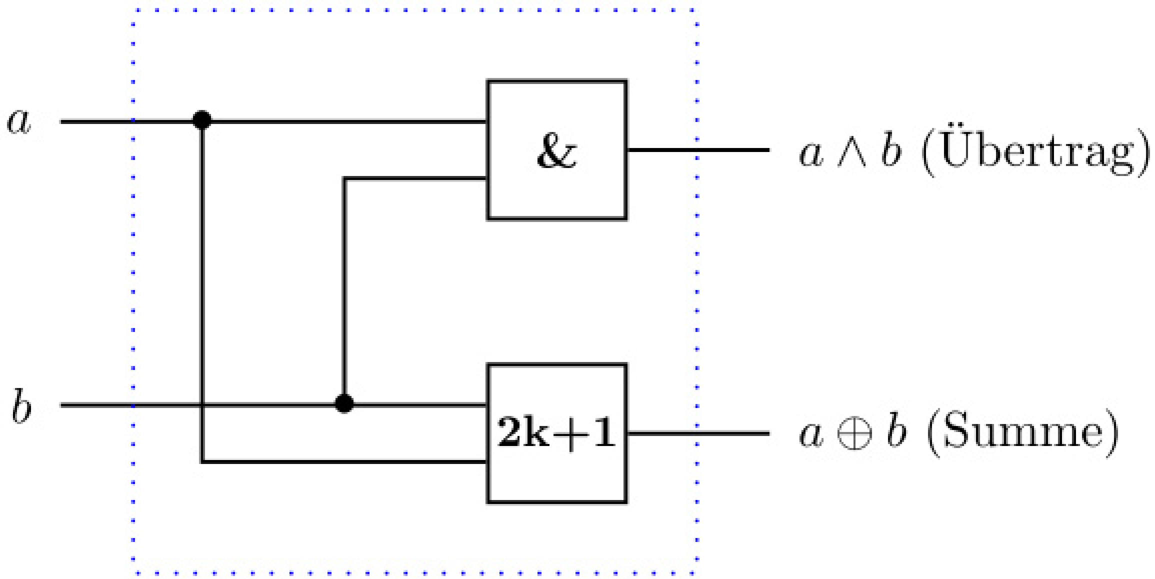
- Halbaddierer
  - Schaltungen sind zur Durchführung der Additionsoperation ein essentieller Bestandteil einer jeden CPU.
  - Zur Auffrischung noch mal die Wahrheitstabelle einer 1-Bit-Addition:

a	*b*	**Summe**	**Übertrag**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

  - Die Summe der Eingänge entspricht dabei einem XOR, der Übertrag entspricht einem AND.

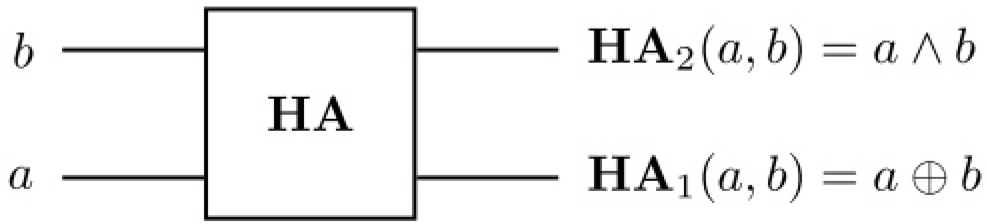
Halbaddierer

- Die Abbildung zeigt das Schaltnetz eines Halbaddierers.
- Ausreichend für die Berechnung der niederwertigen Bits zweier aus mehreren Bits bestehenden Eingabewörter.
- Für die Berechnung von Bits aus der Wortmitte ungeeignet, da der Übertrag nicht durchgeführt wird.
- Für diese Zwecke gibt es den Volladdierer.



Halbaddierer

- Die folgende Abbildung zeigt den gleichen Halbaddierer als Blockschaltung:



- Wobei die Ausgänge wie folgt belegt sind:
  - HA_1 = Summenbit
  - HA_2 = Übertragbit

Volladdierer

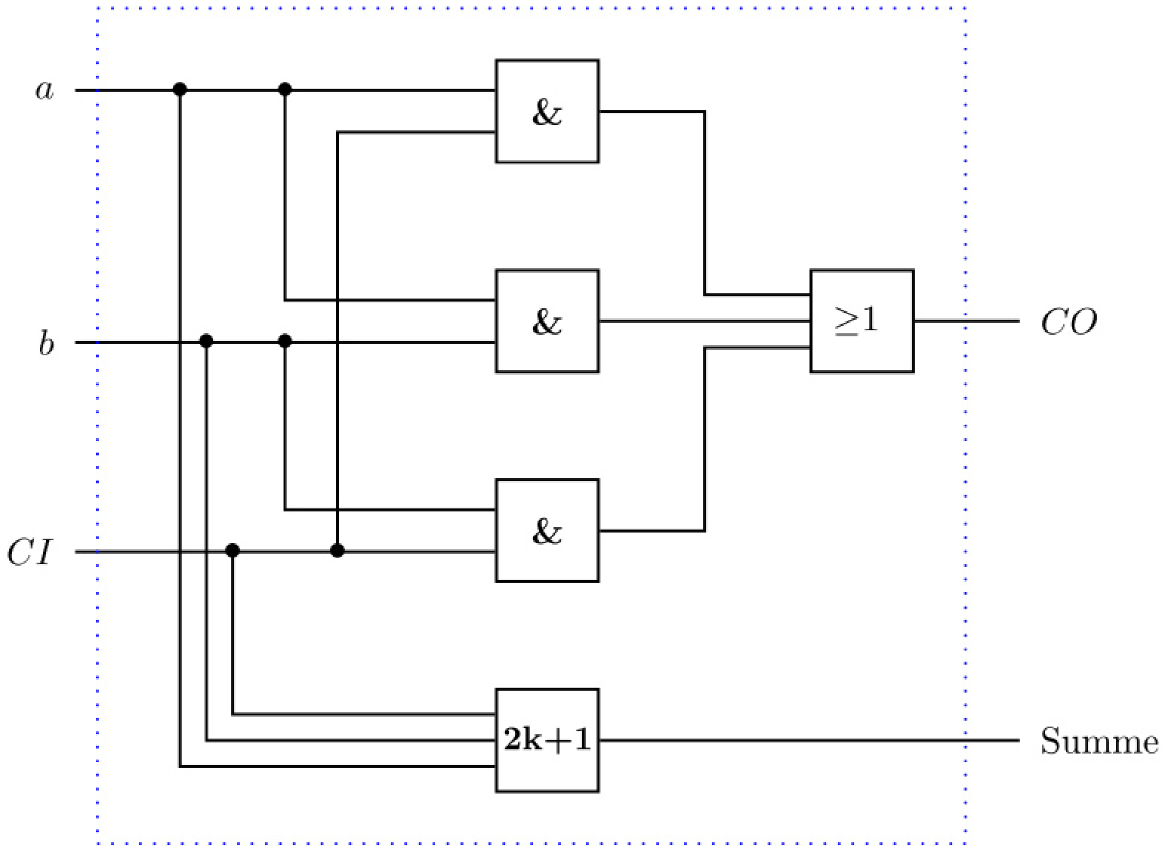
Volladdierer

- Der Volladdierer erweitert den Halbaddierer insofern, als er die Summe zweier Booleschen Variablen a und b unter der Berücksichtung eines Übertrags aus der vorgehenden Stelle (Carry In oder CI) berechnet.
- Das Resultat wird in zwei Ausgangsvariablen Summe und Carry Out dargestellt.

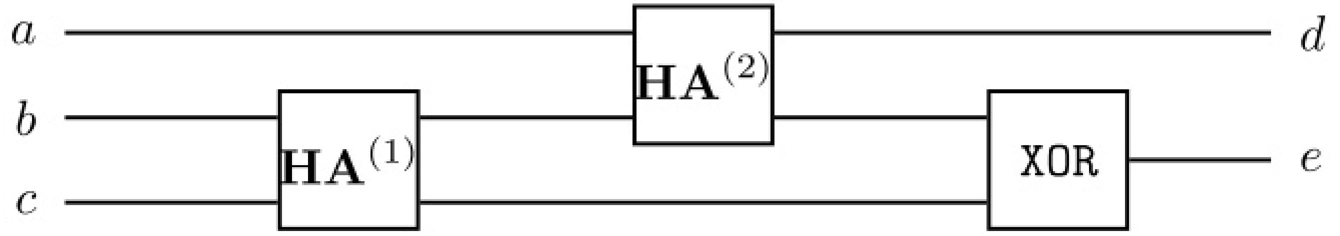
a	*b*	*Carry In*	**Summe**	**Carry Out**
0	0	0	**0**	**0**
0	0	1	**1**	**0**
0	1	0	**1**	**0**
0	1	1	**0**	**1**
1	0	0	**1**	**0**
1	0	1	**0**	**1**
1	1	0	**0**	**1**
1	1	1	**1**	**1**

- Schaltalgebra des Volladdierers:
  - **Summe** = **(a ⊕ b) ⊕ CI**
  - **CO** = (a ^ b) v (a ^ CI) v (b ^ CI)
    = **[a ^ (b ⊕ CI)] ⊕ [b ^ CI]**

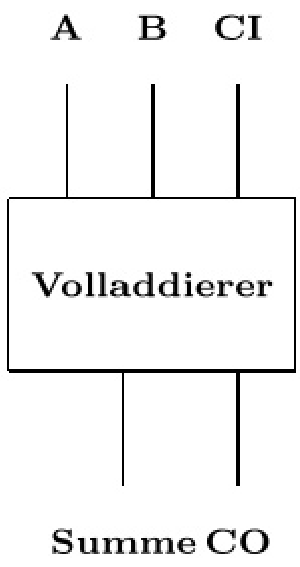
- Volladdierer
  - Die Abbildung zeigt den Schaltungsaufbau eines Volladdierers.



- Volladdierer
  - Alternativ kann der Volladdierer auch als Hintereinanderschaltung zweier Halbaddierer und eines XOR–Gatters konstruiert werden.



  - Abschließend noch das Symbol für das Schaltungsnetz eines Volladdierers:



Ripple Carry Adder

DHBW Duale Hochschule Baden-Württemberg Mannheim

- Ripple Carry Adder
  - Der Ripple Carry Adder (Addierer mit durchlaufendem Übertrag) ist im Prinzip nichts anderes als mehrere miteinander verschaltete Volladdierer.
  - Das Carry-Out Signal eines niedereren Bits wird als Carry-In des nächst höheren eingespeist.
  - Der Addierer berechnet die Summe zweier n-stelliger Dualzahlen

$$s = (CO_{n-1}\ S_{n-1}\ S_{n-2} \ldots S_1\ S_0)$$
$$a = (a_{n-1}\ a_{n-2} \ldots a_1\ A_0)$$
$$b = (b_{n-1}\ b_{n-2} \ldots b_1\ B_0)$$