Steuereinrichtungen - KOMPAKT

1. Übersicht

Steuereinrichtung
/ \
Verbindungsprogrammiert (VPS) — **Speicherprogrammierbar (SPS)**
/ \ / \
Steuerschaltungen aus Schützen, Zeitrelais usw. | Elektronische Steuerungen | „Echte“ SPS | Kleinsteuerungen

Verbindungsprogrammierte Steuerung (VPS)

Bei auf Schützen basierenden Steuerungen werden die logischen Verknüpfungen der Informationen über die Verdrahtung von Schaltern, Tastern, Sensoren und den **kontaktbehafteten** Schützen festgelegt.

In elektronische Steuerungen werden **kontaktlos** schaltenden elektronische Baugruppen eingesetzt. Auch hier ist die Informationsverknüpfung über die Verbindungen der Eingabegrößen und der elektronischen Baugruppen festgelegt. Soll die Steuerfunktion einer VPS geändert werden, muss direkt in die Schaltung eingegriffen werden:

- zusätzliche Schaltelemente installieren
- Verbindungen verändern

→ zeitaufwändig und kostspielig (Stillstandzeiten der Anlage)

Speicherprogrammierbare Steuerung (SPS)

In der Steuerungstechnik haben sich daher in weiten Bereichen Steuerungen durchgesetzt, die auf einen Mikrorechner basieren, der die Informationsverknüpfung mit Hilfe eines **Steuerprogramms** durchführt, dass in einem Halbleiterspeicher abgelegt ist. Eine Änderung der Steuerungsfunktion ist häufig allein durch eine Anpassung des Steuerprogramms realisierbar.

Die Kosten für eine SPS sind jedoch recht hoch, daher sind sie zur Verwendung bei kleinen Steuerungsaufgaben eher ungeeignet.

Eine Abhilfe stellen sogenannte **Logikmodule** (auch **Steuerrelais** oder **Kleinsteuerungen** genannt) dar. Sie bieten den Komfort einer „echten“ SPS, verfügen aber über eine deutlich reduzierte Anzahl an Ein- und Ausgängen und einen eingeschränkten Befehlssatz. Logikmodule finden bspw. häufigen Einsatz in der Gebäudetechnik.

2. Logische Verknüpfungen

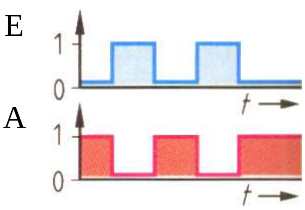
Grundverknüpfungen

NOT (NICHT, Negierung, Invertierung) $A = \overline{E}$ (auch: $A = \neg E$, $A = !E$)

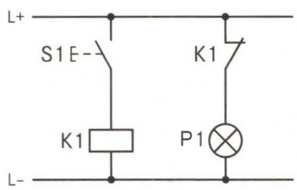
Wahrheitstabelle:

E	A
0	1
1	0

Signaldiagramm:

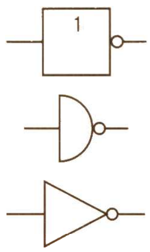


Schützschaltung:



Schaltzeichen:

DIN (neu)
DIN (alt)
ANSI



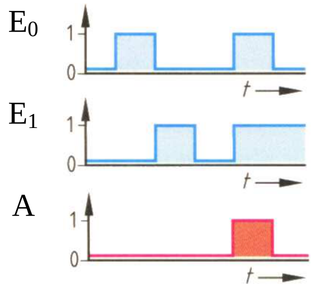
AND (UND, Konjunktion)

$A = E_0 \wedge E_1$ (auch: $A = E_0 \cdot E_1$)

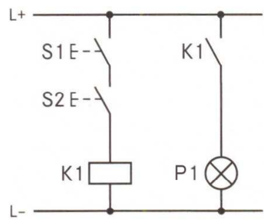
Wahrheitstabelle:

E_1	E_0	A
0	0	0
0	1	0
1	0	0
1	1	1

Signaldiagramm:



Schützschaltung:

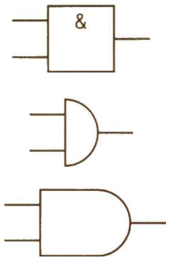


Schaltzeichen:

DIN (neu)

DIN (alt)

ANSI



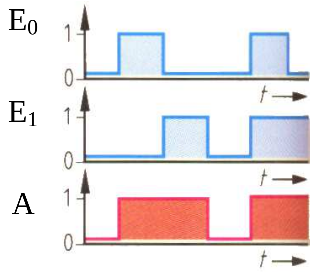
OR (ODER, Disjunktion)

$A = E_0 \vee E_1$ (auch: $A = E_0 + E_1$)

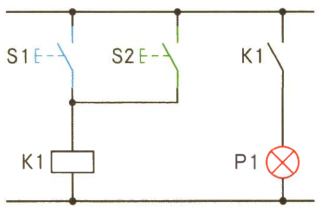
Wahrheitstabelle:

E_1	E_0	A
0	0	0
0	1	1
1	0	1
1	1	1

Signaldiagramm:



Schützschaltung:

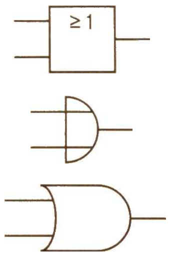


Schaltzeichen:

DIN (neu)

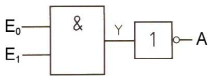
DIN (alt)

ANSI



Kombinierte Verknüpfungen

NAND (NICHT-UND)

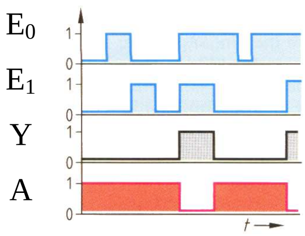


$Y = E_0 \wedge E_1$ $A = \overline{Y}$ ➔ $A = \overline{E_0 \wedge E_1}$

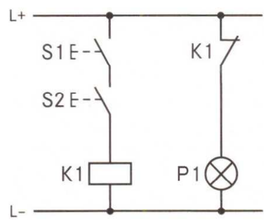
Wahrheitstabelle:

E_1	E_0	Y	A
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Signaldiagramm:



Schützschaltung:

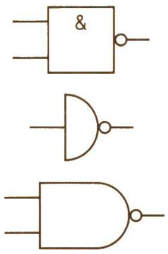


Schaltzeichen:

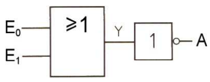
DIN (neu)

DIN (alt)

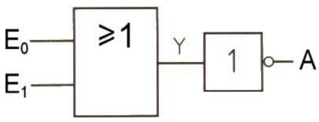
ANSI



NOR (NICHT-ODER)



$Y = E_0 \vee E_1$ $A = \overline{Y}$ ➔ $A = \overline{E_0 \vee E_1}$

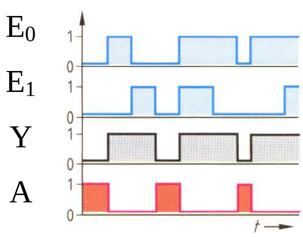


$$Y = E_0 \vee E_1 \qquad A = \overline{Y} \qquad ➔ \qquad A = \overline{E_0 \vee E_1}$$

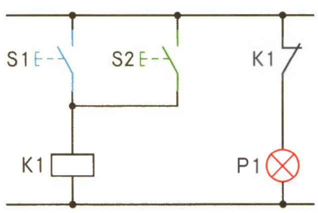
Wahrheitstabelle:

E_1	E_0	Y	A
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Signaldiagramm:



Schützschaltung:

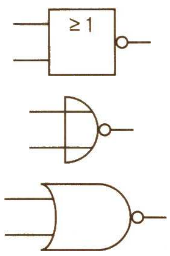


Schaltzeichen:

DIN (neu)

DIN (alt)

ANSI



Die Grundverknüpfungen und kombinierten Verknüpfungen lassen sich zur Lösung vieler Steuerungsaufgaben beliebig miteinander verbinden. Erfolgt die Verbindung rückkopplungsfrei, handelt es sich um ein **Schaltnetz**. Ist die Schaltung rückkopplungsbehaftet, spricht man von einem **Schaltwerk**.

Schaltnetze

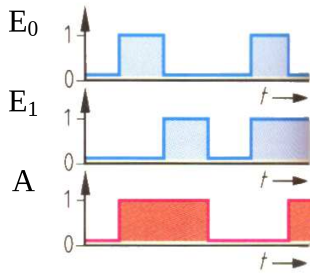
XOR (Antivalenz)

$A = E_0 \oplus E_1$ (auch: $A = E_0 \underline{\vee} E_1$)

Wahrheitstabelle:

E_1	E_0	A
0	0	0
0	1	1
1	0	1
1	1	0

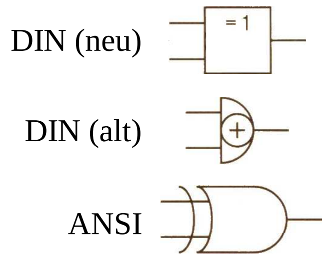
Signaldiagramm:



Schützschaltung:

?

Schaltzeichen:



Herleitung der Schützschaltung:

1. Von der Wahrheitstabelle ausgehend alle Eingangskombinationen suchen, die zu einer 1 am Ausgang führen → **1-Terme**
2. Wenn einer der gefundenen 1-Terme am Eingang anliegt, liefert der Ausgang ein 1-Signal → 1-Terme **verODERn**

Wahrheitstabelle:

E_1	E_0	A
0	0	0
0	1	1
1	0	1
1	1	0

1-Terme:

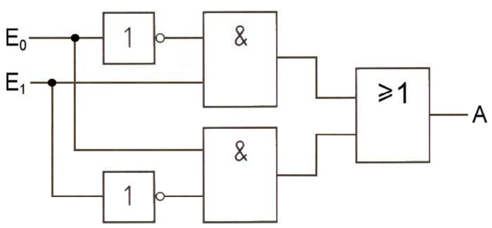
$E_0 \wedge \overline{E_1}$

$\overline{E_0} \wedge E_1$

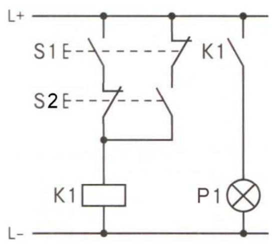
Digitale Schaltfunktion:

$$A = \left(E_0 \wedge \overline{E_1}\right) \vee \left(\overline{E_0} \wedge E_1\right)$$

Schaltnetz:



Schützschaltung:



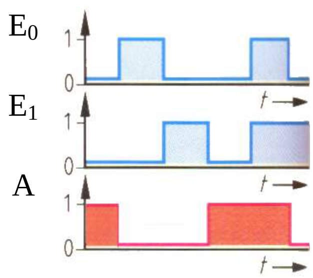
XNOR bzw. NOXOR (Äquivalenz)

$A = E_0 \equiv E_1$ (auch: $A = E_0 \overline{\underline{\vee}} E_1$)

Wahrheitstabelle:

E_1	E_0	A
0	0	1
0	1	0
1	0	0
1	1	1

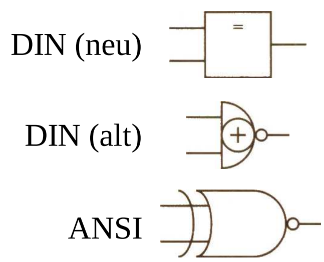
Signaldiagramm:



Schützschaltung:

?

Schaltzeichen:



Wahrheitstabelle:

E_1	E_0	A
0	0	1
0	1	0
1	0	0
1	1	1

1-Terme:

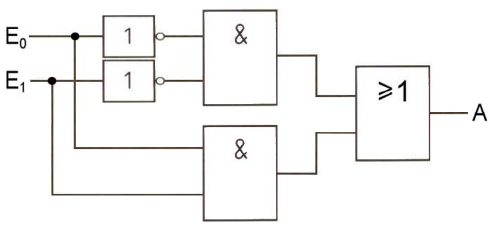
$\overline{E_0} \wedge \overline{E_1}$

$E_0 \wedge E_1$

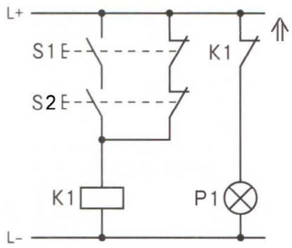
Digitale Schaltfunktion:

$$A = \left(\overline{E_0} \wedge \overline{E_1}\right) \vee \left(E_0 \wedge E_1\right)$$

Schaltnetz:



Schützschaltung:



Da bei der **Synthese** von Schaltnetzen die 1-Terme miteinander ver**ODER**t (Disjunktion) werden, spricht man beim enstehenden Gesamtterm von der **disjunktiven Normalform**.

Ein anderer Ansatz führt zur funktionsgleichen **konjuktiven Normalform**. Dieser kommt zur Anwendung, wenn die Ausgangsspalte der Wahrheitstabelle weniger Nullen als Einsen enthält und liefert dann eine einfachere und somit preiswertere Schaltung.

Herleitung der Schützschaltung:

1. In der Wahrheitstabelle alle Eingangskombinationen suchen, die zu einer 0 am Ausgang führen und jeweils die logische Bedingung aufstellen, die der Eingangskombination widerspricht → **0-Terme**
2. Wenn all diese Bedingungen erfüllt sind, muss es sich folglich um einen 1-Term handeln → 0-Terme **verUNDen**

Wahrheitstabelle:

E_1	E_0	A
0	0	1
0	1	0
1	0	0
1	1	1

0-Terme:

$\overline{E_0} \vee E_1$

$E_0 \vee \overline{E_1}$

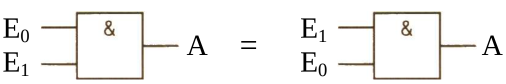
Digitale Schaltfunktion:

$$A = \left(\overline{E_0} \vee E_1\right) \wedge \left(E_0 \vee \overline{E_1}\right)$$

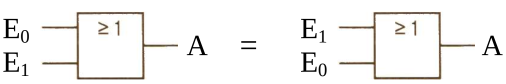
3. Rechenregeln der Digitaltechnik

(= Schaltalgebra oder boolesche Algebra)

Vertauschungsgesetz (Kommutativgesetz):

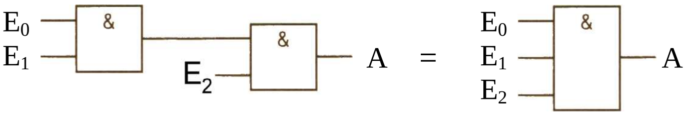


→ $\boxed{E_0 \wedge E_1 = E_1 \wedge E_0}$

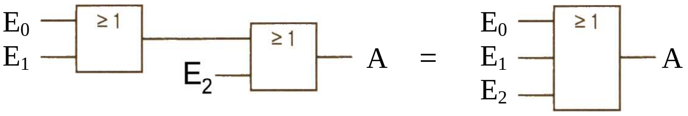


→ $\boxed{E_0 \vee E_1 = E_1 \vee E_0}$

Verbindungsgesetz (Assoziativgesetz):

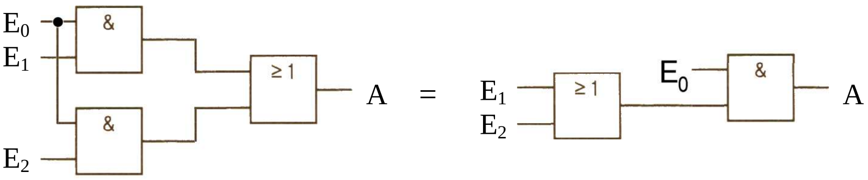


$\Rightarrow$ $\boxed{(E_0 \wedge E_1) \wedge E_2 = E_0 \wedge E_1 \wedge E_2}$



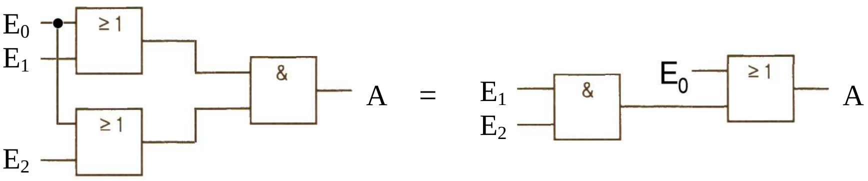
$\Rightarrow$ $\boxed{(E_0 \vee E_1) \vee E_2 = E_0 \vee E_1 \vee E_2}$

Verteilungsgesetz (Distributivgesetz):



$\Rightarrow$ $\boxed{(E_0 \wedge E_1) \vee (E_0 \wedge E_2) = E_0 \wedge (E_1 \vee E_2)}$

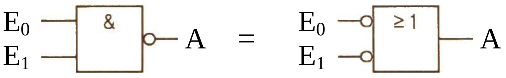
konjunktives Verteilungsgesetz



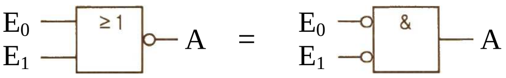
$\Rightarrow$ $\boxed{(E_0 \vee E_1) \wedge (E_0 \vee E_2) = E_0 \vee (E_1 \wedge E_2)}$

disjunktives Verteilungsgesetz

De Morgansche Gesetze:



$\Rightarrow$ $\boxed{\overline{E_0 \wedge E_1} = \overline{E_0} \vee \overline{E_1}}$



$\Rightarrow$ $\boxed{\overline{E_0 \vee E_1} = \overline{E_0} \wedge \overline{E_1}}$

Rückblickend auf die Äquivalenzschaltung des vorangegangenen Kapitels lässt sich jetzt die Funktionsgleichheit der zwei unterschiedlichen Lösungen beweisen:

$A = (\overline{E_0} \vee E_1) \wedge (E_0 \vee \overline{E_1})$	Substitution: $(\overline{E_0} \vee E_1) = Y$
$\Rightarrow A = Y \wedge (E_0 \vee \overline{E_1})$	konjunktives Verteilungsgesetz
$\Rightarrow A = (Y \wedge E_0) \vee (Y \wedge \overline{E_1})$	Resubstitution: $Y = (\overline{E_0} \vee E_1)$
$\Rightarrow A = ((\overline{E_0} \vee E_1) \wedge E_0) \vee ((\overline{E_0} \vee E_1) \wedge \overline{E_1})$	2 x konjunktives Verteilungsgesetz
$\Rightarrow A = ((\overline{E_0} \wedge E_0) \vee (E_1 \wedge E_0)) \vee ((\overline{E_0} \wedge \overline{E_1}) \vee (E_1 \wedge \overline{E_1}))$	$(\overline{E_0} \wedge E_0) = 0$, $(E_1 \wedge \overline{E_1}) = 0$
$\Rightarrow A = (0 \vee (E_1 \wedge E_0)) \vee ((\overline{E_0} \wedge \overline{E_1}) \vee 0)$	$(0 \vee E) = E$
$\Rightarrow A = (E_1 \wedge E_0) \vee (\overline{E_0} \wedge \overline{E_1})$	2 x Vertauschungsgesetz
$\Rightarrow A = (\overline{E_0} \wedge \overline{E_1}) \vee (E_0 \wedge E_1)$	

Der Rechenaufwand lässt sich erheblich reduzieren, wenn die alternativen Schreibweisen für die UND- bzw. ODER-Verknüpfung verwendet werden und die Klammern analog zur Arithmetik „ausmultipliziert“ werden.

$A = (\overline{E_0} + E_1) \cdot (E_0 + \overline{E_1})$	„ausmultiplizieren“
$\Rightarrow A = \overline{E_0} \cdot E_0 + E_1 \cdot E_0 + \overline{E_0} \cdot \overline{E_1} + E_1 \cdot \overline{E_1}$	$\overline{E_0} \cdot E_0 = 0$, $\overline{E_1} \cdot E_1 = 0$
$\Rightarrow A = E_1 \cdot E_0 + \overline{E_0} \cdot \overline{E_1}$	2 x Vertauschungsgesetz
$\Rightarrow A = \overline{E_0} \cdot \overline{E_1} + E_0 \cdot E_1$	