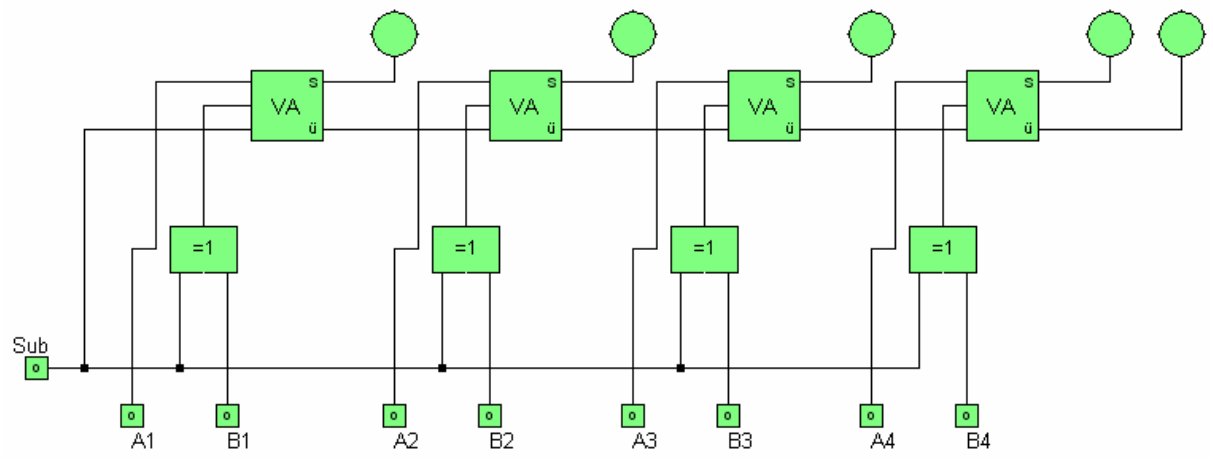
4-Bit Addierwerk (Ripple-Carry-Adder)



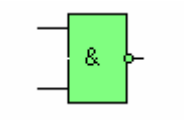
<u>Weitere wichtige Schaltfunktionen</u>

NAND (= NOT AND)

Funktionstabelle:

A	B	$A \cdot B$	$\overline{A \cdot B}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Symbol:



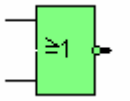
Das NAND ist das negierte AND (UND).

NOR (=NOT OR)

Funktionstabelle:

A	B	A+B	$\overline{A+B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Symbol:



Das NOR ist das negierte OR (ODER).

Vollständige Systeme

Mit AND, OR und NOR kann man jede beliebige Schaltfunktion realisieren
⇨ Vollständiges System

Mit NAND kann man ebenfalls jede beliebige Schaltfunktion realisieren
⇨ Vollständiges System

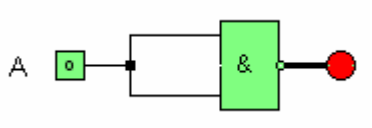
NOR ist ebenfalls ein vollständiges System

Beweis für NAND:

NICHT nur mit NAND

$$\overline{A} = \overline{A \cdot A} \text{ (Idempotenzgesetz)}$$

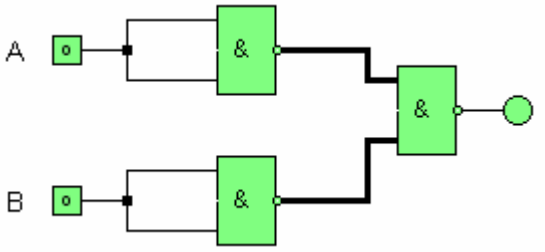
Logikplan:



ODER nur mit NAND

$$A + B = \overline{\left(\overline{A + B}\right)} \quad \text{(doppeltes Komplement)}$$
$$= \overline{\overline{A} + \overline{B}} \quad \text{(De Morgan)}$$
$$= \overline{\overline{A \cdot A} \cdot \overline{B \cdot B}}$$

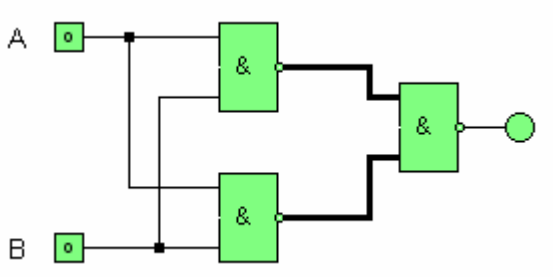
Logikplan:



UND nur mit NAND

$$A \cdot B = \left(A \cdot B\right) + \left(A \cdot B\right) \quad \text{(Idempotenzgesetz)}$$
$$= \overline{\overline{\left(A \cdot B\right) + \left(A \cdot B\right)}} \quad \text{(doppeltes Komplement)}$$
$$= \overline{\overline{A \cdot B} \cdot \overline{A \cdot B}} \quad \text{(De Morgan)}$$

Logikplan:



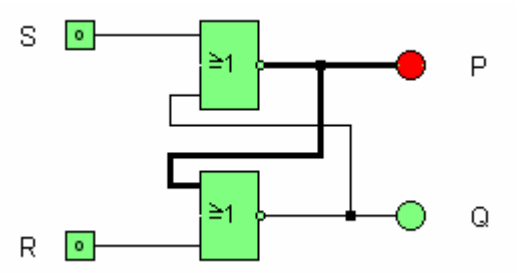
Funktion in DNF umwandeln in Funktion nur mit NAND

$$y = \left(A \cdot \left(B \cdot \overline{C}\right)\right) + \left(A \cdot C\right)$$
$$= \overline{\overline{\left(A \cdot \left(B \cdot \overline{C}\right)\right) + \left(A \cdot C\right)}}$$
$$= \overline{\overline{A} \cdot \overline{\left(B \cdot \overline{C}\right)} \cdot \overline{\left(A \cdot C\right)}}$$

RS-Flipflop

- aufgebaut aus zwei kreuzgekoppelten NOR (oder NAND) Gattern
- Setz und Rücksetzeingang

Logikplan:



Fallunterscheidung:

S = 1; R = 0 → P = 0; Q = 1

S = 0; R = 1 → P = 1; Q = 0

S = 0; R = 0; Q = 1 → P = 0; Q = 1

S = 0; R = 0; Q = 0 → P = 1; Q = 0

Q ist der gespeicherte Binärwert. Am Anschluss P kann der Wert von Q invertiert abgegriffen werden. Man bezeichnet den Anschluss deshalb auch mit $\overline{Q}$.

S = 1; R=1 → P = O; Q = 0, d.h. P = $\overline{Q}$ ist verletzt.
→ Diese Situation ist **verboten**, weil sie so nicht definiert ist!

Funktionstabelle RS-FF

Eingangsvariable		Zustandsvariable		Ausgangsvariable	
S	R	Q	P	Q^+	P^+
0	0	0	1	0	1
0	0	1	0	1	0
1	0	0	1	1	0
1	0	1	0	1	0
0	1	0	1	0	1
0	1	1	0	0	1
1	1	0	1	x	x
1	1	1	0	x	x

Q, P = aktueller Zustand
Q^+, P^+ = neuer Zustand

Die ersten beiden Zeilen zeigen die Speicherfunktion:
Liegt kein Wert an, so wird der aktuelle Wert beibehalten

Die dritte und vierte Zeile zeigen die Setzfunktion:
Liegt bei Q eine 1 an, wird sie in den Speicher geschrieben.

Die fünfte und sechste Zeile zeigen die Rücksetzfunktion:
Liegt bei R eine 1 an, wird Q auf 0 gesetzt.

Die letzten beiden Zeilen zeigen die unzulässige Einstellung, wenn an beiden Eingängen eine 1 anliegt. Für diese Einstellung werden auf Grund der Nicht-Zulässigkeit in der Funktionstabelle „don't cares“ gesetzt.

Weil bei bestimmungsgemäßen Betrieb immer P = Q gilt, wird nachfolgend nur noch Q aufgeführt.

Q ist der gespeicherte Zustand.

Funktionstabelle des RS - FF (nur noch mit Q als Zustand)

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

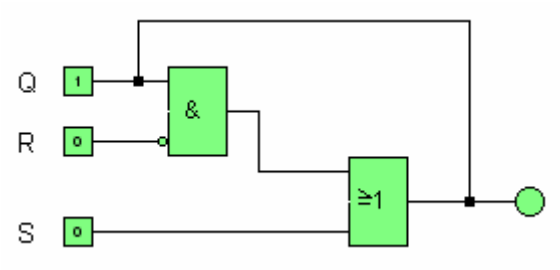
KV – Diagramm:

B \ A				
	0	1	x (1)	0
	1	1	x (1)	0

C

$Q^+ = S + \left(Q \cdot \overline{R}\right)$ (charakteristische Gleichung des RS – FF)

Logikplan:



Zeichenerklärung:

Die erstellten Logikpläne haben einige neue Schaltzeichen, die hier kurz erklärt werden.

1) Schalter: Gibt entweder eine 0 oder 1 auf die angeschlossene Leitung aus.

2) LED: Zeigt auch AUS oder AN den Zustand der Ausgangsvariablen an.