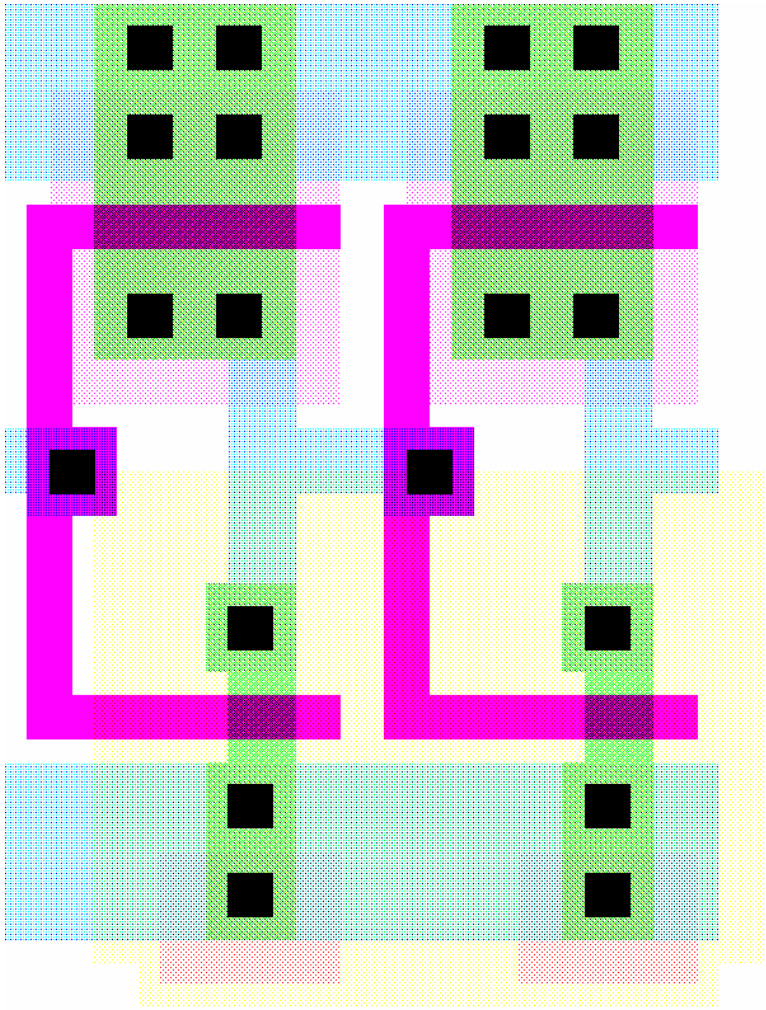
Aussagenlogik und Gatter

- Aussagenlogik, Bool'sche Algebra und Schaltalgebra
- Logische Levels
- Der Inverter, NAND und NOR
- Gemischte Gatter, XOR, XNOR



Aussagenlogik

- In der Aussagenlogik wird einer Aussage einer von zwei Wahrheitswerten **wahr** oder **falsch** zugewiesen
- Wir benutzen dafür auch die Bezeichnungen **true/false, 1/0** oder (später) **high/low**
- Für zwei Aussagen a,b führen wir **Verknüpfungen** ein, die wir über **Wahrheitstafeln** definieren:

Konjunktion
UND
AND
$y = a \cdot b = a \wedge b = ab$

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

Disjunktion
ODER
OR
$y = a + b = a \vee b = a|b$

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

Antivalenz
ENTWEDER ODER
EXCLUSIVE OR
$y = a \oplus b$

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

Äquivalenz

EXCLUSIVE NOR
$y = a \equiv b = a \leftrightarrow b$

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

Negation
NICHT
NOT
$y = \neg a = !a = \overline{a}$

a	y
0	1
1	0

- Diese Definitionen entsprechen (ziemlich) der Umgangssprache

Zwei einfache Beispiele

- Wir betrachten die Aussage 'es blitzt und donnert'.
- Dies ist offenbar die Behauptung **f = a** · **b** mit a: 'es blitzt' b: 'es donnert'
- f ist offenbar **genau dann wahr**, wenn es blitzt (a wahr) und wenn es donnert (b wahr)
- f ist offenbar **nicht wahr**
  - wenn es nicht blitzt (a falsch. Es ist egal, ob es donnert oder nicht)
  - wenn es nicht donnert (b falsch. Es ist egal, ob …)
  - wenn es weder blitzt noch donnert (a falsch und b falsch)

- Wir betrachten folgende Aussagen:
  A: 'Die Sonne scheint'
  B: 'Der Kamin ist angeschürt'
  C: 'Die Heizung ist **aus**geschaltet'
- Eine Heizungssteuerung schaltet die Heizung aus, wenn durch Sonne oder den Kamin der Raum geheizt ist.
- Es gilt: **C** $\leftrightarrow$ **A + B**, also: 'Die Heizung ist **aus**geschaltet, wenn die Sonne scheint **oder** der Kamin an ist'

- Diese Aussage ist offenbar äquivalent zu:
- **!C** $\leftrightarrow$ **!A** · **!B**, also 'Die Heizung ist **ein**geschaltet, wenn die Sonne **nicht** scheint **und** der Kamin **aus** ist'

- Die Äquivalenz der Ausdrücke **C** $\leftrightarrow$ **A + B** und **!C** $\leftrightarrow$ **!A** · **!B** kann man mit der Boole'sche Algebra beweisen.

Boole'sche Algebra

- Die mathematische Grundlage für die Schaltungstechnik bildet die **BOOLE'SCHE ALGEBRA (1854)**

- Man betrachtet darin
  - eine Menge B mit mindestens zwei Elementen
  - zwei binäre Operationen: **+** ('Addition') und • ('Multiplikation') - Vorsicht mit Analogie !
  - eine unitäre Operation: **!** (Inverses) - Nicht das 'Negative' a $\rightarrow$ -a !

- In der Boole'schen Algebra gelten folgende **Axiome**:
  1. Es gibt mindestens zwei unterschiedliche Elemente in B
  2. Abgeschlossenheit: a+b und a • b sind Elemente von B
  3. Kommutativität: a+b = b+a und a • b = b • a
  4. Assoziativität: (a+b)+c = a+(b+c) = a+b+c und (a • b) • c = a • (b • c) = a • b • c
  5. Distributivität: a+(b • c) = (a+b) • (a+c) und a • (b+c) = (a • b) + (a • c)
  6. Neutrale Elemente: $\exists$ ein Element '0' mit a+0=a und $\exists$ ein Element '1' mit a • 1 = a
  7. Inverses Element: a+!a = 1 und a • !a = 0

- N.B.: Für die reellen Zahlen z.B. gelten NICHT BEIDE Distributivgesetze:

$2 \times (3 + 4) = 2 \times 7 = 14$
$(2 \times 3) + (2 \times 4) = 6 + 8 = 14$

➽

$2 + (3 \times 4) = 2 + 12 = 14$
$(2 + 3) \times (2 + 4) = 5 \times 6 = 30$

Nicht erfüllt!

Zweiwertige Schaltalgebra

- Man kann nun zeigen, daß für
  - die Menge mit den **zwei** Elementen (1/0 oder true/false oder high/low)
  - die Verknüpfungen **ODER** ('$\vee$') für '**+**'
  - die Verknüpfungen **UND** ('$\wedge$' oder '$\cdot$' oder kein Verknüpfungszeichen) für '$\bullet$'
  - die Operation **NICHT** ('!') für '**!**'

  alle Axiome erfüllt sind.
- Dies ist die einfachst mögliche Boole'sche Algebra (wg. der zwingenden Existenz von 0 und 1!)

- Überprüfung der Axiome:

  - 0 ist das Neutrale Element für ODER ➳
  - 1 ist das Neutrale Element für UND ➳
  - !a ist das Inverse Element für ODER ➳
  - !a ist das Inverse Element für UND ➳

a	$0 \vee a$	$1 \vee a$	$1 \wedge a$	$0 \wedge a$
0	0	1	0	0
1	1	1	1	0

a	!a	$a \vee$!a	$a \wedge$!a
0	1	1	0
1	0	1	0

Assoziativgesetze der Schaltalgebra

Für ODER:

a	b	c	a∨b	(a∨b)∨c	b∨c	a∨(b∨c)
0	0	0	0	**0**	0	**0**
0	0	1	0	**1**	1	**1**
0	1	0	1	**1**	1	**1**
0	1	1	1	**1**	1	**1**
1	0	0	1	**1**	0	**1**
1	0	1	1	**1**	1	**1**
1	1	0	1	**1**	1	**1**
1	1	1	1	**1**	1	**1**

(a ∨ b) ∨ c ≡ a ∨ (b ∨ c)

Für UND:

a	b	c	a∧b	(a∧b)∧c	b∧c	a∧(b∧c)
0	0	0	0	**0**	0	**0**
0	0	1	0	**0**	0	**0**
0	1	0	0	**0**	0	**0**
0	1	1	0	**0**	1	**0**
1	0	0	0	**0**	0	**0**
1	0	1	0	**0**	0	**0**
1	1	0	1	**0**	0	**0**
1	1	1	1	**1**	1	**1**

(a ∧ b) ∧ c ≡ a ∧ (b ∧ c)

- Konsequenzen aus der Assoziativität:
  - Die Reihenfolge der Auswertung einer Summe / eines Produktes ist egal
  - Die Klammern können daher weggelassen werden

Distributivgesetze der Schaltalgebra

Für ODER:

a	b	c	b∧c	a∨(b∧c)	a∨b	a∨c	(a∨b)∧(a∨c)
0	0	0	0	**0**	0	0	**0**
0	0	1	0	**0**	0	1	**0**
0	1	0	0	**0**	1	0	**0**
0	1	1	1	**1**	1	1	**1**
1	0	0	0	**1**	1	1	**1**
1	0	1	0	**1**	1	1	**1**
1	1	0	0	**1**	1	1	**1**
1	1	1	1	**1**	1	1	**1**

$a \vee (b \wedge c) \equiv (a \vee b) \wedge (a \vee c)$

Für UND:

b∨c	a∧(b∨c)	a∧b	a∧c	(a∧b)∨(a∧c)
0	**0**	0	0	**0**
1	**0**	0	0	**0**
1	**0**	0	0	**0**
1	**0**	0	0	**0**
0	**0**	0	0	**0**
1	**1**	0	1	**1**
1	**1**	1	0	**1**
1	**1**	1	1	**1**

$a \wedge (b \vee c) \equiv (a \wedge b) \vee (a \wedge c)$

- Die Priorität der Operatoren ist wichtig. Normalerweise geht Multiplikation vor Addition, also **UND vor ODER**
- Daher ist $a \cdot b + c = (ab) + c$, während $a + b \cdot c = a+(bc) \neq (a+b)\, c$
- Die Distributivgesetzte lauten also z.B. auch **$(a+b)c = ac + bc$** und **$ab+c = (a+c)(b+c)$** (ungewohnt!)

Weitere Rechenregeln

- Aus den Axiomen folgen mehrere **wichtige** Rechenregeln (Beweis jeweils z.B. mit Wahrheitstafeln):

$a \cdot a$	=	a	'Idempotenzgesetze'
$a + a$	=	a	
$a \cdot 0$	=	0	
$a + 1$	=	1	
$a + a \cdot b$	=	a	'Absorptionsgesetze'
$a \cdot (a+b)$	=	a	
$a + !a \cdot b$	=	$a + b$	
$a \cdot (!a + b)$	=	$a \cdot b$	
$!(!a)$	=	a	'doppelte Negation'
$!(a + b)$	=	$!a \cdot !b$	'De Morgan'sche Gesetze'
$!(a \cdot b)$	=	$!a + !b$	

a	b	a+b	!(a+b)	!a	!b	!a · !b
0	0	0	**1**	1	1	**1**
0	1	1	**0**	1	0	**0**
1	0	1	**0**	0	1	**0**
1	1	1	**0**	0	0	**0**

- Verallgemeinerungen von De Morgan:

$$\overline{X_1 + X_2 + \ldots + X_n} = \overline{X_1} \cdot \overline{X_2} \cdot \ldots \cdot \overline{X_n}$$

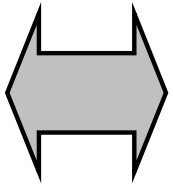
$$\overline{X_1 \cdot X_2 \cdot \ldots \cdot X_n} = \overline{X_1} + \overline{X_2} + \ldots + \overline{X_n}$$

Wichtig !!

Dualität

- Zwei **Verknüpfungen** sind zueinander **dual,** wenn Sie durch Vertauschung aller Einsen und Nullen in den Wahrheitstafeln auseinander hervorgehen.
- Beispiel: UND und ODER sind zueinander dual, XOR und XNOR, NOT und NOT

a	b	a · b
0	0	0
0	1	0
1	0	0
1	1	1



a	b	a + b
1	1	1
1	0	1
0	1	1
0	0	0

- Zwei **Ausdrücke** sind zueinander **dual**, wenn sie ineinander übergehen wenn
  - alle Einsen und Nullen vertauscht werden und umgekehrt
  - alle Verknüpfungen durch ihre dualen Partner ersetzt werden
- Das **Äquivalenzprinzip** besagt:
  'Sind zwei Ausdrücke äquivalent, so sind auch die dualen Ausdrücke einander äquivalent'
  (Daraus folgt z.B. sofort die Gleichwertigkeit der beiden Regel-Paare auf der vorherigen Seite)
- Nützliche Regel:
  'Man erhält das Inverse eines Ausdrucks, indem man im dualen Ausdruck jede Variable einzeln invertiert.'
  (0-1 Vertauschen in der Tabelle entspricht links der Negation der Variablen, rechts der Negation des Ausdrucks.) Daraus folgen die De Morgan'schen Regeln sofort.
- Merke: Beim ‚Durchbrechen' eines Inversions-Strichs werden die Verknüpfungen an der Bruchstelle durch ihr Duales Pendant ersetzt.

XOR und XNOR

- Die Äquivalenz (XNOR) und Antivalenz (XOR) kommen in der Schaltalgebra nicht direkt vor.
- Die zugehörigen Ausdrücke können aus den Wahrheitstafeln abgelesen werden:

XOR

a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

$$a \oplus b = \bar{a} \cdot b + a \cdot \bar{b} = (a + b) \cdot (\bar{a} + \bar{b})$$

ausmultiplizieren

XNOR

a	b	$a \equiv b$
0	0	1
0	1	0
1	0	0
1	1	1

$$a \equiv b = \bar{a} \cdot \bar{b} + a \cdot b = (\bar{a} + b) \cdot (a + \bar{b})$$

Test : $a \oplus b = \overline{a \equiv b} = \overline{\bar{a} \cdot \bar{b} + a \cdot b} = (\overline{\bar{a} \cdot \bar{b}}) \cdot (\overline{a \cdot b}) = (\bar{\bar{a}} + \bar{\bar{b}}) \cdot (\bar{a} + \bar{b}) = (a + b) \cdot (\bar{a} + \bar{b})$

Schaltungstechnische Realisierung der Logischen Werte

Praktische Realisierung von 0/1

- In der schaltungstechnischen Realisierung werden den zwei Elementen meist Spannungen zugeordnet:

- z.B.: - die Spannung +5V repräsentiert eine 1,
  - die Spannung 0V ('Masse') repräsentiert eine 0

- Viele andere Vereinbarungen sind möglich, sie werden in den verschiedenen '**Logikfamilien**' benutzt.

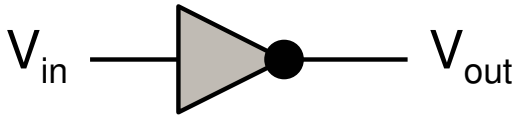
	TTL, CMOS bei 5V	CMOS bei 3.3V	ECL	Differentiell	Stromlogik
'1'	+5V	+3.3V	-0.9V	a>b	Strom
'0'	GND	GND	-1.7V	b>a	kein Strom

- ACHTUNG: Diese Vereinbarungen sind idealisiert und reichen für die Praxis nicht aus, denn dort kommen auch Spannungen von z.B. 0.1V vor. Welches 'Level' ist das dann?

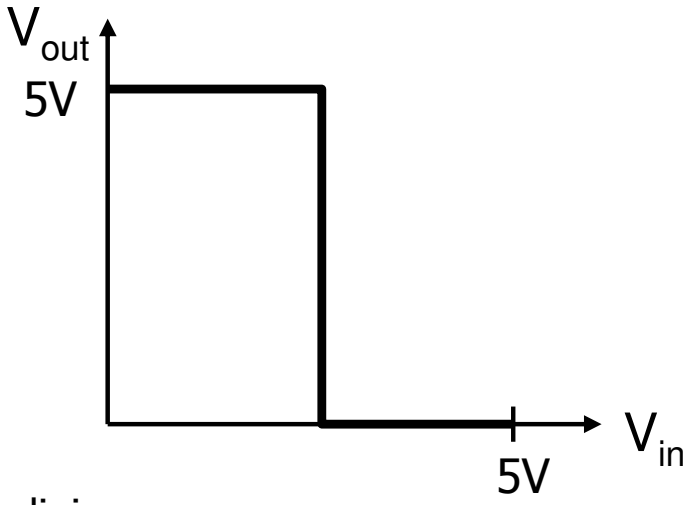
- Man definiert daher Spannungs**bereiche** mit oberen und unteren **Grenzen ...**

Der ideale Inverter

- Schaltsymbol:

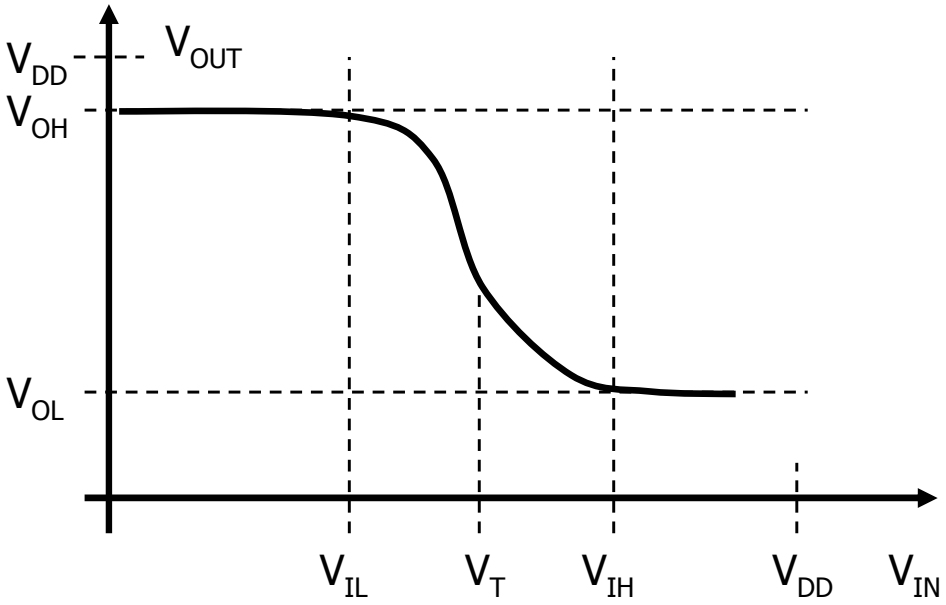


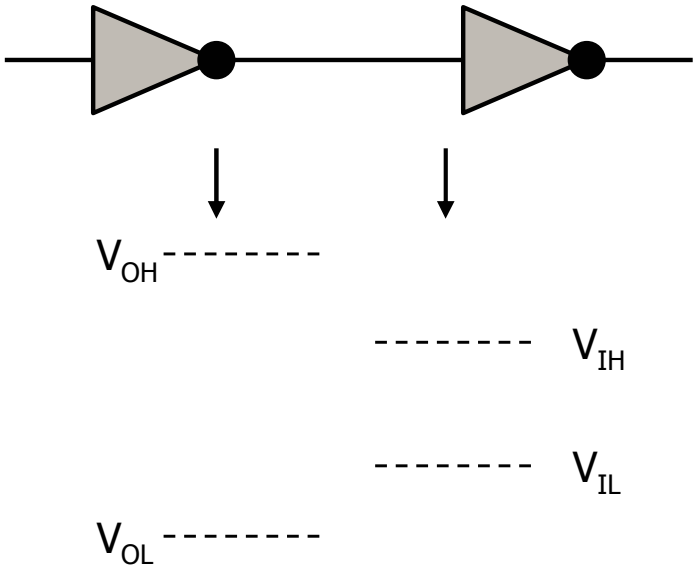
- Ein 'idealer' CMOS Inverter verwandelt 5V $\Rightarrow$ GND und GND $\Rightarrow$ 5V
- Eine reale Schaltung antwortet (natürlich) auch auf Zwischenspannungen am Eingang.
- Im Idealfall sieht die **Übertragungskennlinie** daher etwa so aus:



- Merkmale dieser idealisierten Kennlinie:
  - Der Ausgang erzeugt immer volle Logikpegel
  - Der Umschaltpunkt (die 'Schwelle') liegt in der Mitte zwischen High und Low Pegeln.
  - Der Übergang ist sehr steil (hohe Verstärkung)

Der reale Inverter: Kennlinie, Signalpegel, Störabstand



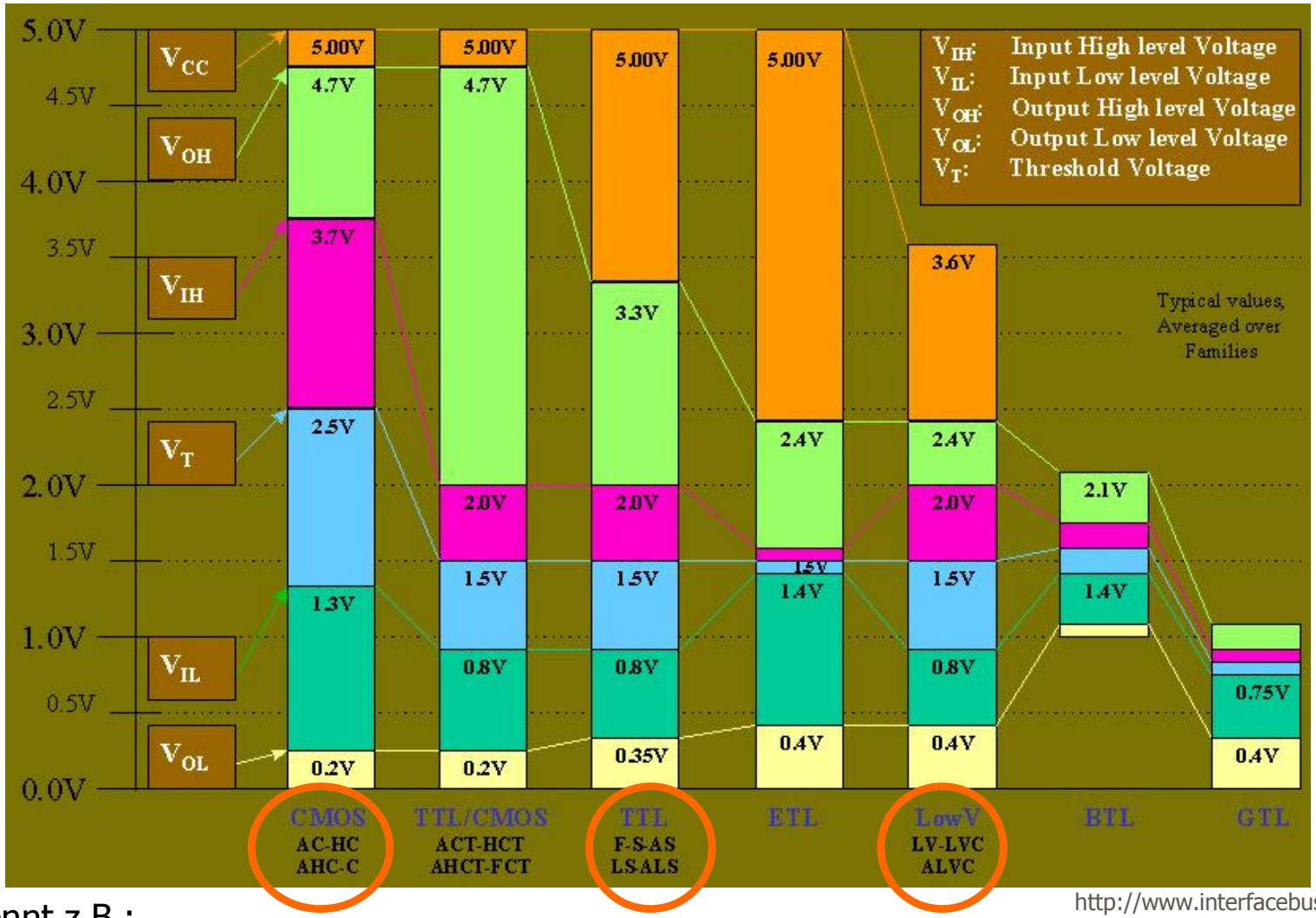


- Man garantiert **am Ausgang** eine Spannung $\mathbf{V_{out} \geq V_{OH}}$ **für eine 1** und $\mathbf{V_{out} \leq V_{OL}}$ **für eine 0**
- Man garantiert, daß **am Eingang** eine Spannung $\mathbf{V_{in} < V_{IL}}$ **als 0** und $\mathbf{V_{in} > V_{IH}}$ **als 1** erkannt wird.
- Der Ausgang eines Inverters (Gatters) muß den Eingang des nächsten sicher ansteuern können. Daher muß

$$\mathbf{V_{OH} \gg V_{IH} \quad und \quad V_{OL} \ll V_{IL}}$$

- Die Differenzen $V_{IL} - V_{OL} > 0$ bzw. $V_{OH} - V_{IH} > 0$ nennt man **Störabstände**. Sie sollen möglichst groß sein, damit Störungen auf den Signalen die nächste Stufe nicht versehentlich schaltet.
- Ursachen für solche Störungen ('Noise') sind z.B.
  - kapazitives Übersprechen von Nachbarleitungen
  - Durch Zuleitungswiderstände führen kurzzeitige hohe Stromflüsse zu Spannungsabfällen und dadurch zu Störungen in den Versorgungsspannung oder der Masse $\Rightarrow$ Leitungen müssen niederohmig sein!

Beispiele für Signalpegel



http://www.interfacebus.com/voltage_threshold.html

Man erkennt z.B.:

- Ein 5V CMOS Bauelement könnte ein low-V-CMOS Element korrekt ansteuern ($V_{OH,CMOS} > V_{IH,LV}$, $V_{OL,CMOS} < V_{IL,LV}$)
  (in der Praxis führt $V_{OH,CMOS} > VDD_{LV}$ zu Problemen!)
- 5V CMOS kann TTL ansteuern, TTL **kann aber nicht** 5V CMOS ansteuern (high-Pegel!)

Aufbau und Eigenschaften des Inverter und einfacher Gatter

(eine erste Übersicht, mehr Details folgen später)

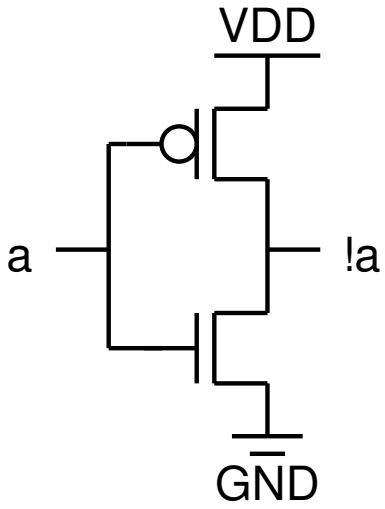
Aufbau des CMOS Inverters

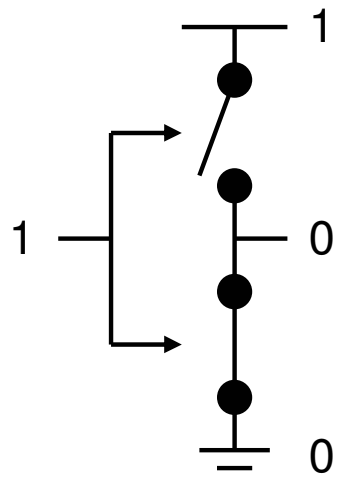
- Ein CMOS Inverter besteht aus 2 unterschiedlichen **Feldeffekttransistoren** (mehr dazu später...)

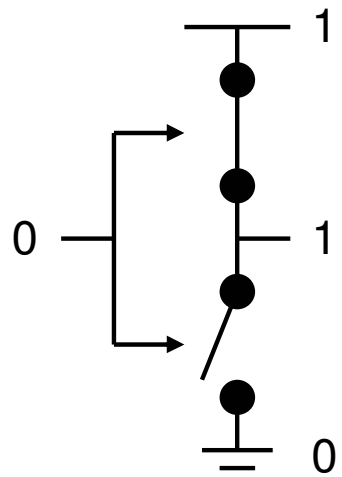
Ein **NMOS** Transistoren wirkt (**sehr** vereinfacht) wie ein **Schalter**, der **geschlossen** ist, wenn das **'Gate' auf positiver Spannung** liegt.

Ein **PMOS** Transistoren wirkt wie ein Schalter, der geschlossen ist, wenn das Gate auf **negativer** Spannung (relativ zur 'Source') liegt.

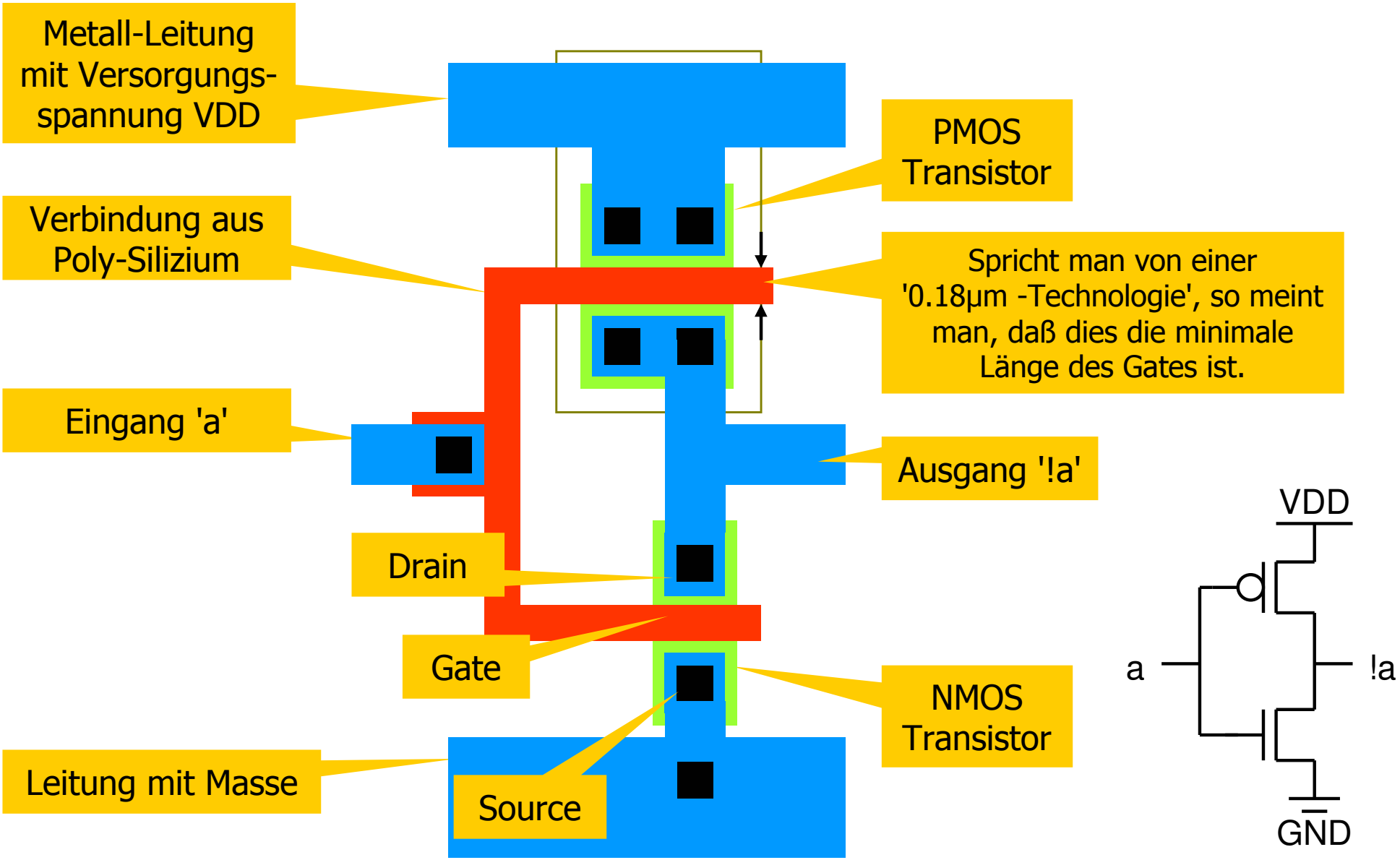
- Etwas genauer: Die **Gate-Source Spannung** muß eine bestimmte '**Schwellenspannung**' (0.5V...1V) über- (NMOS) bzw. unterschreiten (PMOS), damit der **Drain-Source-Kanal** leitet.
- Der CMOS-Inverter sieht so aus:



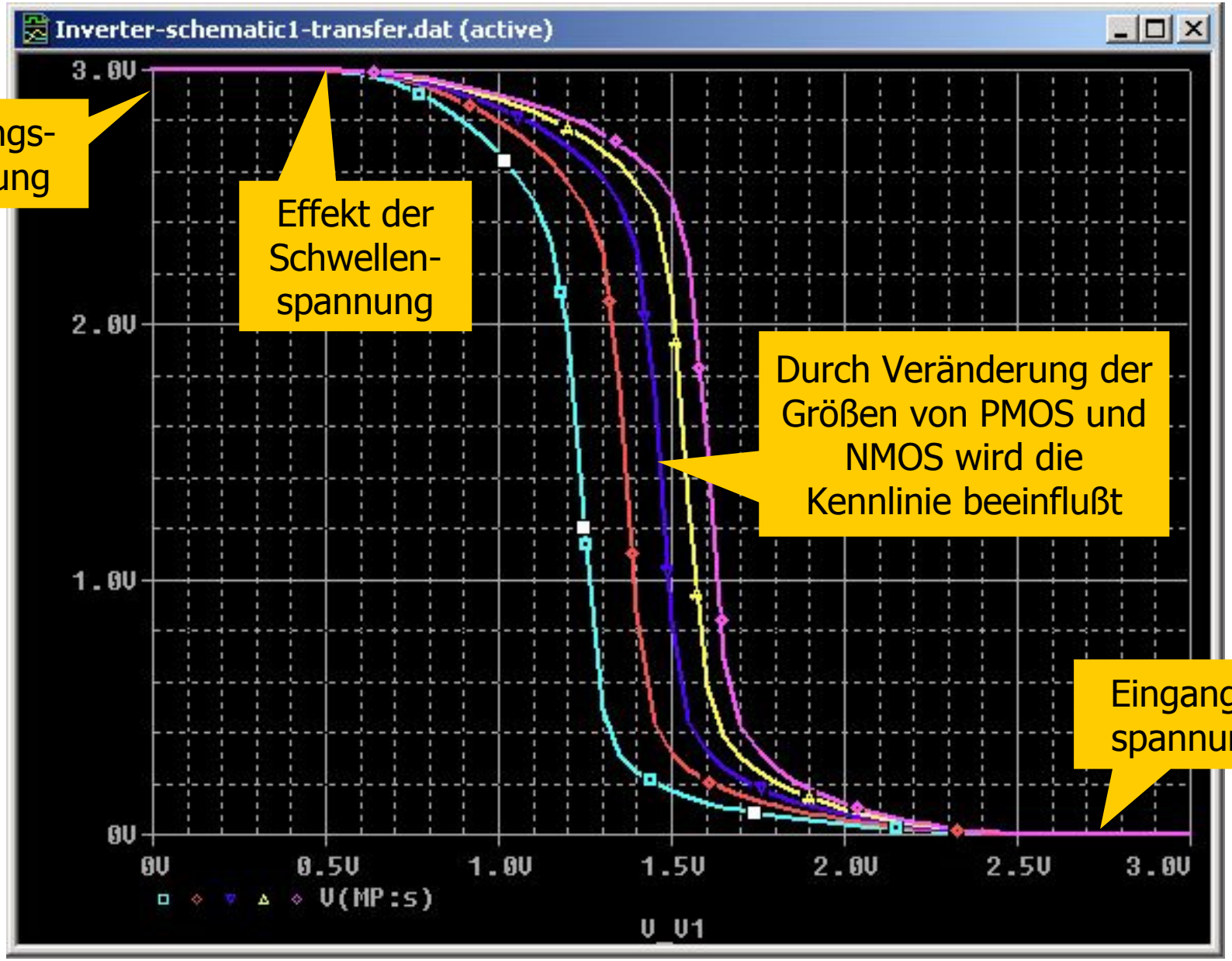




Vorgriff: Layout Inverter

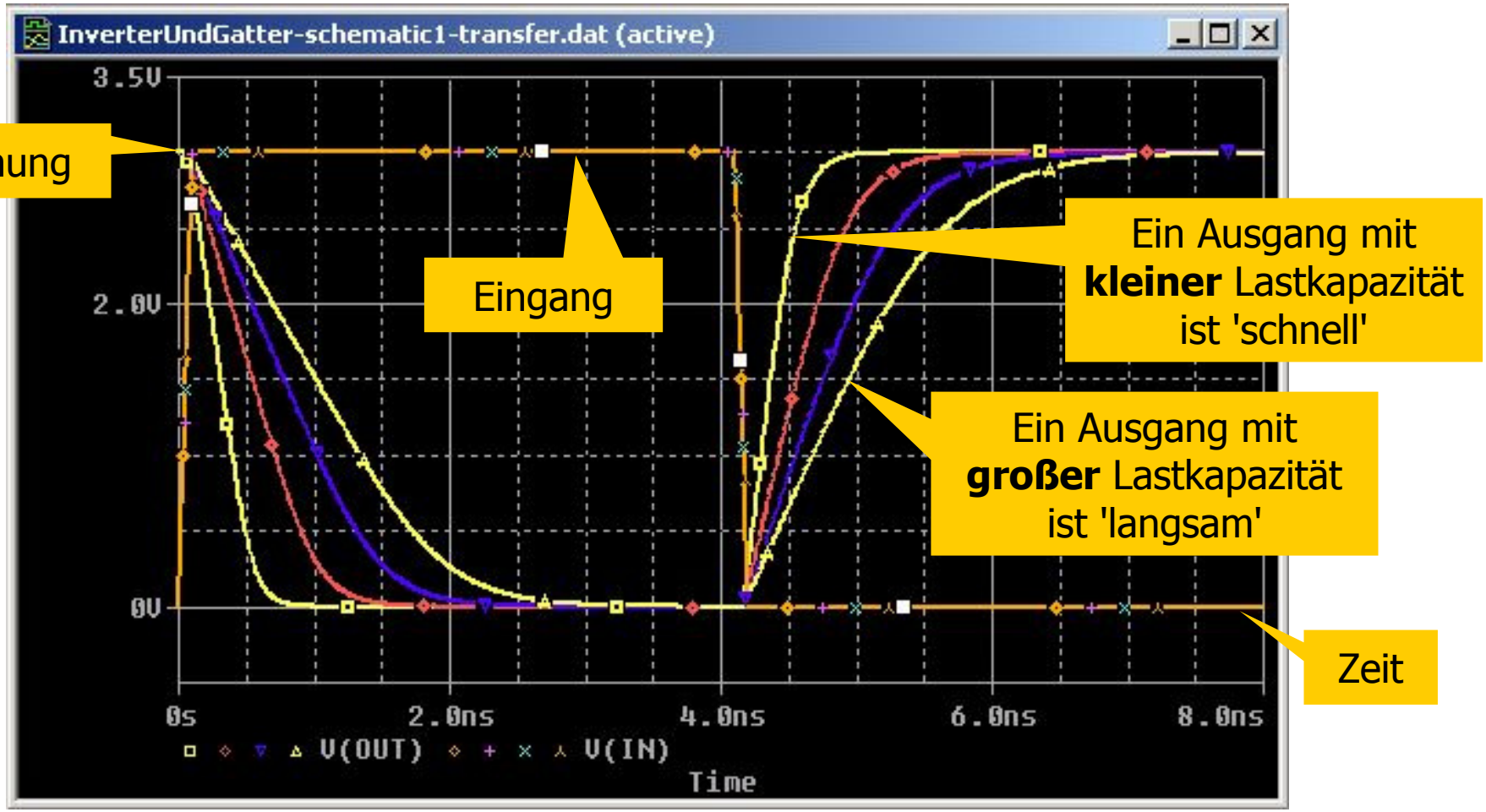


Übertragungskennlinie eines realen CMOS Inverters



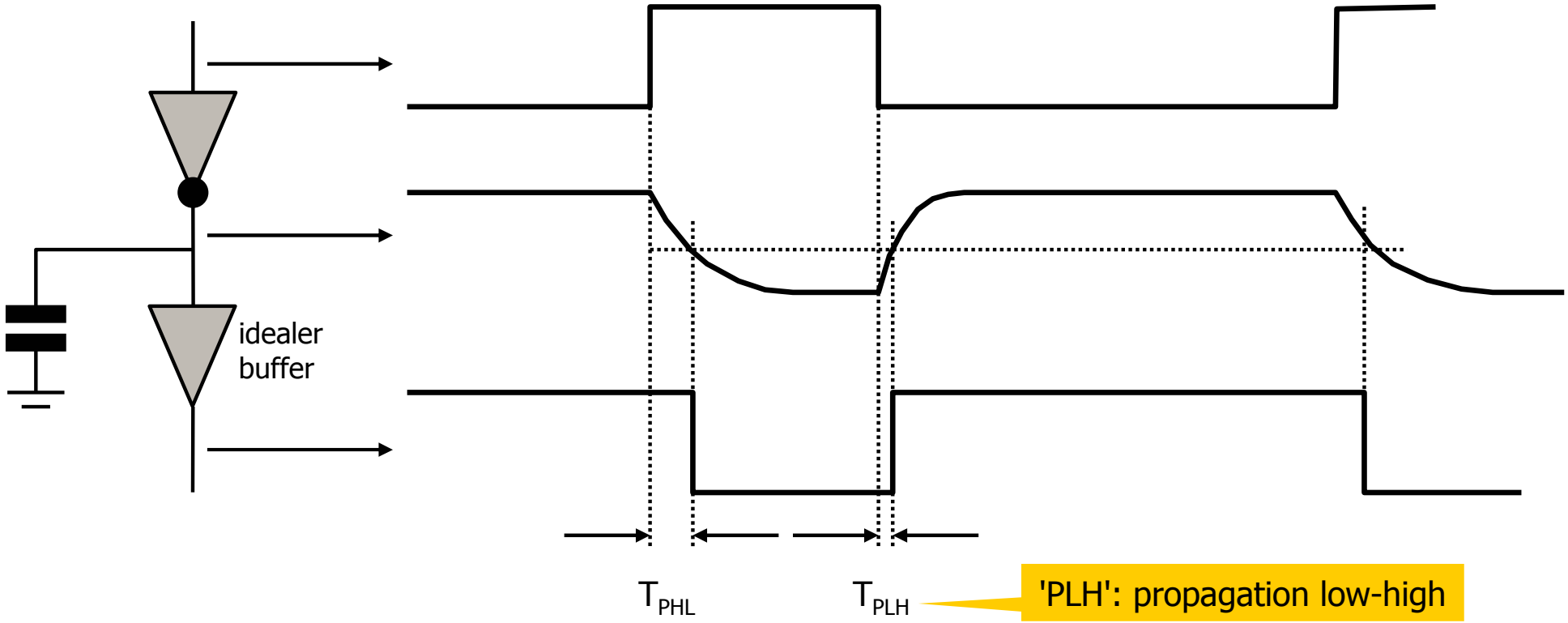
Zeitverhalten des realen CMOS Inverters

- Am Ausgang eines Inverters (Gatters) befindet sich immer eine **Lastkapazität** (Leitungen, Eingänge nachfolgender Gatter)
- Das **Umladen** der Lastkapazität (VDD ⇒ GND und GND ⇒ VDD) **erfordert Zeit**
- **Je größer die Kapazität, desto langsamer der Umladevorgang**



Verzögerung

- Durch den verlangsamten Signalanstieg am Ausgang schalten nachfolgende Stufen etwas verzögert:



- Merke:
  - Durch Lastkapazitäten kommt es bei jedem Gatter zu Verzögerungen
  - Die Verzögerung steigt mit steigender kapazitiver Belastung des Ausgangs ('Fan-Out')
  - Fallende (T_{PHL}) und steigende (T_{PLH}) Flanke haben i.a. NICHT gleiche Verzögerung (sie werden durch unterschiedliche Transistoren gemacht!)
  - Die Verzögerungen und deren Lastabhängigkeiten müssen bei der Simulation genau berücksichtigt werden, denn sie limitieren letztlich die Geschwindigkeit einer Schaltung

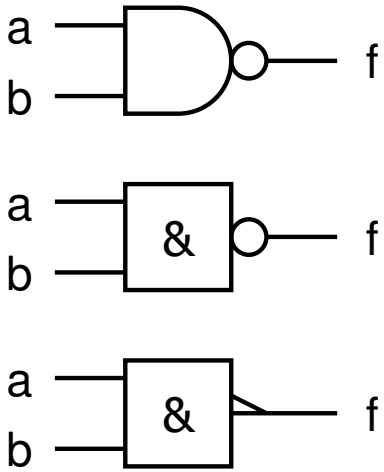
CMOS NAND Gatter mit 2 Eingängen

- NAND Gatter mit 2 Eingängen ('NAND2'):

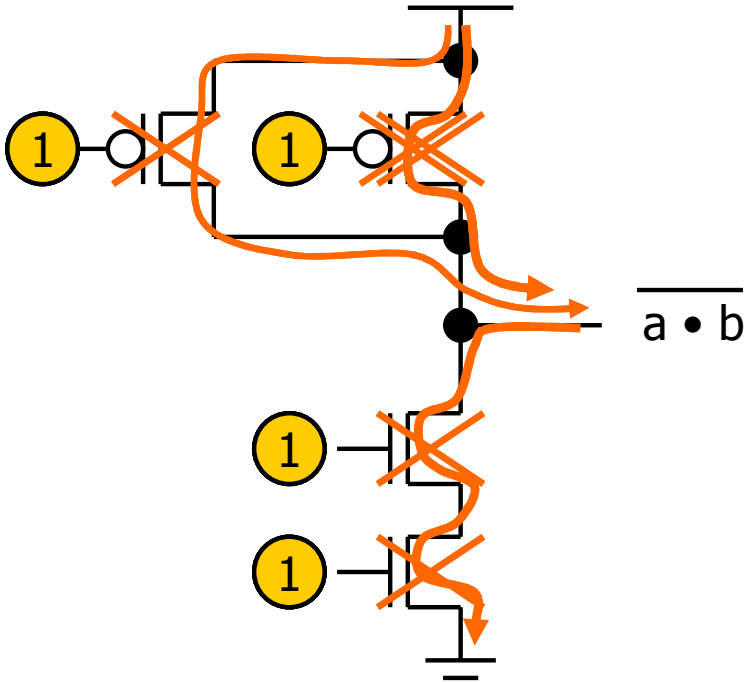
NAND2

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

Schaltsymbole



Realisierung



- Bemerkungen:
  - Die Schaltung (der NMOS Transistoren!) ist NICHT (ganz) symmetrisch in (a,b)
  - Die **NMOS** Transistoren machen die **fallende Flanke**, die **PMOS** Transistoren die **steigende Flanke**
  - Die NMOS-Transistoren sind in Reihe geschaltet. Daher erhöht sich der Widerstand. Die Entladung des Ausgangs nach Masse ist langsamer. T_{PHL} wird daher höher...

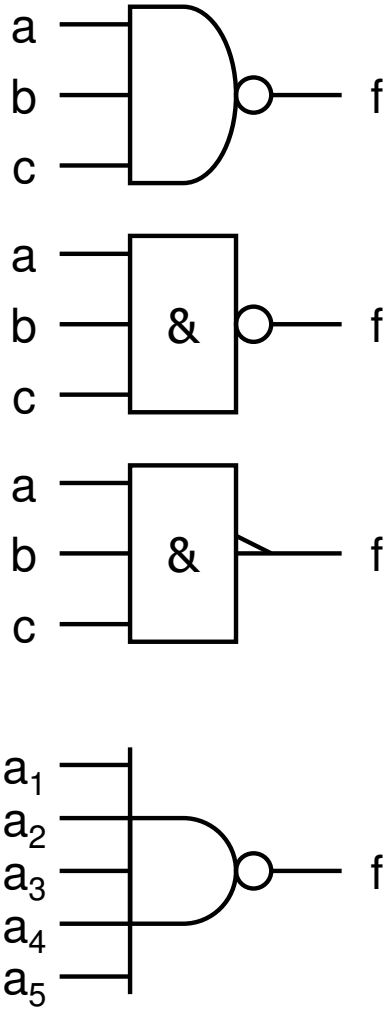
CMOS NAND Gatter mit >2 Eingängen

- z.B. NAND3:

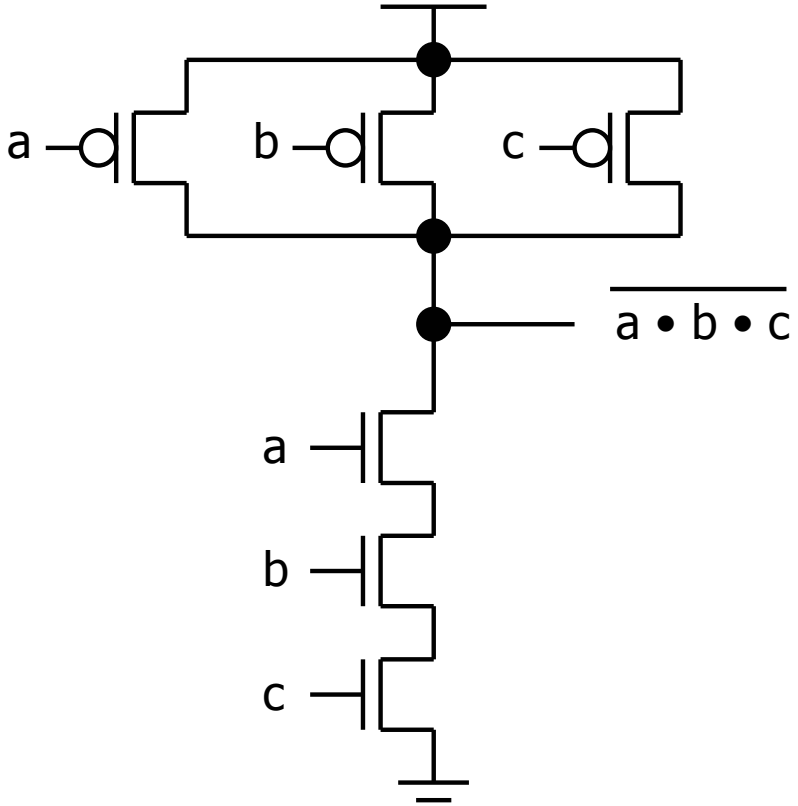
NAND3

a	b	c	$\overline{a \cdot b \cdot c}$
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	0

Schaltsymbole



Realisierung



- Viele Eingänge:
- Merke:
  Mehr als ~ 5 Eingänge sind ungebräuchlich !

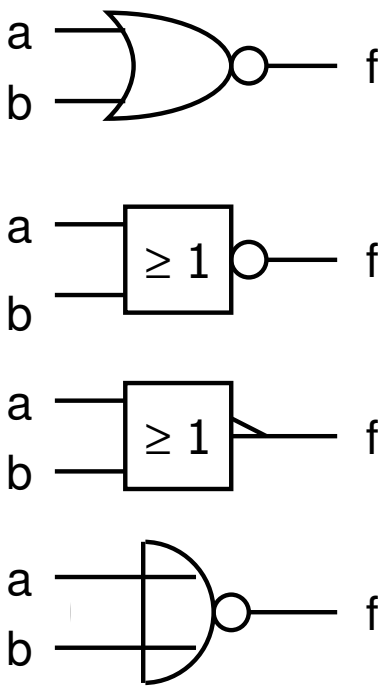
CMOS NOR Gatter mit 2 Eingängen

- NOR Gatter mit 2 Eingängen ('NOR2'):

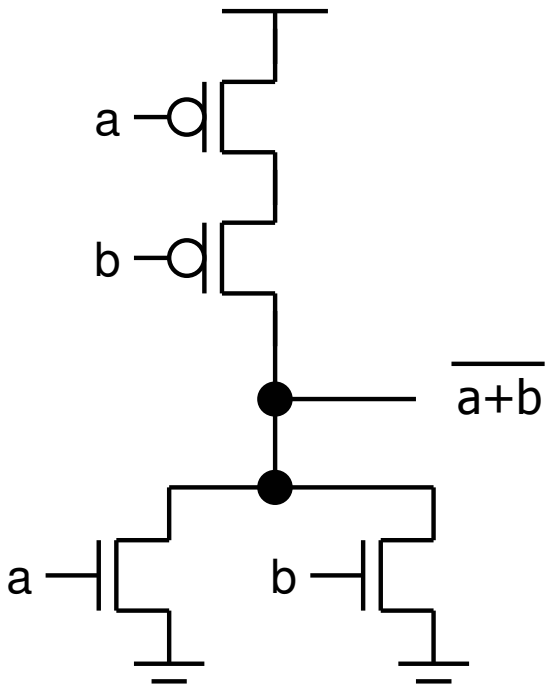
NOR2

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

Schaltsymbole

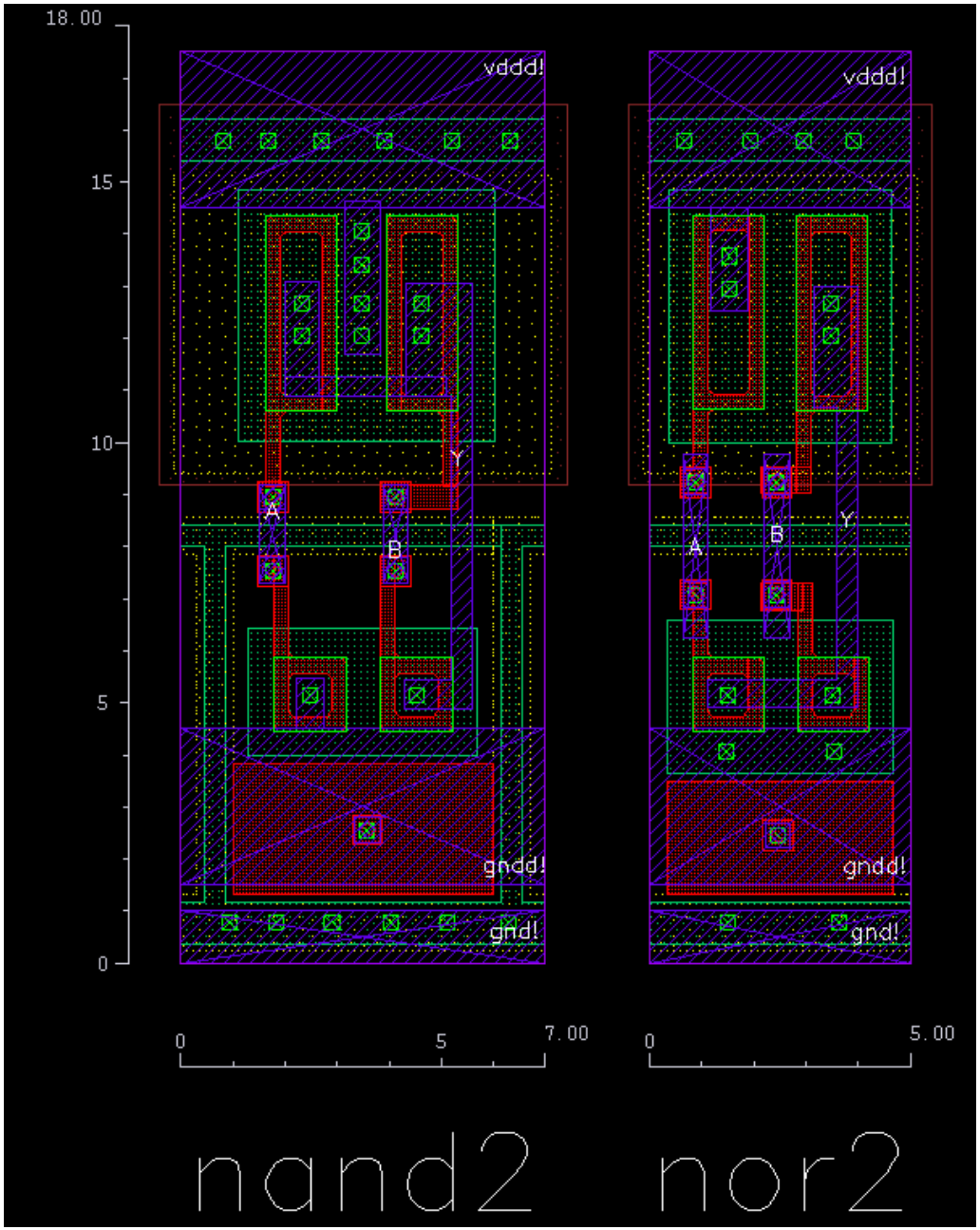


Realisierung



- Hier sind die PMOS-Transistoren in Reihe geschaltet.
  Die Aufladung des Ausgangs nach VDD wird dadurch langsamer. T_{PLH} wird daher höher...

Vorgriff: Layouts von NAND und NOR

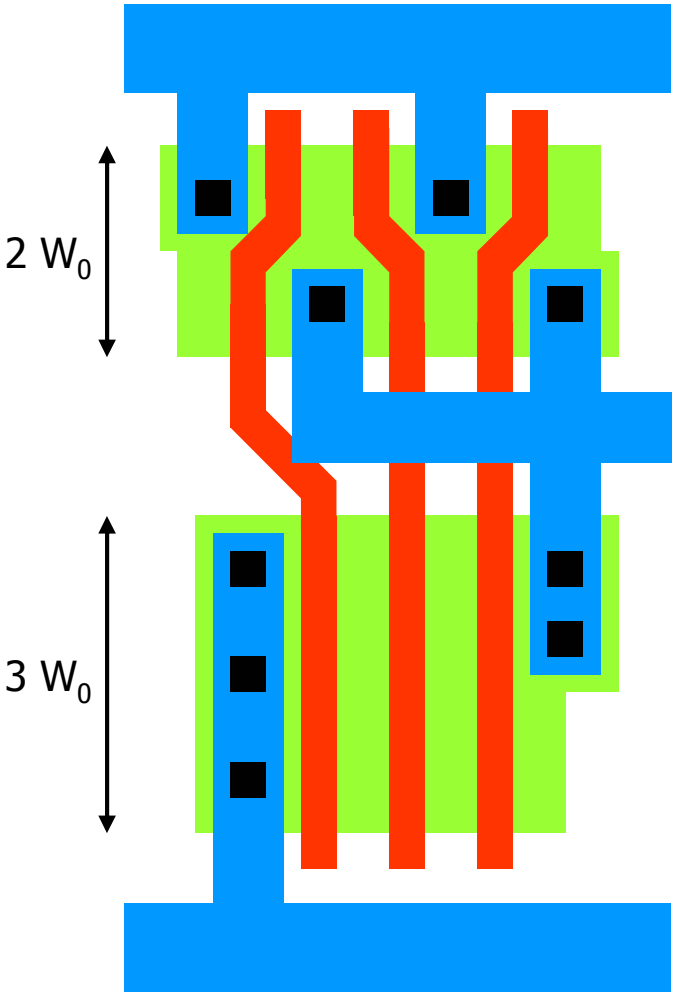


Bei diesem speziellen Layout wurden 'ringförmige' Transistoren benutzt (um Leckströme bei Bestrahlung der Chips zu vermeiden)

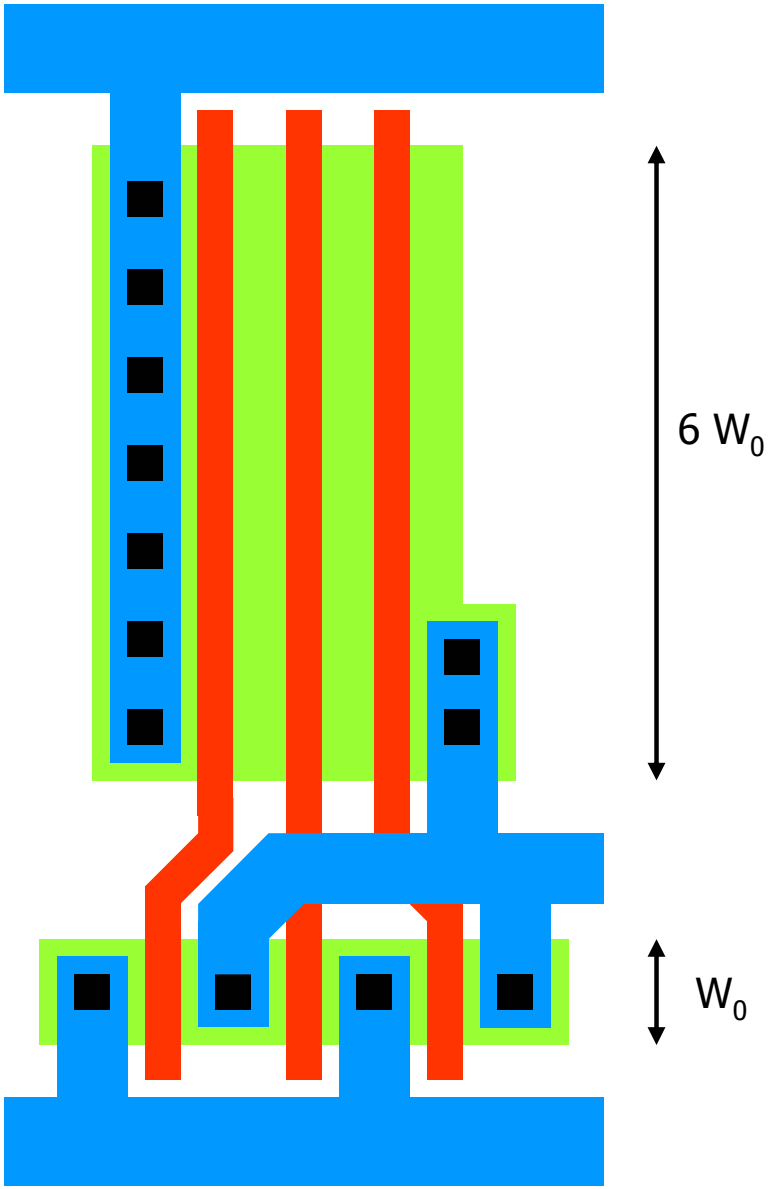
In diesem Fall ist das NOR2 Gatter wesentlich kleiner als das NAND2 Gatter !

Vorgriff auf VLSI Design: nor3 vs. nand3

- Gleiche 'Pulldown'-Stärke ($K_N=2K_P$):
- (Wannen der PMOS u. Substratkontakte nicht gezeigt!)



nand3



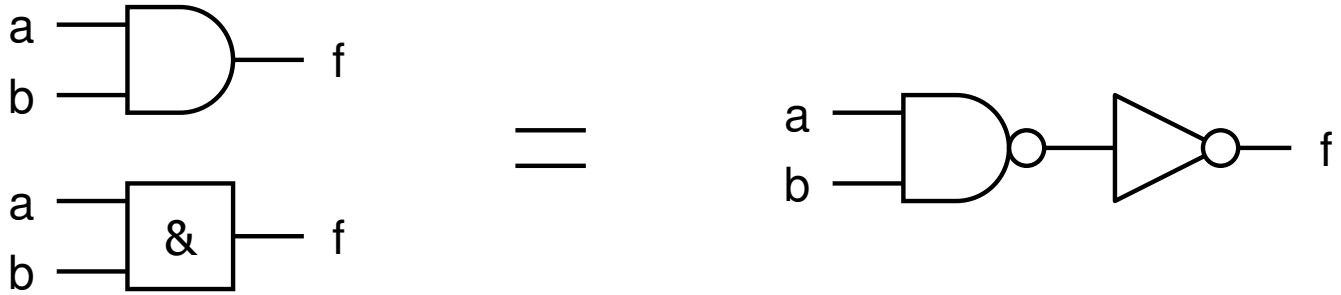
nor3

AND und OR Gatter

- Durch die Eigenschaften der NMOS- und PMOS Transistoren werden immer **invertierende** Funktionen erzeugt. Daher:

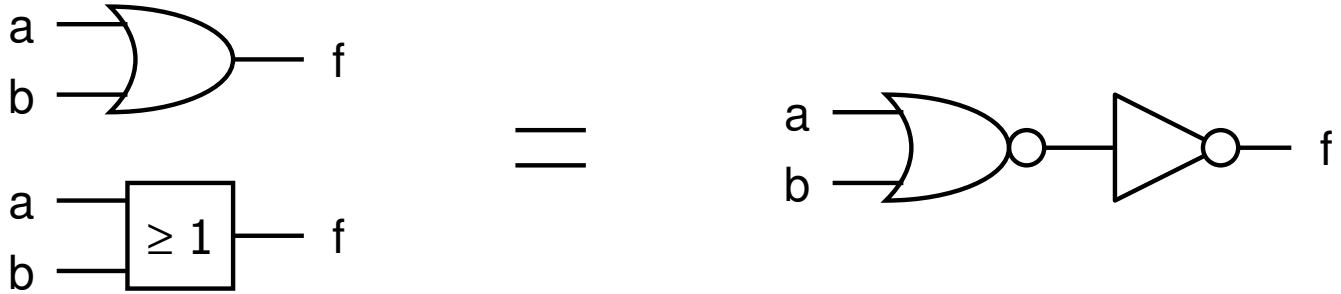
AND und OR Gatter werden aus NAND und NOR Gattern erzeugt !

- AND:



=

- OR:

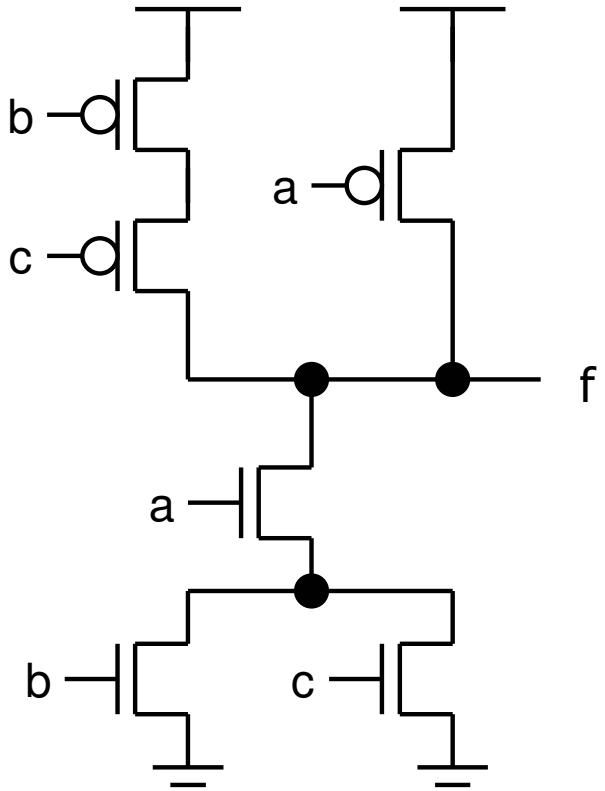


=

- Merke: AND und OR Gatter sind (in CMOS) langsamer als NAND und NOR und werden daher wenig benutzt!

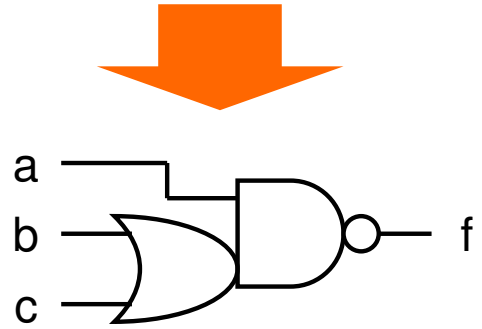
Gemischte Gatter

- Betrachte die Schaltung:



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

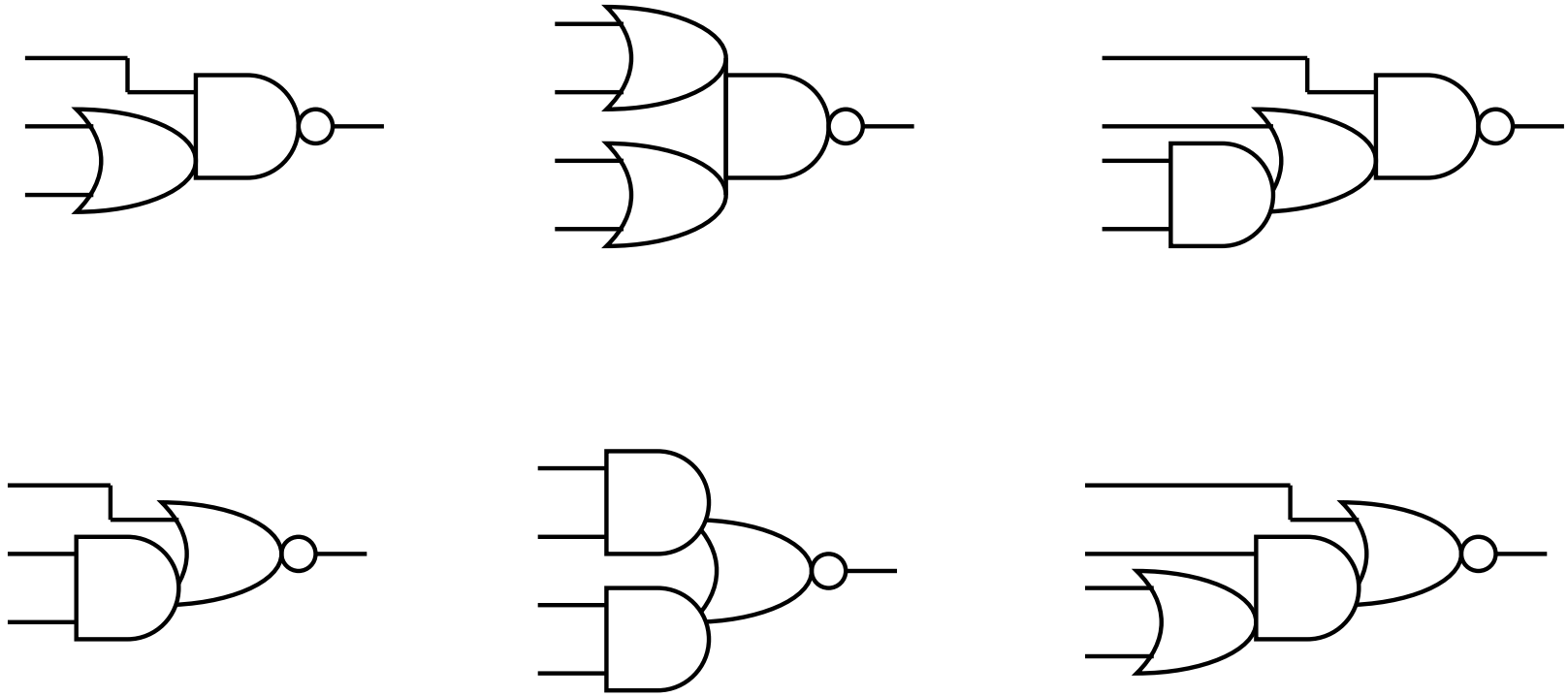
$$f = \overline{a\bar{b}c + ab\bar{c} + abc}$$
$$= \overline{a(\bar{b}c + b\bar{c} + bc)} \quad (\textit{Distributivgesetz})$$
$$= \overline{a(\bar{b}c + b(\bar{c} + c))} \quad (\textit{Distributivgesetz})$$
$$= \overline{a(\bar{b}c + b)} \quad (\textit{Neutrales Element})$$
$$= \overline{a(c + b)}$$
$$f = \overline{a(b + c)} \quad (\textit{Kommutativgesetz})$$



- Achtung: Für keine Eingangskombination darf ein Kurzschluß zwischen VDD und GND entstehen!
- Beachte: Die Verschaltung der PMOS Transistoren ist **dual** zu den NMOS Transistoren!
  Hier a(b+c) $\Rightarrow$ a+(bc)

Weitere gemischte Gatter

- Entsprechend kann man in CMOS direkt implementieren:



etc.

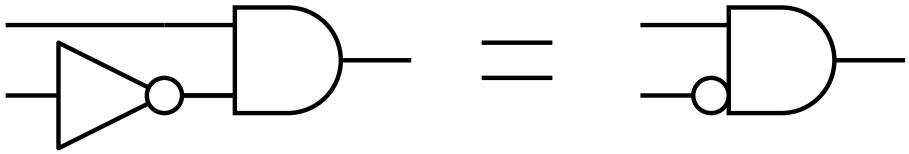
- Diese direkt implementierten Gatter benötigen bei N Eingängen 2N Transistoren (N x NMOS, N x PMOS)

'Bullets' und Active Low Signale

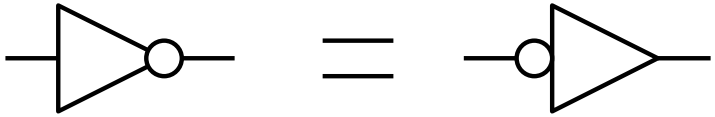
- In den verschiedenen Symbolen kam am Ausgang oft ein Kreis vor.

Der Kreis ('Bullet') symbolisiert allgemein eine Inversion des Signals

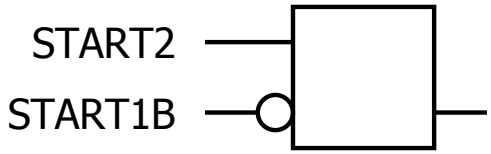
- Man benutzt Bullets auch am Eingang von Gattern.



- Den Inverter zeichnet man manchmal auch so:

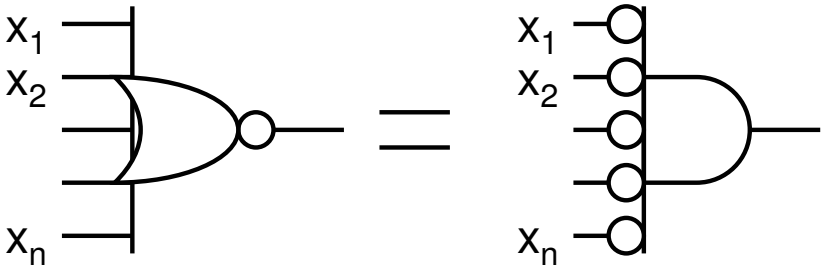


- Signale, die ihren 'aktiven' Zustand bei 0 haben, bezeichnet man als **Active-Low Signale** (im Gegensatz zu Active-High Signalen). Im Schaltsymbol tragen Active-Low-Signale daher eine Bullet.
- Man sollte Active-Low Signale z.B. durch ein angehängtes 'b' oder ‚_N' im Signalnamen kennzeichnen!
- Beispiel: - Taste 1 erzeugt einen 0-Pegel wenn sie gedrückt ist ('START1B')
  - Taste 2 erzeugt einen 1-Pegel ('START2')
  - Ein Schaltsymbol sieht dann so aus:

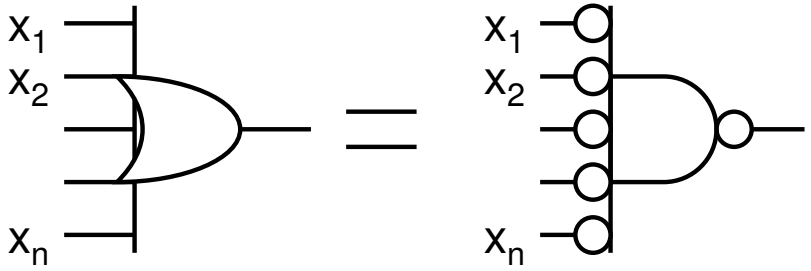


Nochmal De Morgan

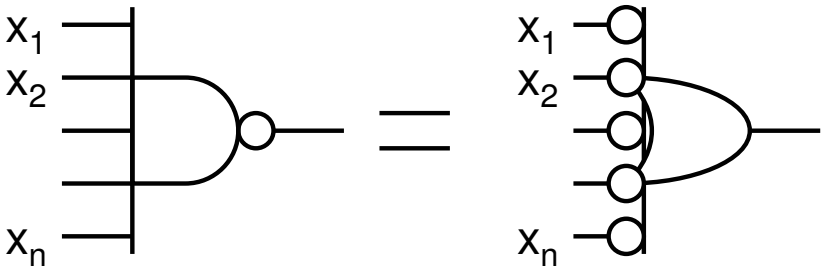
$$\overline{x_1 + x_2 + \ldots + x_n} = \overline{x_1} \cdot \overline{x_2} \cdot \ldots \cdot \overline{x_n}$$



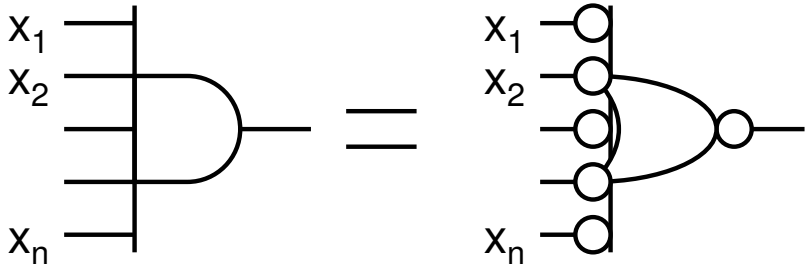
$$x_1 + x_2 + \ldots + x_n = \overline{\overline{x_1} \cdot \overline{x_2} \cdot \ldots \cdot \overline{x_n}}$$



$$\overline{x_1 \cdot x_2 \cdot \ldots \cdot x_n} = \overline{x_1} + \overline{x_2} + \ldots + \overline{x_n}$$

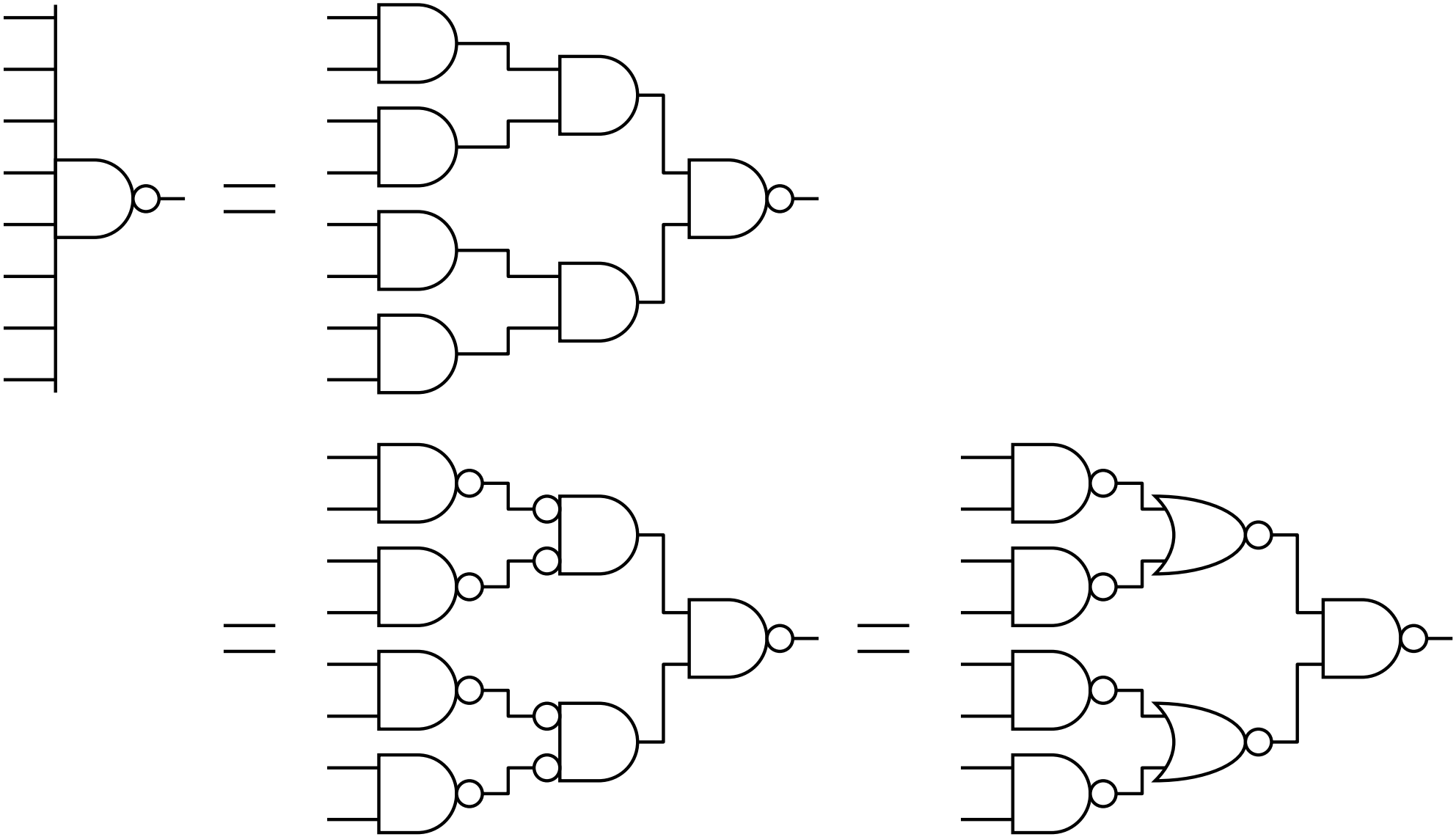


$$x_1 \cdot x_2 \cdot \ldots \cdot x_n = \overline{\overline{x_1} + \overline{x_2} + \ldots + \overline{x_n}}$$



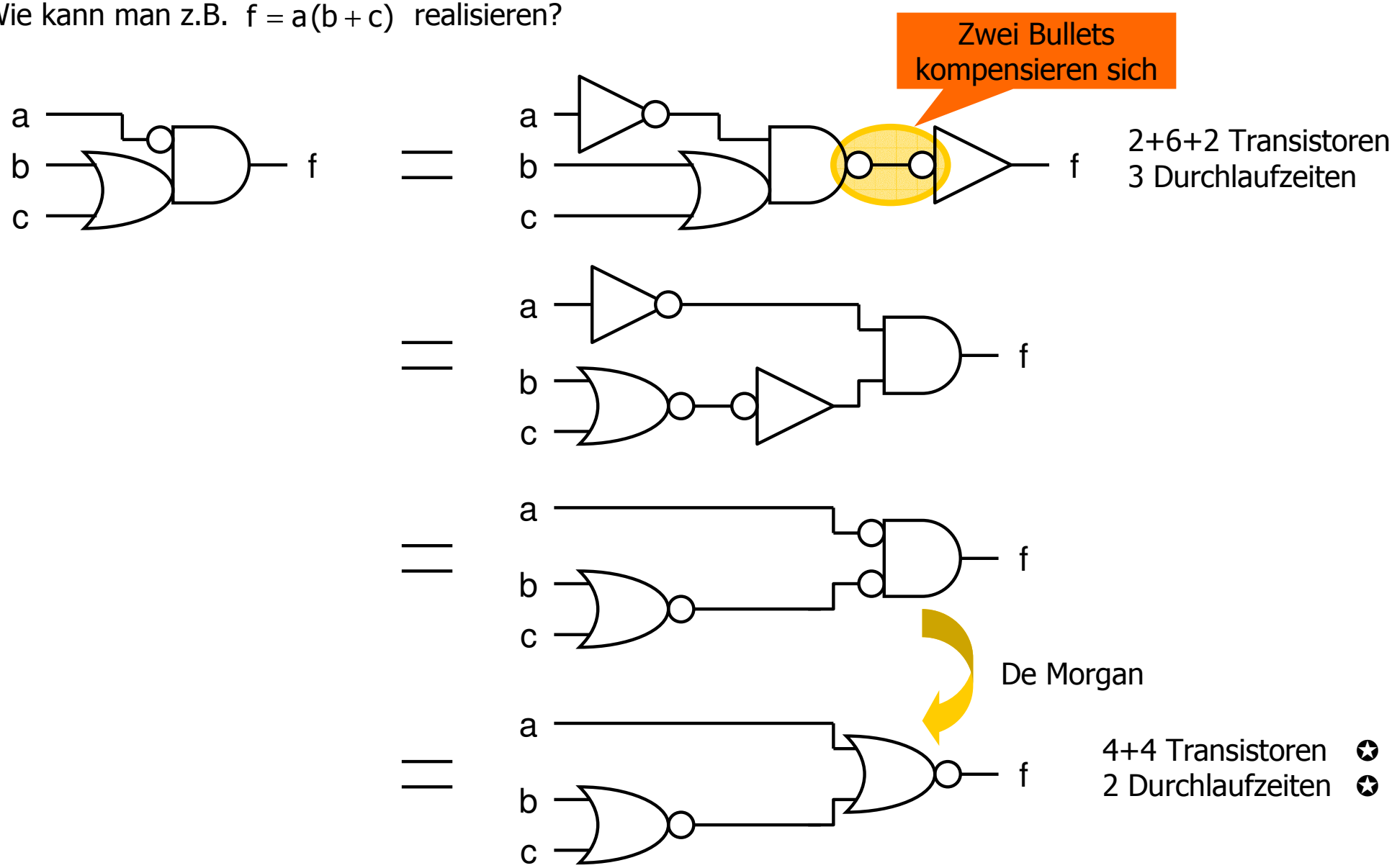
Gatter mit hohem Fan-In

- Gatter mit hohem Fan-In (viele Eingänge) sind langsam. Sie werden oft durch kleinere Gatter ersetzt:



'Umformungen'

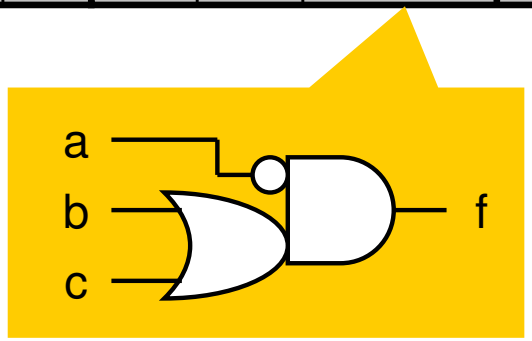
- Wie kann man z.B. $f = \bar{a}(b + c)$ realisieren?

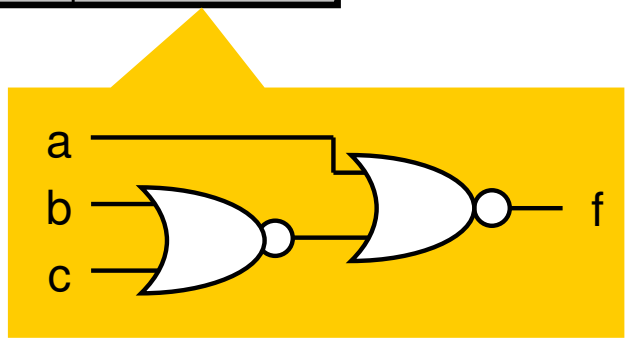


Test

$$f = \bar{a} \cdot (b + c) = \bar{a} \cdot \overline{\overline{(b + c)}} = \overline{a + \overline{(b + c)}}$$

a	b	c	!a	b+c	!a·(b+c)	!(b+c)	!(a + !(b+c))
0	0	0	1	0	**0**	1	**0**
0	0	1	1	1	**1**	0	**1**
0	1	0	1	1	**1**	0	**1**
0	1	1	1	1	**1**	0	**1**
1	0	0	0	0	**0**	1	**0**
1	0	1	0	1	**0**	0	**0**
1	1	0	0	1	**0**	0	**0**
1	1	1	0	1	**0**	0	**0**





Noch ein Beispiel für Logikminimierung

Gewünschte Schaltung:

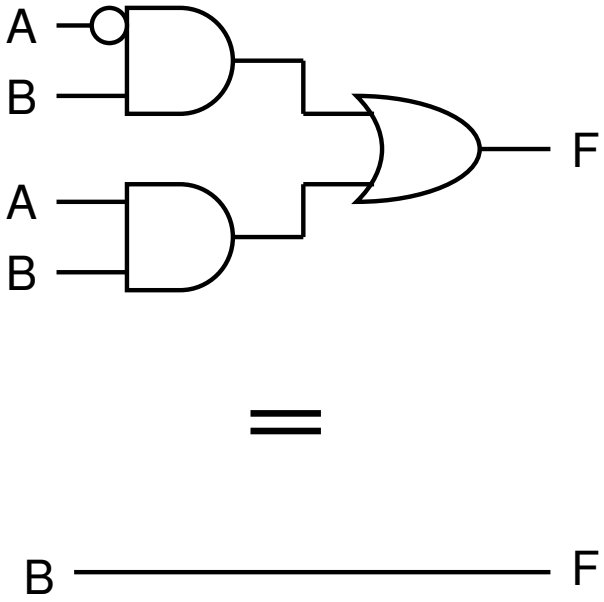


Tabelle:

A	B	F
0	0	0
0	1	1
1	0	0
1	1	1

Funktion:

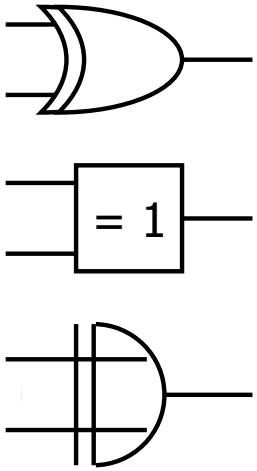
$$F = !A \cdot B + A \cdot B$$
$$= (!A + A) \cdot B$$
$$= 1 \cdot B$$
$$= B$$

XOR/XNOR aus NAND Gattern

XOR

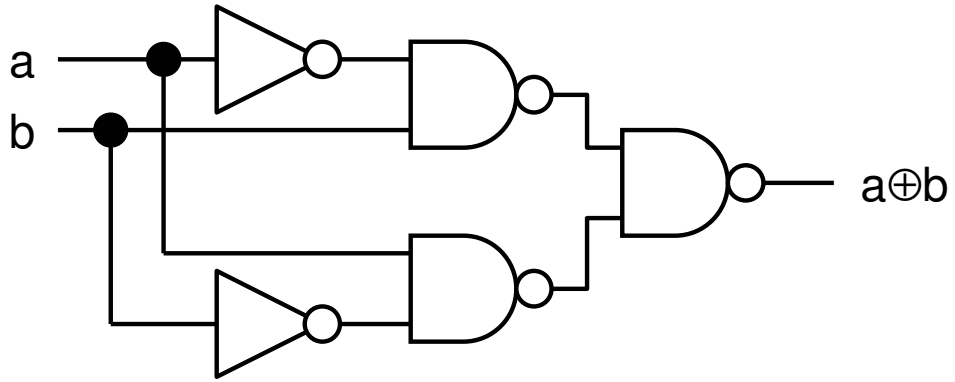
a	b	a ⊕ b
0	0	0
0	1	1
1	0	1
1	1	0

Schaltsymbole



Umformung für die Verwendung von NAND Gattern:

$$a \oplus b = \bar{a} \cdot b + a \cdot \bar{b} = \overline{\overline{\bar{a} \cdot b + a \cdot \bar{b}}} = \overline{\overline{\bar{a} \cdot b} \cdot \overline{a \cdot \bar{b}}}$$

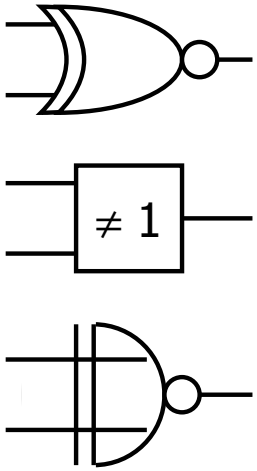


2+2+4+4+4 = 16 Transistoren
3 Durchlaufzeiten - schlecht!

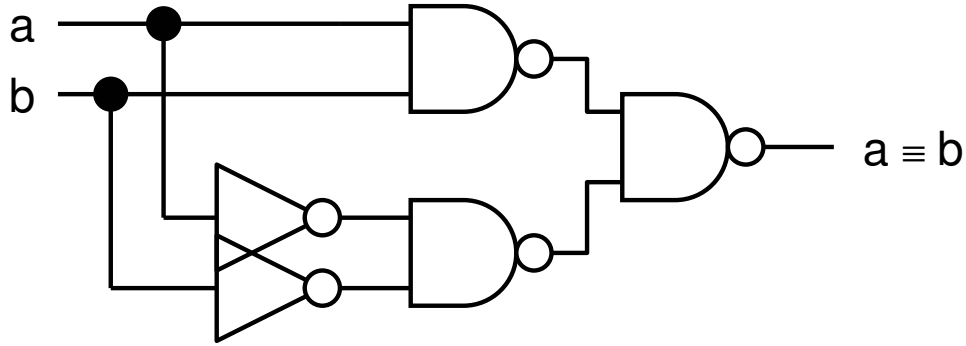
XNOR

a	b	a ≡ b
0	0	1
0	1	0
1	0	0
1	1	1

Schaltsymbole



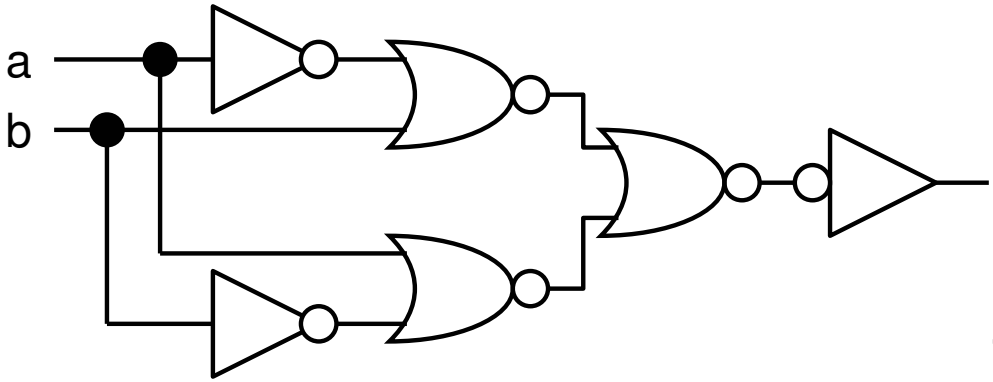
$$a \equiv b = \bar{a} \oplus b$$



XOR/XNOR mit NOR Gattern

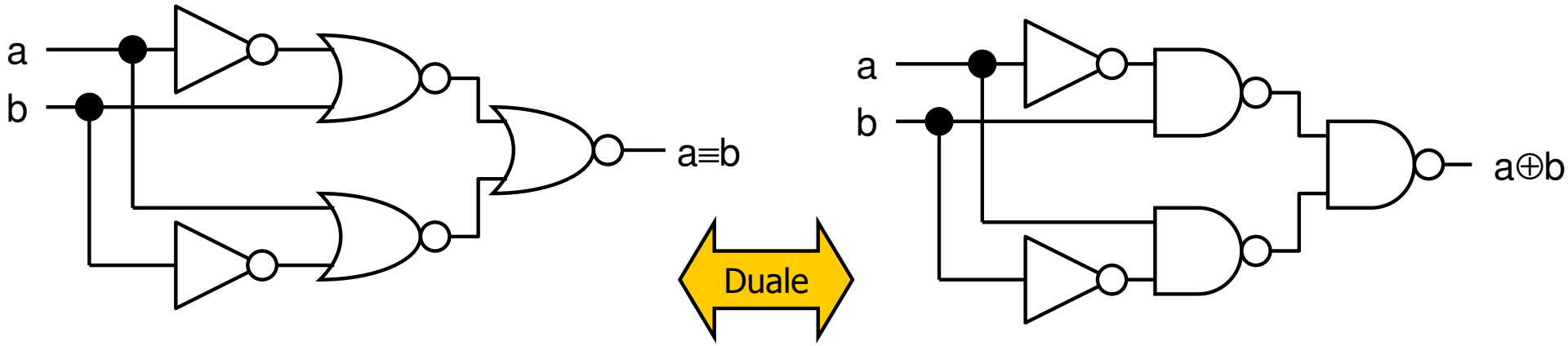
XOR Umformung für die Verwendung von NOR Gattern, z.B.:

$$a \oplus b = \overline{\overline{a \cdot b} \cdot \overline{a \cdot b}} = \overline{(\bar{a} + \bar{b}) \cdot (a + \bar{b})} = \overline{(a + \bar{b})} + \overline{(\bar{a} + b)}$$



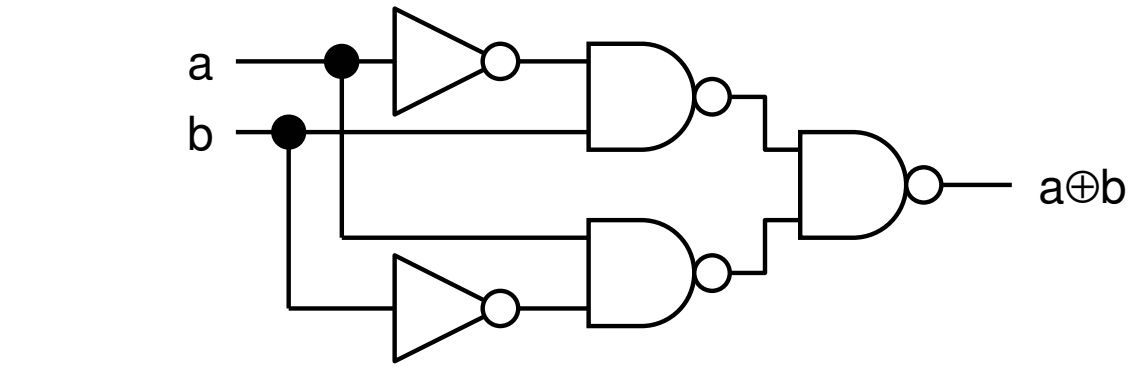
2+2+4+4+4+2 = 18 Transistoren
4 Durchlaufzeiten – sehr schlecht!

XNOR $a \equiv b = \overline{a \oplus b}$

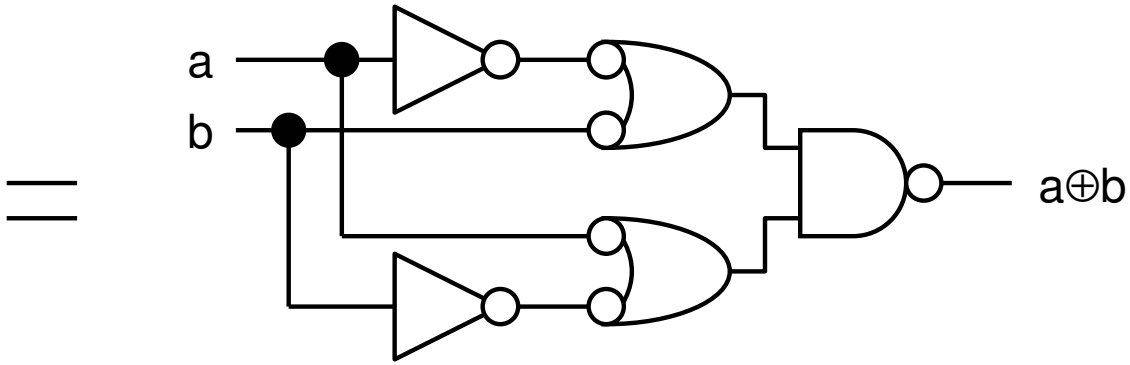


XOR aus gemischten Gattern

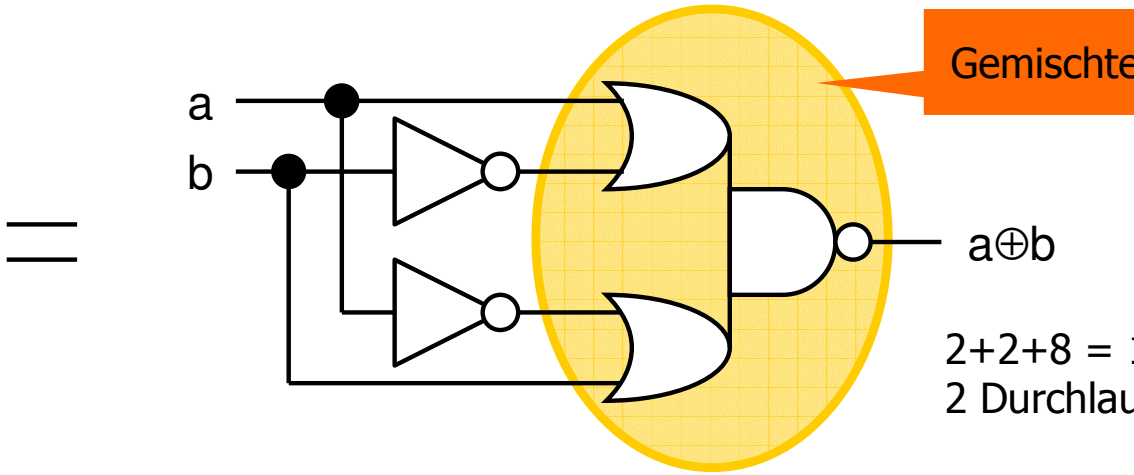
XOR



=



=

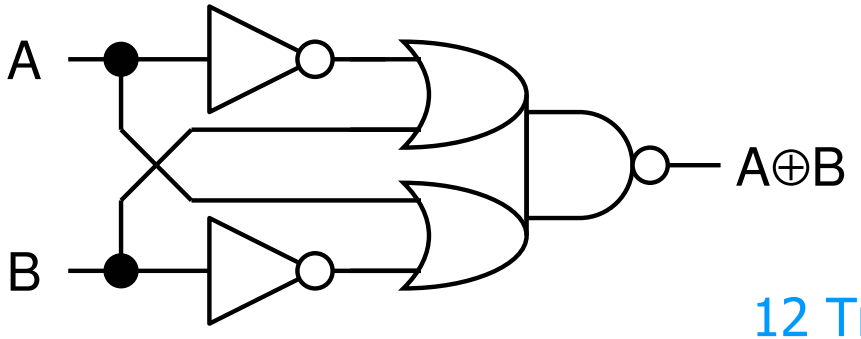


Gemischtes Gatter

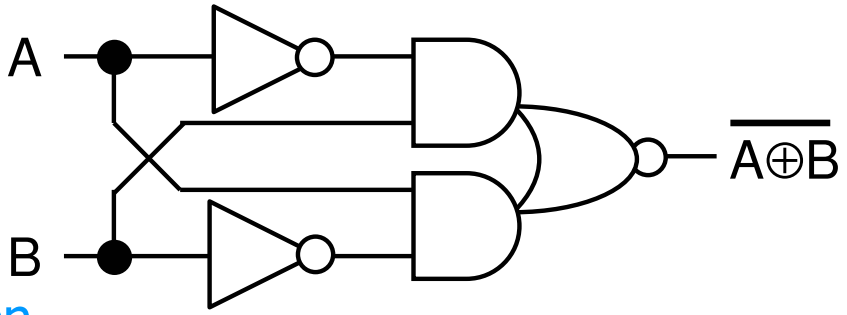
2+2+8 = 12 Transistoren
2 Durchlaufzeiten – nicht schlecht!

XOR und XNOR Gatter

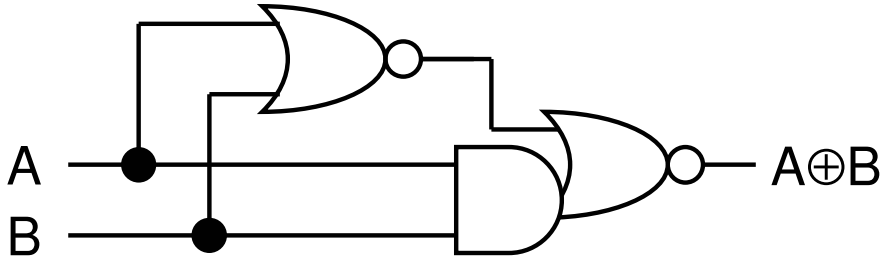
XOR

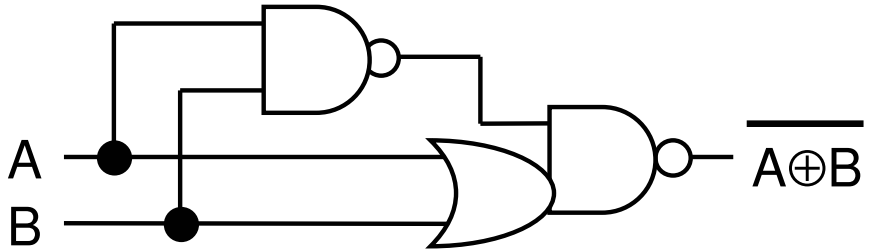


XNOR



12 Transistoren





10 Transistoren

$$\overline{\overline{a+b}+ab} = \overline{\overline{(a+b)}\cdot\overline{ab}} = (a+b)\cdot(\bar{a}+\bar{b}) = a\bar{a}+b\bar{a}+a\bar{b}+b\bar{b} = 0+b\bar{a}+a\bar{b}+0 = b\bar{a}+a\bar{b} = a\oplus b$$

- Alle diese Schaltungen haben 2 Gatter Laufzeit !
- 'Bessere' Schaltungen auf Transistorebene sehen wir später...
- XOR und XNOR werden in Zählern und Vergleichern oft benötigt

Zusammenfassung Gatter

- Die Grundelemente sind Inverter, NAND und NOR.
- Dazu gibt es gemischte Funktionen. Alle sind ‚am Ausgang' invertierend.
- In CMOS Technologie werden NMOS und PMOS Transistoren benutzt. Ein Netz aus NMOS-Transistoren erzeugt die Nullen, ein Netz aus PMOS-Transistoren erzeugt die Einsen.
- Die NMOS und PMOS Netze sind zueinander dual.
- Ein ‚einfaches' CMOS Gatter mit N Eingängen hat N NMOS und N PMOS Transistoren

- XOR und XNOR Gatter sind komplizierter, sie erfordern in einer 'klassischen' Implementierung mindestens 2 Gatterebenen.

- Jede Funktion kann auf mehrere Arten implementiert werden.
- Die Gesetze der Boole'schen Algebra werden zur Umformung benutzt.
- Der Übergang von NAND zu NOR erfolgt mit den De Morgan'schen Regeln.
- Die 'optimale' Darstellung hängt z.B. von der verwendeten Technologie ab. (PAL, FPGA, Standard Zellen)
- Sie ist u.U. nicht einfach zu finden.

Kenngrößen von Inverter und Gattern

- Gatter und Inverter werden u.a. charakterisiert durch:

- Die **Verzögerung**
  - Diese ist i.a. für unterschiedliche Flanken unterschiedlich
  - Bei mehreren Eingängen ist sie i.a. für verschiedene Eingänge unterschiedlich
  - Sie hängt von der Lastkapazität ab und wird daher oft für unterschiedliche Lasten angegeben
  - Sie hängt auch von der Anstiegszeit der Eingangssignale ab
- Den **Leistungsverbrauch**
  - Er hängt bei CMOS stark von der Schaltungsaktivität ab und wird daher oft in Watt/MHz angegeben
- Die **Fläche** des Layouts (bei Chips)
  - So kann die Gesamtfläche eines Designs abgeschätzt bzw. optimiert werden
- Die **Eingangskapazität**
  - Sie wird benötigt, um die Belastung der vorherigen Stufe zu berechnen
- Die **Eingangsströme** (falls in die Eingänge dc-Strom fließt, wie z.B. bei TTL)
  - Bei TTL sind sie unterschiedlich für einen high- oder low-Pegel am Eingang (mehr später...)
- Die verfügbaren **Ausgangsströme**
  - Diese sind meist unterschiedlich für high- und low-Pegel am Ausgang
- Die **Anstiegszeiten** der Ausgangssignale
  - Sind eine Funktion der Lastkapazität

Beispiel für Kenngrößen: TTL Gatter

SN5400, SN7400
QUADRUPLE 2-INPUT POSITIVE-NAND GATES

recommended operating conditions

		SN5400			SN7400			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.75	5	5.25	V
V_{IH}	High-level input voltage	2			2			V
V_{IL}	Low-level input voltage			0.8			0.8	V
I_{OH}	High-level output current			−0.4			−0.4	mA
I_{OL}	Low-level output current			16			16	mA
T_A	Operating free-air temperature	−55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS†	SN5400			SN7400			UNIT
		MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IK}	V_{CC} = MIN, I_I = −12 mA			−1.5			−1.5	V
V_{OH}	V_{CC} = MIN, V_{IL} = 0.8 V, I_{OH} = −0.4 mA	2.4	3.4		2.4	3.4		V
V_{OL}	V_{CC} = MIN, V_{IH} = 2 V, I_{OL} = 16 mA		0.2	0.4		0.2	0.4	V
I_I	V_{CC} = MAX, V_I = 5.5 V			1			1	mA
I_{IH}	V_{CC} = MAX, V_I = 2.4 V			40			40	μA
I_{IL}	V_{CC} = MAX, V_I = 0.4 V			−1.6			−1.6	mA
I_{OS}§	V_{CC} = MAX	−20		−55	−18		−55	mA
I_{CCH}	V_{CC} = MAX, V_I = 0 V		4	8		4	8	mA
I_{CCL}	V_{CC} = MAX, V_I = 4.5 V		12	22		12	22	mA

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
§ Not more than one output should be shorted at a time.

switching characteristics, V_{CC} = 5 V, T_A = 25°C (see note 2)

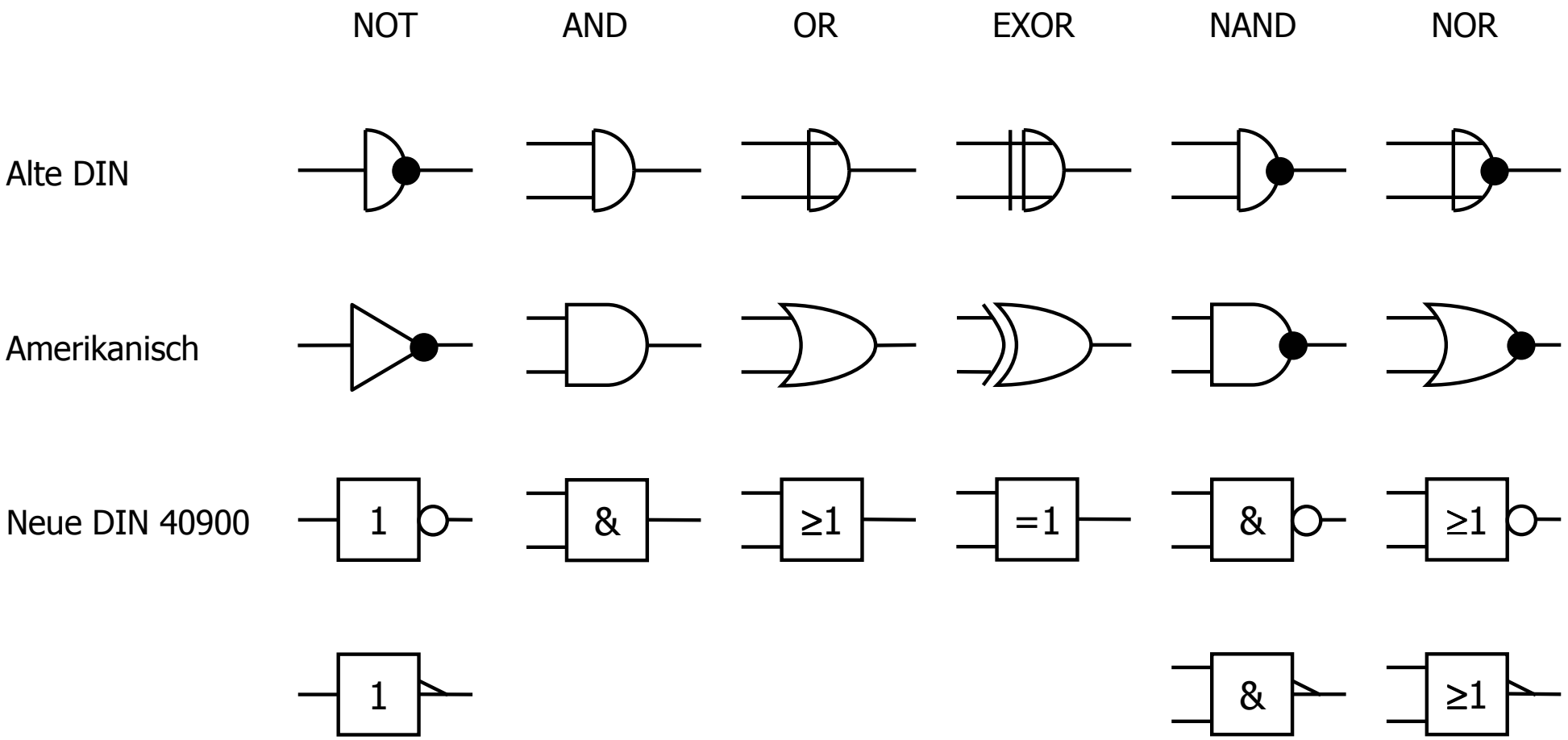
PARAMETER	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{PLH}	A or B	Y	R_L = 400 Ω, C_L = 15 pF		11	22	ns
t_{PHL}					7	15	ns

Ungewöhnliche Symbole:
V_{IK}: input clamp voltage
I_I: input current at maximum input voltage
I_{OS}: short circuit output current

$\langle I_{CC} \rangle$ = 8 mA (Chip)
$\langle P \rangle$ = 5Vx2 mA
d.h. **10mW**/Gate

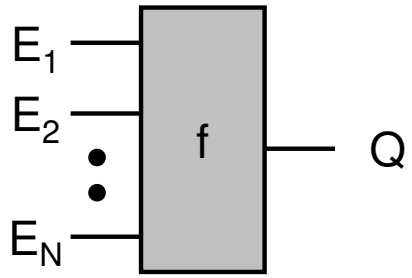
t_{PLH} = 11 ns
t_{PHL} = 7 ns

Zusammenfassung Schaltsymbole



Kennen wir alle Verknüpfungen?

Verknüpfungen von N Eingängen ⇒ 1 Ausgang



- **N** Eingangsvariablen können M = $\mathbf{2^N}$ verschiedene Muster annehmen
  (Jeder hinzukommende Eingang verdoppelt die Zahl der Muster: 0xx...xx, 1xx...xx)

  Für N=1: Muster = {0, 1}
  Für N=2: Muster = {00, 01, 10, 11}

- Eine Verknüpfung **f** weist dem Ausgang für jedes der M Eingangsmuster einen Wert (0 oder1) zu.
  Es gibt daher $\mathbf{2^M}$ mögliche Verknüpfungen.

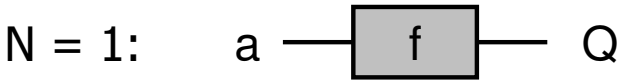
Bei N Eingangsvariablen gibt es 2^{2^N} mögliche Verknüpfungen (Funktionen) f

N:	0	1	2	3	4
M:	1	2	4	8	16
f:	2	4	16	256	65536

  Die Möglichkeiten steigen also SEHR SCHNELL an. Man kann nicht für jede Möglichkeit ein Bauteil haben!

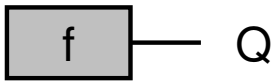
N = 0,1

N = 1:



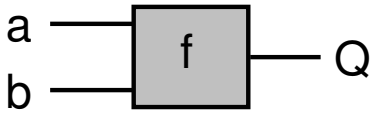
a = 01	Funktion	Bezeichnung	englisch
Q = 00	0	Nie	false
01	a	Identität	
10	!a	Negation	NOT
11	1	Immer	true

N = 0:



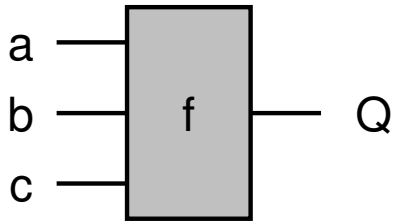
	Funktion	Bezeichnung	englisch
Q = 0	0	Nie	false
1	1	Immer	true

N = 2



a = 0011 b = 0101	Funktion	Bezeichnung	englisch
0000	0	Nie	false
0001	a·b	Konjuntion	AND
0010	a·!b		
0011	a	Identität	
0100	!a·b		
0101	b	Identität	
0110	a⊕b	Antivalenz	XOR
0111	a+b	Disjunktion	OR
1000	!(a+b)	Negierte Disjunktion	NOR
1001	a≡b	Äquivalenz	XNOR
1010	!b	Negation	NOT
1011	a+!b		
1100	!a	Negation	NOT
1101	!a+b		
1110	!(a·b)	Negierte Konjunktion	NAND
1111	1	Immer	true

N = 3



a = 00001111 b = 00110011 c = 01010101	Funktion	Bezeichnung
00000000		
00000001		
00000010		
00000011		
00000100		
●		
●		
●		

Zu viele Möglichkeiten –
wir brauchen eine allgemeinere
Beschreibung!
z.B. Karnaugh-Tafeln...