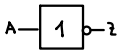
Kombinatorische Logik

Bildet Systeme ohne Speicher ab, welche einfache logische Operationen mit Gattern (Gates) ausführen.

Inverter

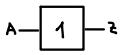
$Z = !A$



A	Z
0	1
1	0

Buffer

$Z = A$



A	Z
0	0
1	1

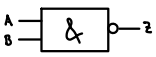
AND

$Z = A \& B$

A	B	Z
0	0	0
0	1	0
1	0	0
1	1	1

NAND

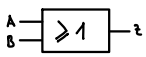
$Z = !(A \& B)$



A	B	Z
0	0	1
0	1	1
1	0	1
1	1	0

OR

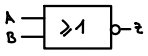
$Z = A \# B$



A	B	Z
0	0	0
0	1	1
1	0	1
1	1	1

NOR

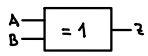
$Z = !(A \# B)$



A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0

XOR / EXOR

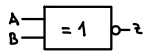
$Z = A \$ B$



A	B	Z
0	0	0
0	1	1
1	0	1
1	1	0

XNOR / EXNOR

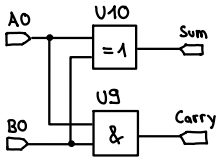
$Z = !(A \$ B)$



A	B	Z
0	0	1
0	1	0
1	0	0
1	1	1

1-Bit Halb Addierer

Addition von zwei 1-Bit Inputs



$Sum = A0 \$ B0$

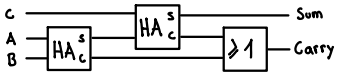
$Carry = A0 \& B0$

A0	B0	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

	0	1	1	1	A0
+	0	1	0	1	B0
0	1	1	1		Carry
0	1	1	0	0	Sum

1-Bit Voll-Addierer

Addition von zwei 1-Bit Inputs mit Carry-in



$Sum = A \$ B \$ C$

$Carry = (A \& B) \$ ((A \$ B) \& C)$

	0	1	1	1	A0
+	0	1	0	1	B0
0	1	1	1		Carry (C)
0	1	1	0	0	Sum

C	A	B	Sum	Carry
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

Boolesche Gleichung

Augenzahl	$b_2 b_1 b_0$	L1	L2	L3	L4	L5	L6	L7
0	000	0	0	0	0	0	0	0
1	001	0	0	0	0	0	0	1
2	010	1	0	0	0	0	1	0
3	011	1	0	0	0	0	1	1
4	100	1	0	1	1	0	1	0
5	101	1	0	1	1	0	1	1
6	110	1	1	1	1	1	1	0
7	111	1	1	1	1	1	1	1

Boolesche Funktion

$L1 = !((!b_2 \& !b_1 \& !b_0) \# (!b_2 \& !b_1 \& b_0)) = b_2 \# b_1$

$L2 = (b_2 \& b_1 \& !b_0) \# (b_2 \& b_1 \& b_0) = b_2 \& b_1$

$L3 = b_2$

$L4 = L3$

$L5 = L2$

$L6 = L1$

$L7 = b_0$

! Disjunkt: AND in Klammer, OR ausserhalb
z.B. $(b_2 \& b_1) \# (b_2 \& !b_1)$

! Konjunkt: OR in Klammer, AND ausserhalb
z.B. $(b_2 \# b_1) \& (b_2 \# !b_1)$

"don't care" markierte Elemente mit "x" können für die Gleichung ignoriert werden.