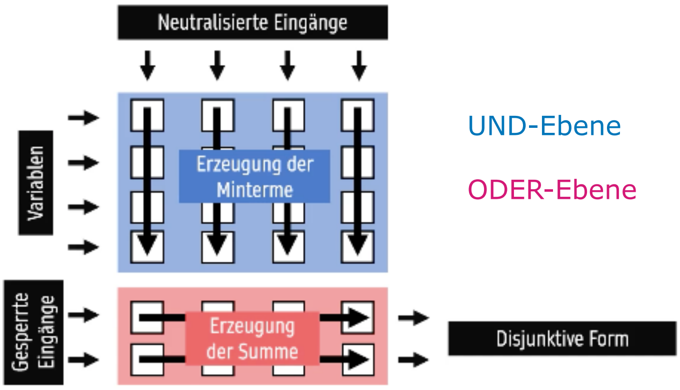
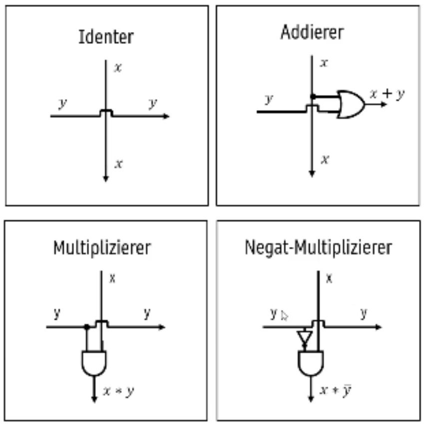
• PLA (Programmierbares Logisches Array)





PLA Konstruktion:

Schritt 1:
Wie viele verschiedene Minterme werden benötigt?
→minimale Anzahl der Spalten des PLAs

Schritt 2:
Wie viele Schaltfunktionen sollen realisiert werden?
→Anzahl Variablen + Anzahl Schaltfunktionen = Anzahl Zeilen

Schritt 3:
Ausfüllen des PLAs

• Resolutionsregel

Voraussetzung: Min- oder Maxterme unterscheiden sich nur in einer Komponente

Hintereinanderausführung von Distributivgesetz, Komplementärgesetz und Neutralitätsgesetz

$f_1(x_1, x_2, x_3)$		$f_2(x_1, x_2, x_3)$
$= x_1x_2x_3 + \overline{x_1}x_2x_3$		$= (x_1 + x_2 + x_3) * (\overline{x_1} + x_2 + x_3)$
$= (x_1 + \overline{x_1})x_2x_3$	Distributivgesetz	$= (x_1 * \overline{x_1}) + x_2 + x_3$
$= 1\, x_2x_3$	Komplementärgesetz	$= 0 + x_2 + x_3$
$= x_2x_3$	Neutralitätsgesetz	$= x_2 + x_3$

• Karnaugh-Diagramm

- Ein Karnaugh-Diagramm ist eine graphische Darstellung der Funktionstafel einer Funktion f
- zwei zyklisch benachbarte Spalten oder Zeilen unterscheiden sich nur genau in einer komplementären Variable

x_3x_4 \ x_1x_2	00	01	11	10
10	1	1	1	1
11	1			1
01	1			1
00				

- Blöcke der Größe $2^n \times 2^m$
- Möglichst große Blöcke
- Mit möglichst wenigen Blöcken alle Einsen abdecken
- Blöcke können auch über die Ränder hinweg verlaufen

$$f(x_1, x_2, x_3, x_4) = x_3\overline{x_4} + \overline{x_2}x_4$$

$y_1 = (x_1x_2\overline{x_3}) + (x_1\overline{x_2}\overline{x_3}) + (\overline{x_1}\overline{x_2}\overline{x_3}) + (\overline{x_1}x_2\overline{x_3}) + (x_1\overline{x_2}x_3) + (x_1x_2x_3)$

x_3 \ x_1x_2	00	01	11	10
0	1	1	1	1
1			1	1

$\longrightarrow y_1 = \overline{x_3} + x_1$

• Don’t Care Argumente

- Nicht bei jeder Schaltfunktion sind alle der 2^n möglichen Kombinationen festgelegt
- Im Karnaugh-Diagramm kennzeichnen wir diese Fälle mit „D“
- Mit D gekennzeichnete Felder **können** verwendet werden **müssen** aber nicht

$f(a,b,c,d) = \boxed{c}$

	$\bar{a}\bar{b}$	$\bar{a}b$	ab	$a\bar{b}$
$c\bar{d}$	1	1	1	1
cd	1	1	D	1
$\bar{c}d$				
$\bar{c}\bar{d}$				

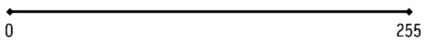
• Quine-McCluskey Verfahren

Schritt 1: Implikanten bestimmen
Schritt 2: Implikanten verkürzen => Primimplikanten
Schritt 3: Mit Primimplikanten verkürzte Boolsche Funktion bestimmen

• Darstellung ganzer Zahlen

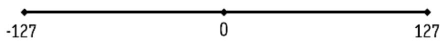
Sign-Magnitude Darstellung	Einerkomplement Darstellung	Zweierkomplement Darstellung

Beispiel mit 8 Bits:
Zahlen können von 0 - 255 dargestellt werden



- Sign-Magnitude Darstellung

Beispiel mit 8 Bits:
Das höchstwertigste Bit zeigt das Vorzeichen an. Die restlichen Bits (im Beispiel 7 Bits) werden für die Darstellung der Zahl verwendet.
Nachteil: Es gibt 2 Darstellungen für die Null (+0 und -0)



- Einerkomplement Darstellung

Beispiel mit 8 Bits:
Der Zahlenbereich (256) wird in 2 Abschnitte geteilt. Dabei entstehen 2 Zahlenbereiche mit je 128 Zahlen.
> jeder dieser Zahlenbereiche enthält die 0

<u>Einerkomplement Operation:</u>
Bitweise invertieren der positiven Darstellung der Zahl:
Nur bei negativen Zahl einerkomplement Darstellung anwenden.

$$K_1(50) \quad = 00110010$$
$$K_1(-50) \ = 11001101$$

- Zweierkomplement Darstellung:

Beispiel mit 8 Bits:
Der Zahlenbereich (256 Zahlen) wird in 2 Abschnitte geteilt.
Der negative Zahlenbereich wird um 1 erweitert -> keine 2 Darstellung für die 0 wie bei der Einerkomplement Darstellung.

<u>Zweierkomplement Operation:</u>
Bitweise invertieren und +1 rechnen der positiven Darstellung der Zahl.

$$K_2(50) \quad = 00110010$$
$$K_2(-50) \ = 11001101 + 1 = 11001110$$

Unterschied zwischen „2er Komplement“ und „2er Komplement-Darstellung“:

- **2er-Komplement** bezeichnet Rechenoperation auf einem Bitmuster (nämlich: Bits invertieren und 1 addieren)
- **2er-Komplement-Darstellung** ist eine Art der Zahlendarstellung, in der bei der Darstellung negativer Zahlen das 2er-Komplement zum Einsatz kommt
- Leider wird oftmals „2er-Komplement“ gesagt, wenn eigentlich „2er-Komplement-Darstellung“ gemeint ist

Beispiele:

Addiere $(-56)_{10}$ und $(-72)_{10}$ binär (Einerkomplement-Darstellung)

$$(56)_{10} = (00111000)_2$$
$$\rightarrow (-56)_{10} = (11000111)_2$$

$$(72)_{10} = (01001000)_2$$
$$\rightarrow (-72)_{10} = (10110111)_2$$

```
  11000111   (-56)10
+ 10110111   (-72)10
1      111
----------
  01111110
+ 00000001   Übertrag
----------
  01111111   (+127)10
```

Es hat ein Überlauf stattgefunden, da zwei negative Zahlen addiert werden, jedoch eine positive Zahl als Ergebnis der Addition erhalten wird.

Addiere $(-56)_{10}$ und $(-72)_{10}$ binär (Zweierkomplement-Darstellung)

$$(56)_{10} = (00111000)_2$$
$$\rightarrow (-56)_{10} = (11000111)_2 + 1 = (11001000)_2$$

$$(72)_{10} = (01001000)_2$$
$$\rightarrow (-72)_{10} = (10110111)_2 + 1 = (10111000)_2$$

```
  11001000   (-56)10
+ 10111000   (-72)10
(1)1111
----------
  10000000   (-128)10
```

Übertrag weglassen

Es hat **kein** Überlauf stattgefunden, da das Ergebnis der Addition mit den zur Verfügung stehenden Bits dargestellt werden kann.

• Darstellung reeller Zahlen

Festkommazahlen	Gleitkommazahlen	
Sign-Magnitude-Darstellung	Gleitkomma-darstellung	IEEE 754 Standard

- Sign-Magnitude Darstellung

Das Komma steht an beliebiger, aber fester Stelle

$$111{,}011 = -(1 * 2^1 + 1 * 2^0 + 0 * 2^{-1} + 1 * 2^{-2} + 1 * 2^{-3})$$

Probleme

- Man kann mit einer bestimmten Anzahl von Bits nur einen beschränkten Wertebereich abdecken.
- Es muss separat gekennzeichnet oder allgemeingültig für alle Darstellungen vereinbart werden, an welcher Stelle sich das Komma befindet.
- Wenn man sehr große und sehr kleine Zahlen Darstellen möchte braucht man sehr viele Bits

- Gleitkommadarstellung

Schritt 1: Normalisieren

$$1,x \ldots x * 2^{y \ldots y}$$

Schritt 2: Sign, Exponent und Significand eintragen

$$z = \begin{cases} (-1)^S * (1 + Significand) * 2^E, falls\ E \neq 0 \\ (-1)^S * Significand * 2^E, falls\ E = 0 \end{cases}$$

Zweierkomplement Darstellung

31 S	30 29 28 27 26 25 24 23 E	22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 Significand
1Bit	8 Bit	23 Bit

Beispiel

$$1,101 * 2^{-1}$$

Schritt 2: Sign, Exponent und Significand eintragen

$$z = \begin{cases} (-1)^S * (1 + Significand) * 2^E, falls\ E \neq 0 \\ (-1)^S * Significand * 2^E, falls\ E = 0 \end{cases}$$

Sign	Exponent	Significand
0	11111111	10100000000000000000000