HOCHSCHULE KONSTANZ
FAKULTÄT INFORMATIK

Digitaltechnik

Vorlesung

Prof. Dr.-Ing. I. Schoppa

Organisation

- Prof. Dr.-Ing. Irenäus Schoppa
- Fakultät Informatik
- Gebäude F, Zimmer F124
- Telefon: 07531 / 206 – 644
- E-Mail: irenaeus.schoppa@htwg-konstanz.de
- Sprechstunden nach Vereinbarung

- Unterlagen:
  \\merkur(.fh-konstanz.de)\lehre\schoppa\DIGI\Vorlesung
  \\merkur(.fh-konstanz.de)\lehre\schoppa\DIGI\Uebung

Organisation

Zeit	Mo	Di	Mi	Do	Fr
08:00 09:30		VL DIGI			
09:45 11:15					
11:30 13:00					
14:00 15:30		UE DIGI	VL DIGI	UE DIGI	
15:45 17:15		UE DIGI		UE DIGI	
17:30 19:00					

Themenübersicht

- Einführung in Digitaltechnik
  - Zahlensysteme
  - Rechenarithmetik
  - Boolesche Algebra
- Schaltnetze
  - Minimierungsverfahren:
    - Graphische Minimierung nach Karnaugh-Veitch
    - Algorithmische Minimierung nach Quine-McCluskey
  - Synthese von Schaltnetzen:
    - Multiplexer, Schaltketten, prog. Logikbausteine
  - Logikfamilien und deren Kenndaten
  - dynamisches Verhalten von Schaltnetzen

Themenübersicht

- Schaltwerke
  - asynchrone Speicherelemente
  - synchrone Speicherelemente
  - Register, Schieberegister
  - Zähler und Frequenzteiler
  - Zustandsautomaten
  - Registertransferoperationen
  - Realisierung von Steuerwerken
  - Synthese von Schaltwerken

- Basisliteratur:
  1. Schoppa, I.: *Vorlesungs- und Übungsunterlagen*, HTWG Konstanz.
  2. Beuth, K: *Elektronik 4. Digitaltechnik*, Vogel Fachbuchverlag, 2006.
  3. Pernards, P.: *Digitaltechnik*, Hüthig Verlag, 1992.
  4. Pernards, P.: *Digitaltechnik 2, Einführung in die Schaltwerke*, Hüthig Verlag, 1995.
- Ergänzende Literatur:
  1. Liebig, H.: *Logischer Entwurf digitaler Systeme*, Springer Verlag, 4. Auflage, 2005.
  2. Borucki, L.: *Digitaltechnik*, Teubner Verlag, 5. Auflage, 2000.
  3. Seifart, M. und Beikirch, H.: *Digitale Schaltungen*, Verlag Technik, 5. Auflage, 1998
  4. Biere, A.; Kroening, D.; Weissenbacher, G.; Wintersteiger, Ch.: *Digitaltechnik: Eine praxisnahe Einführung*, Springer Verlag, 1. Auflage, 2008.

Motivation und Ziel

- Die Digitaltechnik ist ein faszinierendes Gebiet der modernen Elektronik.
- Durch den Einsatz neuer Technologien hat die Digitaltechnik in den letzten Jahren eine rasante Entwicklung erlebt.
- Die Komplexität der heutigen und vor allem zukunftigen Aufgaben erfordert interdisziplinäre Lösungen, die eine Kombination aus Software- und Hardware-Komponenten darstellen (sog. Hardware-Software-Codesign).
- Heutzutage sind Kenntnisse der Digitaltechnik für das Verständnis von vielen technischen Anwendungen unerlässlich (z.B. Mikrocomputertechnik und Telekommunikation).

Motivation und Ziel

- Warum die Digitaltechnik im Informatikstudium?
  - Der Aufbau und die Funktionsweise von Computersystemen ist nicht nur eine wichtige Grundlage für die Informations-, Mikrosystem- und Automatisierungstechnik, sondern hat auch starke Auswirkungen auf Systemsoftware.
  - Im Bereich der eingebetteten Systeme beschäftigt man sich oft mit der Grenze zwischen Hardware und Software. Das ist ein wichtiger Arbeitsmarkt, speziell für Europa. Der Entwicklungsbedarf an eingebetteter Software wird in den kommenden Jahren drastisch steigen.
- Informatiker müssen verstehen, wie informationsverarbeitenden Systeme auf der Hardware-Ebene funktionieren.
- Informatiker sollen nicht nur in der Software kompetent sein, sondern auch in der Hardware.

Motivation und Ziel

- Die Vorlesung *Digitaltechnik*
  - gibt einen breiten Überblick über das Fachgebiet,
  - definiert Grundbegriffe der Digitaltechnik,
  - vermittelt wesentliche theoretische und praktische Grundlagen,
  - umfaßt die Bau- und Funktionsweise digitaler Komponenten,
  - erklärt Methoden und Techniken zur Schaltungsanalyse und Schaltungssynthese,
  - zeigt Möglichkeiten zur systematischen Vorgehensweise bei der Schaltungsminimierung,
  - versucht für das Fachgebiet zu begeistern,
  - ist nicht einfacher aber auch nicht schwieriger als andere Vorlesungen.

Motivation und Ziel

- Der Begriff *Digital* ist abgeleitet
  - vom lateinischen Wort *digitus* = Finger, was soviel bedeutet wie „mit Hilfe der Finger rechnen“,
  - aus dem englischen Wort *digit* = Ziffer oder Stelle, was man als „in Ziffernform“ interpretieren kann.

- Für die Darstellung von digitalen Größen verwendet man abzählbare Elemente.

- Die Digitaltechnik ist ein Teilgebiet der Elektronik und Informationstechnik, das sich mit der Erfassung, Verarbeitung, Darstellung und Übertragung digitaler Signale befasst.

- Analyse digitaler Systeme
  - Analyse bezeichnet die Zerlegung, Zergliederung eines digitalen Systems in seine Bestandteile mit anschließender systematischer Untersuchung des Verhaltens und zwar unter Berücksichtigung von Teilaspekten (z.B. Signallaufzeit, Verlustleistung).

- Synthese (Entwurf) digitaler Systeme
  - Synthese bezeichnet die Vereinigung, Zusammenschaltung von zwei oder mehreren digitalen Komponenten mit bekanntem Verhalten zu einer neuen Einheit, so daß daraus das gewünschte Verhalten resultiert, und zwar unter Einhaltung vorgegebener Randbedingungen (wirtschaftliche oder technologische Aspekte):
    - Herstellungskosten,
    - Verlustleistung,
    - Signallaufzeit (Timing Constraints),
    - Chip-Fläche (Area Constraints)).

- Top-Down-Entwurf (von oben nach unten)
  - iterative oder rekursive Zerlegung eines Ausgangsproblems in einfachere Teilprobleme, um letztendlich die Gesamtlösung aus einfachen Grundfunktionen zusammensetzen zu können.
- Bottom-Up-Entwurf (von unten nach oben)
  - sukzessive (schrittweise) Zusammensetzung von Teilsystemen aus elementaren Bausteinen zu komplexeren Komponenten, bis das gewünschte Systemverhalten erreicht ist.
- In der Praxis werden beide Entwurfsprinzipien zumeist kombiniert eingesetzt.
  - Man versucht beim Top-Down-Entwurf schon möglichst bald Teil-probleme für eine endgütige Realisierung zu erkennen und diese dann im Bottom-Up-Entwurf zu lösen.

Anwendungen der Digitaltechnik

- Consumer-Bereich
  - Fahrkartenautomat, Waschmaschine, „intelligenter“ Kühlschrank, Fotokamera, CD-/DVD-/MP3-Player, Smart-Phone, ...
- Automobile-Bereich
  - ABS, ESP, Motorsteuerung, Automatikgetriebe, „intelligenter“ Allradantrieb (xDrive), Navigationssystem, ...
- Office-Bereich
  - FAX, Telefonanlagen, Drucker, Kopierer, Kassensysteme, Zeiter-fassung, ...
- Medizin-Bereich
  - Herzschrittmacher, Infusionspumpe, Computertomographie, ...
- Wissenschaft/Technik
  - Messgeräte, Signalgeneratoren, Oszilloskope, ...

Anwendungen der Digitaltechnik

- Luft-/Raumfahrt
  - Sateliten, Weltraumsonden, Autopilot, ...
- Automatisierungstechnik
  - Prozeßrechner, speicherprogrammierbare Steuerungen, CNC-Technik, Industrieroboter, ...
- Nachrichten-/Kommunikationstechnik
  - Kommunikationsnetze (ISDN, DSL, Internet), Mobilfunk (GPS, UMTS, LTE), Sprach- und Bildübertragung (DVB-T/-S/-C, IPTV), ...
- Computertechnik
  - Universalrechner, Minicomputer (Workstation), Personal Computer (Mikrocomputer), Notebook, ...
- Militärbereich
  - Sonar-/Radarsysteme, Feuerleitsysteme, unbemannte Flugkörper, ..

- Eine Zahl wird durch Symbole (i.d.R. Ziffern) dargestellt, die nach bestimmten Regeln aneinander gereiht werden.

- Ein *Zahlensystem* wird durch die Gesamtheit der Symbole und zugehörigen Regeln gebildet.

- In der Praxis und technischen Anwendungen sind *Stellenwert*- oder *Positionssysteme* relevant.

- Im Rahmen von *Digitaltechnik* lernen wir
  - die wichtigsten Zahlensysteme,
  - Grundrechenarten im Dualsystem und
  - Zahlenformate

- Stellenwert-/Positionssystem
  - heute allgemein verwendetes Zahlensystem
  - Dezimalsystem: zehn verschiedene Symbole (arabische Ziffern) $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
  - Notationregel: Jeder Stelle (Position) innerhalb einer Zahl ist ein bestimmter Wert zugeordnet. Man bezeichnet ihn als Stellenwert.
- eine n-stellige nicht negative Zahl zur Basis b wird notiert:

$$(X)_b = (x_{n-1}x_{n-2}\ldots x_i \ldots x_1 x_0)_b$$

$$(X)_b = x_{n-1} \cdot b^{n-1} + x_{n-2} \cdot b^{n-2} + \ldots\ x_i \cdot b^i \ \ldots + x_1 \cdot b^1 + x_0 \cdot b^0$$

  - b Basis des Stellenwertsystems, $b \in \{2, 3, \ldots, \}$
  - n Anzahl der Ziffernstellen
  - x_i Ziffer an der i-ten Position, $x_i \in \{0, 1, 2, \ldots, b-2, b-1\}$
  - x_{n-1} Ziffer an der höchstwertigen Stelle
  - x_0 Ziffer an der niedrigstwertigen Stelle

- gebräuchliche Zahlensysteme in der Digitaltechnik
  - Duales Zahlensystem
    - Basis b=2, Ziffernmenge = {0, 1}
  - Oktales Zahlensystem
    - Basis b=8, Ziffernmenge = {0, 1, 2, 3, 4, 5, 6, 7}
  - Dezimales Zahlensystem
    - Basis b=10, Ziffernmenge = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
  - Hexadezimales Zahlensystem
    - Basis b=16, Ziffernmenge = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}
    - mit A := 10, B := 11, C := 12, D := 13, E := 14 und F := 15
- Beispiele

$$(11101)_2 = 1 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$$

$$(527)_8 =$$

$$(FE01)_{16} =$$

- Umwandlung einer nicht negativen Dualzahl in eine Hexadezimalzahl (b=16)
  - Besteht eine Dualzahl aus einer nicht durch 4 ohne Rest teilbaren Anzahl von Stellen, so wird sie linksseitig mit Nullen aufgefüllt.
  - Eine Dualzahl wird in 4er-Blöcke (sog. Tetraden) aufgeteilt, beginnend mit der niedrigstwertigen Stelle.
  - Jede Tetrade wird separat in ihre hexadezimale Darstellung laut der folgenden Tabelle transformiert:

Tetrade	0000	0001	0010	0011	0100	0101	0110	0111
HEX	0	1	2	3	4	5	6	7

Tetrade	1000	1001	1010	1011	1100	1101	1110	1111
HEX	8	9	A	B	C	D	E	F

  - Die Hexadezimaldarstellung der Ausgangszahl ergibt sich aus der Konkatenation (Verkettung) dieser Transformationen unter Berücksichtigung der ursprünglichen Reihenfolge.

- Umwandlung einer nicht negativen Dualzahl in eine Oktalzahl (b=8)
  - Besteht eine Dualzahl aus einer nicht durch 3 ohne Rest teilbaren Anzahl von Stellen, so wird sie linksseitig mit Nullen aufgefüllt.
  - Eine Dualzahl wird in 3er-Blöcke aufgeteilt, beginnend mit der niedrigstwertigen Stelle.
  - Jeder 3er Block wird separat in seine oktale Darstellung laut der folgenden Tabelle transformiert:

3er Block	000	001	010	011	100	101	110	111
OCT	0	1	2	3	4	5	6	7

  - Die Oktaldarstellung der Ausgangszahl ergibt sich aus der Konkatenation (Verkettung) dieser Transformationen unter Berücksichtigung der ursprünglichen Reihenfolge.

Beispiel: $(X)_2 = (1101101001)_2 \quad (X)_{16} = ?$

$(X)_2 =$

$(X)_{16} =$

Beispiel: $(X)_2 = (1101111011)_2 \quad (X)_8 = ?$

$(X)_2 =$

$(X)_8 =$

Beispiel: $(X)_{16} = (1F5)_{16} \quad (X)_2 = ?$

$(X)_2 =$

$(X)_2 =$

- Umwandlung einer n-stelligen nicht negativen Zahl X zur Basis b in eine Dezimalzahl Y mit der Direktmethode

  $(Y)_{10} = \Sigma(x_i)_b \cdot b^i$ für $0 \leq i \leq n-1$

  Beispiel: $(X)_2 = (100101)_2$, $(Y)_{10} = ?$

  $(Y)_{10} =$

  $(Y)_{10} =$

  $(Y)_{10} =$

  Beispiel: $(X)_5 = (3412)_5$, $(Y)_{10} = ?$

  $(Y)_{10} =$

  $(Y)_{10} =$

  $(Y)_{10} =$

- Umwandlung einer n-stelligen nicht negativen Zahl X zur Basis b in eine Dezimalzahl Y nach dem Horner-Schema

$(Y)_{10} = \Sigma(x_i)_b \cdot b^i \quad$ für $0 \leq i \leq n\text{-}1$

$\Sigma(x_i)_b \cdot b^i = x_{n-1} \cdot b^{n-1} + x_{n-2} \cdot b^{n-2} + \ldots + x_1 \cdot b^1 + x_0 \cdot b^0$

$= b \cdot (x_{n-1} \cdot b^{n-2} + x_{n-2} \cdot b^{n-3} + \ldots + x_1) + x_0$

$= b \cdot (b \cdot (x_{n-1} \cdot b^{n-3} + x_{n-2} \cdot b^{n-4} + \ldots) + x_1) + x_0$

$= \ldots$

$= b \cdot (b \cdot \ldots\ b \cdot (x_{n-1} \cdot b + x_{n-2}) + \ldots + x_1) + x_0$

ergibt

$(Y)_{10} := 0$

$(Y)_{10} := (Y)_{10} \cdot b + (x_i)_b \quad$ für $i = \{n\text{-}1, n\text{-}2, \ldots, 1, 0\}$

Beispiel $(X)_2 = (101011)_2$, $(Y)_{10} = ?$, $b := 2$, $n := 6$

$(Y)_{10} := 0$

$(Y)_{10} := (Y)_{10} \cdot 2 + (x_i)_2$ für $i = \{5, 4, 3, 2, 1, 0\}$

$(Y)_{10} := (Y)_{10} \cdot 2 + 1 = 0 \cdot 2 + 1 = 1$

$(Y)_{10} := (Y)_{10} \cdot 2 + 0 = 1 \cdot 2 + 0 = 2 + 0 = 2$

$(Y)_{10} := (Y)_{10} \cdot 2 + 1 = 2 \cdot 2 + 1 = 4 + 1 = 5$

$(Y)_{10} := (Y)_{10} \cdot 2 + 0 = 5 \cdot 2 + 0 = 10 + 0 = 10$

$(Y)_{10} := (Y)_{10} \cdot 2 + 1 = 10 \cdot 2 + 1 = 20 + 1 = 21$

$(Y)_{10} := (Y)_{10} \cdot 2 + 1 = 21 \cdot 2 + 1 = 42 + 1 = 43$

$(Y)_{10} = 43$

Beispiel $(X)_8 = (1705)_8$, $(Y)_{10} = ?$

$(Y)_{10} := 0$

$(Y)_{10} := (Y)_{10} \cdot 8 + (x_i)_8 \quad$ für $i = \{3, 2, 1, 0\}$

$(Y)_{10} :=$

$(Y)_{10} :=$

$(Y)_{10} :=$

$(Y)_{10} :=$

$(Y)_{10} \;=$

- Umwandlung einer nicht negativen Dezimalzahl X in eine n-stellige Zahl Y zur Basis b:
  - gegeben $(X)_{10} \geq 0$, gesucht $(Y)_b = (y_{n-1}y_{n-2}...y_1y_0)_b$ mit $b \geq 2$
  - Lösung: iteratives Verfahren, in dem die Dezimalzahl X durch die Basis b dividiert wird, und der Rest aus der Division als die Ziffer an der i-ten Stelle notiert wird.

    solange $(X)_{10} > 0$ berechne für $i = \{0, 1, ... n\text{-}1\}$:

    1. Schritt: $(X)_{10} / b = (Z)_{10} + (r)_{10}$ => $(y_i)_b := (r)_{10}$
    2. Schritt: $(X)_{10} := (Z)_{10}$

    mit
  - $n := \lceil \log_b(X)_{10} \rceil$
  - / Divisionsoperator für ganzzahlige Division
  - $(Z)_{10}$ Ganzzahlquotient
  - $(r)_{10}$ Rest aus der ganzzahligen Division

Beispiel: $(X)_{10} = (366)_{10}$, $(Y)_2 = ?$

↓ Rest aus der Division

$$
\begin{aligned}
366 / 2 &= 183 + 0 &\Rightarrow (y_0)_2 = 0\\
183 / 2 &= 91 + 1 &\Rightarrow (y_1)_2 = 1\\
91 / 2 &= 45 + 1 &\Rightarrow (y_2)_2 = 1\\
45 / 2 &= 22 + 1 &\Rightarrow (y_3)_2 = 1\\
22 / 2 &= 11 + 0 &\Rightarrow (y_4)_2 = 0\\
11 / 2 &= 5 + 1 &\Rightarrow (y_5)_2 = 1\\
5 / 2 &= 2 + 1 &\Rightarrow (y_6)_2 = 1\\
2 / 2 &= 1 + 0 &\Rightarrow (y_7)_2 = 0\\
1 / 2 &= 0 + 1 &\Rightarrow (y_8)_2 = 1
\end{aligned}
$$

↑ Werte von unten nach oben auslesen

$$(Y)_2 = (y_8\ y_7\ y_6\ y_5\ y_4\ y_3\ y_2\ y_1\ y_0)_2 = (101101110)_2$$

Beispiel: $(X)_{10} = (5949)_{10}$, $(Y)_8 = ?$

- eine (n+m)-stellige nicht negative Zahl zur Basis b wird notiert:

$$(X)_b = (x_{n-1} x_{n-2} \ldots x_1 x_0 \, , x_{-1} \ldots x_{-m})_b$$

$$(X)_b = \underbrace{x_{n-1} \cdot b^{n-1} + \ldots + x_0 \cdot b^0}_{\substack{\text{n-stellige} \\ \text{Vorkommazahl } (V_X)_b}} + \underbrace{x_{-1} \cdot b^{-1} + \ldots + x_{-m} \cdot b^{-m}}_{\substack{\text{m-stellige} \\ \text{Nachkommazahl } (N_X)_b}}$$

$$(X)_b = (V_X, N_X)_b = (V_X)_b + (N_X)_b$$

  - b Basis des Stellenwertsystems, $b \in \{2, 3, \ldots, \}$
  - n Anzahl der Vorkommastellen
  - m Anzahl der Nachkommastellen
  - x_i Ziffer an der i-ten Position, $x_i \in \{0, 1, 2, \ldots, b\text{-}2, b\text{-}1\}$
  - x_{n-1} Ziffer an der höchstwertigen Stelle
  - x_{-m} Ziffer an der niedrigstwertigen Stelle

- Umwandlung einer (n+m)-stelligen nicht negativen Zahl X zur Basis b in eine Dezimalzahl Y mit der Direktmethode

  $(Y)_{10} = (V_X,N_X)_b = \Sigma(x_i)_b \cdot b^i \qquad$ für $-m \le i \le n-1$

  Beispiel: $(X)_2 = (1001{,}11)_2$, $(Y)_{10} = ?$

  $(Y)_{10} =$

  $(Y)_{10} =$

  $(Y)_{10} =$

  Beispiel: $(X)_5 = (31{,}413)_5$, $(Y)_{10} = ?$

  $(Y)_{10} =$

  $(Y)_{10} =$

  $(Y)_{10} =$

- Umwandlung einer (n+m)-stelligen nicht negativen Zahl X zur Basis b in eine Dezimalzahl Y nach dem Horner-Schema

$(Y)_{10} = (V_Y,N_Y)_{10} = (V_X,N_X)_b = \Sigma(x_i)_b \cdot b^i$ für $-m \le i \le n\text{-}1$

$= \Sigma(x_k)_b \cdot b^k + \Sigma(x_i)_b \cdot b^i$ für $0 \le k \le n\text{-}1$ und $-m \le i \le -1$

$(N_Y)_{10} = \Sigma(x_i)_b \cdot b^i = x_{-1} \cdot b^{-1} + \ldots + x_{-m+1} \cdot b^{-m+1} + x_{-m} \cdot b^{-m}$

$= b^{-1} \cdot (x_{-1} + x_{-2} \cdot b^{-1} + \ldots + x_{-m+1} \cdot b^{-m+2} + x_{-m} \cdot b^{-m+1})$

$= \ldots$

$= b^{-1} \cdot (x_{-1} + b^{-1} \cdot (x_{-2} + \ldots + b^{-1} \cdot (x_{-m+1} + x_{-m} \cdot b^{-1}) \ldots))$

ergibt

$(N_Y)_{10} := 0{,}0$

$(N_Y)_{10} := b^{-1} \cdot ((x_i)_b + (N_Y)_{10})$ für $i = \{-m, -m+1, \ldots, -2, -1\}$

Beispiel:

$(X)_2 = (1{,}1011)_2,\ (Y)_{10} = (V_Y{,}N_Y)_{10} = ?$

$(V_Y)_{10} := 0$

$(V_Y)_{10} := (V_Y)_{10} \cdot 2 + 30 = 0 \cdot 2 + 1 = 1$

$(N_Y)_{10} := 0{,}0$

$(N_Y)_{10} := 2^{-1} \cdot (1 + (N_Y)_{10}) = 0{,}5 \cdot (1 + 0{,}00) = 0{,}5$

$(N_Y)_{10} := 2^{-1} \cdot (1 + (N_Y)_{10}) = 0{,}5 \cdot (1 + 0{,}50) = 0{,}75$

$(N_Y)_{10} := 2^{-1} \cdot (0 + (N_Y)_{10}) = 0{,}5 \cdot (0 + 0{,}75) = 0{,}375$

$(N_Y)_{10} := 2^{-1} \cdot (1 + (N_Y)_{10}) = 0{,}5 \cdot (1 + 0{,}375) = 0{,}6875$

Beispiel:

$(X)_5 = (31{,}413)_5$, $(Y)_{10} = (V_Y,N_Y)_{10} = ?$

$(V_Y)_{10} := 0$

$(V_Y)_{10} :=$

$(V_Y)_{10} :=$

$(N_Y)_{10} := 0{,}0$

$(N_Y)_{10} :=$

$(N_Y)_{10} :=$

$(N_Y)_{10} :=$

- Umwandlung einer nicht negativen Dezimalzahl X in eine (n+m)-stelligen Zahl Y zur Basis b:
  - gegeben $(X)_{10} = (V_X, N_X)_{10} \geq 0$, und Anzahl der Nachkommastellen m
  - gesucht $(Y)_b = (y_{n-1} y_{n-2} \ldots y_1 y_0, y_{-1} y_{-2} \ldots y_{-m})_b$ mit $b \geq 2$
  - Lösung: Aufteilung von $(X)_{10}$ in $(V_X, N_X)_{10}$, Vorkommastellen $(V_X)_{10}$ werden mit der Division-mit-Rest-Methode bestimmt, Nachkommastellen $(N_X)_{10}$ werden mit Multiplikation-mit-Abschneiden-Methode berechnet:

    für i := {-1, -2, ..., -m-1, -m} berechne

    1. Schritt: $(N_X)_{10} := (N_X)_{10} \cdot b$
    2. Schritt: $(y_i)_b := \text{trunc}\,(\,(N_X)_{10}\,)$
    3. Schritt: $(N_X)_{10} := (N_X)_{10} - (y_i)_b$

    mit einem Restfehler $\varepsilon = (N_X)_{10} \cdot b^{-m}$

    Funktion trunc(x) schneidet Nachkommastellen von x ab, (sie rundet also x in Richtung Null)

Beispiel: $(X)_{10} = (251{,}79)_{10}$, $m=8$, $(Y)_2 = ?$

$(X)_{10} = (V_X, N_X)_{10}$ => $(V_X)_{10} = (251)_{10}$ => $(V_Y)_2 = (11111011)_2$

$(N_X)_{10} = (0{,}79)_{10}$

$0{,}79 \cdot 2 = 1{,}58 = 1 + 0{,}58$ => $(y_{-1})_2 = 1$

$0{,}58 \cdot 2 = 1{,}16 = 1 + 0{,}16$ => $(y_{-2})_2 = 1$

$0{,}16 \cdot 2 = 0{,}32 = 0 + 0{,}32$ => $(y_{-3})_2 = 0$

$0{,}32 \cdot 2 = 0{,}64 = 0 + 0{,}64$ => $(y_{-4})_2 = 0$

$0{,}64 \cdot 2 = 1{,}28 = 1 + 0{,}28$ => $(y_{-5})_2 = 1$

$0{,}28 \cdot 2 = 0{,}56 = 0 + 0{,}56$ => $(y_{-6})_2 = 0$

$0{,}56 \cdot 2 = 1{,}12 = 1 + 0{,}12$ => $(y_{-7})_2 = 1$

$0{,}12 \cdot 2 = 0{,}24 = 0 + 0{,}24$ => $(y_{-8})_2 = 0$

$(N_Y)_2 = (11001010)_2$ mit Restfehler $\varepsilon = 0{,}24 \cdot 2^{-8} = 0{,}0009357$

$(Y)_2 = (V_Y, N_Y)_2 = (11111011{,}11001010)_2$ mit $\varepsilon = 0{,}24 \cdot 2^{-8}$

Beispiel:

$(X)_{10} = (251{,}79)_{10}, \quad m=3, \quad (Y)_8 = ?$

$(X)_{10} = (V_X, N_X)_{10} \quad => \quad (V_X)_{10} = (251)_{10} \quad => \quad (V_Y)_8 = (373)_8$

$(N_X)_{10} = (0{,}79)_{10}$

- Im Stellenwertsystem erfolgen die Addition und die Subtraktion (n+m)-stelliger nicht negativer Zahlen positionsweise, und zwar beginnend mit der niedrigstwertigen Stelle (von rechts nach links).
  - **Übertrag**: bei der Addition zweier Ziffer an der i-ten Stelle kann ein Übertrag (Carry) auf die höherwertiege Stelle (i+1-te Stelle) auftreten.
    Der Übertrag von der höchstwertigen Stelle (n-1-ten Stelle) auf die n-te Stelle wird im Ergebnis als die neue höchstwertige Stelle mit der Ziffer 1 interpretiert.
  - **Borgen**: bei der Subtraktion zweier Ziffer an der i-ten Stelle kann ein Borgen (Borrow) von der höherwertigen Stelle (i+1-te Stelle) auftreten.

- Addition und Subtraktion zweier (n+m)-stelligen nicht negativen Zahlen $(X)_b$ und $(Y)_b$ zur Basis $b \geq 2$

  für $-m \leq i \leq n-1$ $\quad$ $i := \{-m, -m-1, \ldots, -2, -1, 0, 1, 2, \ldots, n-2, n-1\}$

	Bedingung	Übetrag	Ergebnis der Operation
Addition	wenn $x_i + y_i + c_i < b$	$c_{i+1} := 0$	$s_i := x_i + y_i + c_i$
	wenn $x_i + y_i + c_i \geq b$	$c_{i+1} := 1$	$s_i := x_i + y_i + c_i - b$
Subtraktion	wenn $x_i - y_i - c_i \geq 0$	$c_{i+1} := 0$	$s_i := x_i - y_i - c_i$
	wenn $x_i - y_i - c_i < 0$	$c_{i+1} := 1$	$s_i := x_i - y_i - c_i + b$

  mit der Annahme, daß $(Y)_b < (X)_b$

  - s_i Ergebnis der Addition/Subtraktion an der i-ten Stelle
  - c_{i+1} Übertrag von der i-ten auf die i+1-te Stelle bei der Addition (Borgen von der i+1-ten Stelle bei der Subtraktion)
  - c_{-m} := 0 (Initialisierungswert für Übertag/Borgen)

Beispiel:

gegeben $(X)_{10} = (6172{,}765)_{10}$ und $(Y)_{10} = (1253{,}174)_{10}$

gesucht $(Z)_{10} = (X)_{10} + (Y)_{10}$

Lösung: $b = 10$, $n = 4$, $m = 3$, $i := \{-3, -2, -1, 0, 1, 2, 3\}$, $c_{-3} := 0$

i	x_i	y_i	c_i	$x_i+y_i+c_i$	Bed.	c_{i+1}	Kor.	s_i
-3	5	4	0	9	< 10	0	–	9
-2	6	7	0					
-1	7	1						
0	2	3						
1	7	5						
2	1	2						
3	6	1						

$(Z)_{10} =$

Beispiel:

gegeben $(X)_{10} = (6172{,}736)_{10}$ und $(Y)_{10} = (1255{,}174)_{10}$

gesucht $(Z)_{10} = (X)_{10} - (Y)_{10}$

Lösung: $b = 10$, $n = 4$, $m = 3$, $i := \{-3, -2, -1, 0, 1, 2, 3\}$, $c_{-3} := 0$

i	x_i	y_i	c_i	$x_i - y_i - c_i$	Bed.	c_{i+1}	Kor.	s_i
-3	6	4	0	2	≥ 0	0	–	2
-2	3	7						
-1	7	1						
0	2	5						
1	7	5						
2	1	2						
3	6	1						

$(Z)_{10} =$

Carry		0
$(X)_2 =$		100110,0101
$(Y)_2 =$	+	001101,1101

Carry		0
$(X)_2 =$		110110,0101
$(Y)_2 =$	-	101101,1101

Carry		0
$(X)_8 =$		5321,41
$(Y)_8 =$	+	1071,67

Carry		0
$(X)_8 =$		5321,41
$(Y)_8 =$	-	1071,67

- In der Darstellung positiver und negativer n-stelliger Zahlen zur Basis b wird die höchstwertige Ziffer x_{n-1} als das Vorzeichen interpretiert:
  - $x_{n-1} = 0$ für positive Zahlen
  - $x_{n-1} = b-1$ für negative Zahlen (z.B. F für HEX, 7 für OCT, 1 für BIN)
- für positive Zahl gilt immer: $(X)_b = (0.x_{n-2}x_{n-3} \dots x_1x_0)_b$
- zur Darstellung negativer Zahlen gibt es drei Möglichkeiten:
  - Vorzeichen und Betrag
  - (b–1)–Komplement
  - b–Komplement
- in den Darstellungen „Vorzeichen und Betrag“ sowie (b–1)–Komplement hat die Zahl 0 jeweils zwei Darstellungen (sog. positive Null und negative Null).

- für Dualzahlen (d.h. zur Basis b=2) ergibt sich:
  - Bei Verwendung von Vorzeichen und Betrag wird zur Codierung einer negativen n-stelligen Zahl $(X)_2$ das Vorzeichenbit x_{n-1} invertiert.

    Beispiel: $(45)_{10} = (\mathbf{0}.101101)_2$ $-(45)_{10} = (\mathbf{1}.101101)_2$

  - Das (b–1)–Komplement wird **Einerkomplement** genannt; die Codierung von $-(X)_2$ erhält man, indem man alle Bitstellen einer positiven n-stelligen Zahl $(X)_2$ invertiert.

    Beispiel: $(45)_{10} = (\mathbf{0}.101101)_2$ $-(45)_{10} = (\mathbf{1}.010010)_2$

  - Das b-Komplement wird **Zweierkomplement** genannt; die Codierung von $-(X)_2$ erhält man, indem man zum Einerkomplement eine $(1)_{10} = (000...01)_2$ hinzuaddiert.

    Beispiel: $(45)_{10} = (\mathbf{0}.101101)_2$ $-(45)_{10} = (\mathbf{1}.010011)_2$

- **Einerkomplement**: der darstellbare Zahlenbereich für n-stellige Dualzahlen $(X)_2$: $-2^{n-1}+1 \le X \le 2^{n-1}-1$
- Beispiel: für n = 8
  - kleinste negative darstellbare Zahl: $(1.0000000)_2 = -127_{10}$
  - größte negative darstellbare Zahl: $(1.1111110)_2 = -1_{10}$
  - größte positive darstellbare Zahl: $(0.1111111)_2 = 127_{10}$
  - alle Zahlen $-127 \le X \le 127$ können dargestellt werden
- Vor- und Nachteile des Einerkomplements:
  - \+ der darstellbare Zahlenbereich ist symmetrisch zu 0
  - \+ sehr einfache Umwandlung von positiver zu negativer Zahl und umgekehrt durch Invertierung aller Bits ($0 \Leftrightarrow 1$)
  - – zwei Darstellungen für die Null: $(0.00...00)_2$ und $(1.11...11)_2$
  - – Addierwerke sind aufwendig, weil das Ergebnis beim Auftreten negativer Zahlen in manchen Fällen korrigiert werden muß.
    => Einerkomplement wird daher i.a. nicht verwendet.

- **Zweierkomplement**: der darstellbarere Zahlenbereich für n-stellige Dualzahlen $(X)_2$: $-2^{n-1} \leq X \leq 2^{n-1}-1$
- Beispiel: für n = 8
  - kleinste negative darstellbare Zahl: $(1.0000000)_2 = -128_{10}$
  - größte negative darstellbare Zahl: $(1.1111111)_2 = -1_{10}$
  - größte positive darstellbare Zahl: $(0.1111111)_2 = 127_{10}$
  - alle Zahlen $-128 \leq X \leq 127$ können dargestellt werden
- Vor- und Nachteile des Zweierkomplements:
  - \+ eindeutige Darstellung der Null als $(0.00...00)_2$
  - \+ einfache Realisierung der Addition auch beim Auftreten negativer Zahlen ohne zusätzlichen Aufwand
  - – darstellbarer Zahlenbereich ist asymmetrisch
  - – Umwandlung von positiver zu negativer Zahl und umgekehrt erfordert die Invertierung aller Bits sowie eine Addition von $(0.00..01)_2$

Beispiel:

$(X)_{10} = -(109)_{10} \Rightarrow (X)_2 = -(1101101)_2$ (7-stellige Zahl)

Erweiterung der Zahl um das Vorzeichen zu einer 8-stelligen Zahl

$(X)_2 = -(0.1101101)_2$

1-Komplement: $-(0.1101101)_2 \Rightarrow$

2-Komplement: 1-Komplement + $(0.00..01)_2$

$-(0.1101101)_2 \Rightarrow$

$(X)_{10} = -(317{,}75)_{10} \Rightarrow (X)_2 = -(100111101{,}11)_2$ (11-stellige Zahl)

Erweiterung der Zahl um das Vorzeichen zu einer 12-stelligen Zahl

$(X)_2 = -(0.100111101{,}11)_2$

1-Komplement: $-(0.100111101{,}11)_2 \Rightarrow$

2-Komplement: $-(0.100111101{,}11)_2 \Rightarrow$

- Addition und Subtraktion zweier (n+m)-stelligen Dualzahlen $(X)_2$ und $(Y)_2$ erfolgt positionsweise

für $-m \le i \le n-1$			Addition		Subtraktion	
c_i	x_i	y_i	s_i	c_{i+1}	s_i	c_{i+1}
0	0	0	0	0	0	0
0	0	1	1	0	1	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
1	0	0	1	0	1	1
1	0	1	0	1	0	1
1	1	0	0	1	0	0
1	1	1	1	1	1	1

- s_i Ergebnis der Addition/Subtraktion an der i-ten Stelle
- c_{i+1} Übertrag von der i-ten auf die i+1-te Stelle bei der Addition (Borgen von der i+1-ten Stelle bei der Subtraktion)

Carry	0	
$(X)_2 =$	0.0110111	$(55)_{10}$
$(Y)_2 =$	+ 0.0101010	$(42)_{10}$
		$(97)_{10}$

Carry	0	
$(X)_2 =$	0.0110111	$(55)_{10}$
$(Y)_2 =$	+ 1.1010110	$(-42)_{10}$
		$(13)_{10}$

Carry	0	
$(X)_2 =$	1.1001001	$(-55)_{10}$
$(Y)_2 =$	+ 1.1010110	$(-42)_{10}$
		$(-97)_{10}$

Carry	0	
$(X)_2 =$	0.0101010	$(42)_{10}$
$(Y)_2 =$	+ 1.1001001	$(-55)_{10}$
		$(-13)_{10}$

$$
\begin{array}{lrl}
\text{Carry} & 0 & \\
(X)_2 = & 0.0110111 & (55)_{10} \\
(Y)_2 = & -\ \ 0.0101010 & (42)_{10} \\
\hline
 & & (13)_{10}
\end{array}
$$

$$
\begin{array}{lrl}
\text{Carry} & 0 & \\
(X)_2 = & 0.0110111 & (55)_{10} \\
(Y)_2 = & -\ \ 1.1010110 & (-42)_{10} \\
\hline
 & & (97)_{10}
\end{array}
$$

$$
\begin{array}{lrl}
\text{Carry} & 0 & \\
(X)_2 = & 0.0101010 & (42)_{10} \\
(Y)_2 = & -\ \ 0.0110111 & (55)_{10} \\
\hline
 & & (-13)_{10}
\end{array}
$$

Carry	0	
$(X)_2$ =	1.1010110	$(-42)_{10}$
$(Y)_2$ =	- 1.1001001	$(-55)_{10}$
		$(13)_{10}$

Carry	0	
$(X)_2$ =	1.1001001	$(-55)_{10}$
$(Y)_2$ =	- 1.1010110	$(-42)_{10}$
		$(-13)_{10}$

Carry	0	
$(X)_2$ =	1.1010110	$(-42)_{10}$
$(Y)_2$ =	- 0.0110111	$(55)_{10}$
		$(-97)_{10}$

- Subtraktion zweier (n+m)-stelligen Zahlen zur Basis b kann durch eine Addition mit einem b-Komplement ersetzt werden mit dem Ansatz:
  $(X)_b - (Y)_b = (X)_b + (-Y)_b = (X)_b + (b^n - (Y)_b)$

- Paraktische Realisierung der Subtraktion zweier (n+m)-stelligen Dualzahlen $(X)_b - (Y)_b$
  1. Bildung des Einerkomplements von Y durch Invertierung aller Bitstellen
  2. Bildung des Zweierkomplements von Y durch Addition von $(0..01)_2$ (in der Praxis wird dieser Schritt aus Effizienzgründen durch das Setzen der ersten Übertragsposition c_0 bzw. c_{-m} auf 1 realisiert)
  3. Addition der beiden Dualzahlen
  4. Übertrag von c_{n-1} auf c_n wird ignoriert

$$
\begin{array}{lrl}
(X)_2 = & 0.0110111 & (55)_{10} \\
(Y)_2 = & -\;\; 0.0101010 & (42)_{10} \\
\hline
 & & (13)_{10}
\end{array}
\qquad
\begin{array}{lrl}
\text{Carry} & \mathbf{1} & \\
(X)_2 = & 0.0110111 & (55)_{10} \\
(Y)_2 = & +\;\; 1.1010101 & (-42)_{10} \\
\hline
 & & (13)_{10}
\end{array}
$$

$$
\begin{array}{lrl}
(X)_2 = & 1.1001001 & (-55)_{10} \\
(Y)_2 = & -\;\; 1.1010110 & (-42)_{10} \\
\hline
 & & (-13)_{10}
\end{array}
\qquad
\begin{array}{lrl}
\text{Carry} & & \\
(X)_2 = & 1.1001000 & (-55)_{10} \\
(Y)_2 = & +\;\; 0.0101010 & (42)_{10} \\
\hline
 & & (-13)_{10}
\end{array}
$$

Boolesche Algebra

- ist eine algebraische Struktur, definiert durch
  - eine zweielementige Wertemenge B = {0, 1} (sog. boolesche Konstanten), (alternative Schreibweise: { Low, High }, { False, True }, { falsch, wahr }, { offen, geschlossen })
  - einen einstelligen Operator { ’ } (NICHT-Operator)
  - zwei zweistellige Operatoren { · , + } (UND- und ODER-Operatoren)
  - mit folgender Rangfolge der Operatoren: NICHT vor UND vor ODER mit Klammern (...) kann die Rangfolge geändert werden
  - zehn Axiome { A1, ..., A10 }
  - neun Gesetze { G1, ..., G9 }
- oft ein Synonymbegriff für Schaltalgebra
- Die boolesche Algebra ist ein Formalismus, das uns ermöglicht, das Verhalten von Digitalschaltungen mit formalen Methoden zu beschreiben, zu analysieren und zu minimieren, ohne diese Schaltung technisch aufbauen zu müssen.

- Boolesche Variable
  - ein symbolischer Bezeichner (Name), bestehend aus Buchstaben und ggf. Ziffern, der nur Werte aus der Wertemenge B annimmt.

- Boolescher Term (Formel, Ausdruck)
  - eine Zeichenkette, bestehend aus booleschen Konstanten, booleschen Variablen sowie booleschen Operatoren und Funktionen.

- n-stellige boolesche Funktion $y = f(e_0, e_1, e_2, \ldots, e_{n-1})$
  - diskrete Zuordnung f zwischen einer (abhängigen) booleschen Variable y und (unabhängigen) booleschen Variablen $e_0, e_1, \ldots, e_{n-1}$ (Funktionsargumenten, Operanden, Eingangsvariablen).

Boolesche Algebra

- Funktionstabelle (Wahrheitstabelle)
  - geordnete, tabelarische Darstellung boolescher Funktionen
  - In einer Funktionstabelle sind die Ergebnisse der booleschen Funktionen für sämtliche Kombinationsmöglichkeiten der beteiligten Funktionsargumente aufgelistet.
  - Für n Funktionsargumente ergeben sich 2^n Zeilen

Funktions-argumente			Funktions-werte	
u	x	y	s	v
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

Funktionstabelle für zwei Funktionen
$s = f(x, y, u)$
$v = g(x, y, u)$
mit drei Argumenten x, y und u

- NICHT-Operator
  - Boolescher Komplement
  - ist definiert durch
    - eine Eingangsvariable, die nur die Werte 0 oder 1 annehmen kann,
    - eine Ausgangsvariable, die genau den entgegengesetzten Wert der Eingangsvariable annimmt.
    - Wahrheitstabelle des NICHT-Operators:

E	E’
0	1
1	0

  - Symbolik für die Notation: ’ (¯, ¬, ~, !, /, not, nicht)
    Beispiel: X’

Boolesche Algebra

- UND-Operator
  - Boolesches Produkt, Konjunktion
  - ist definiert durch
    - zwei Eingangsvariablen, die nur die Werte 0 oder 1 annehmen können,
    - eine Ausgangsvariable, die den Wert 1 **nur dann** annimmt, wenn **beide** Eingangsvariablen gleichzeitig den Wert 1 haben.
    - Wahrheitstabelle des UND-Operators

E1	E2	$E1 \cdot E2$
0	0	0
0	1	0
1	0	0
1	1	1

  - Symbolik für die Notation: $\cdot$ (&, *, $\wedge$, and, und)
    Beispiel: $X \cdot Y$

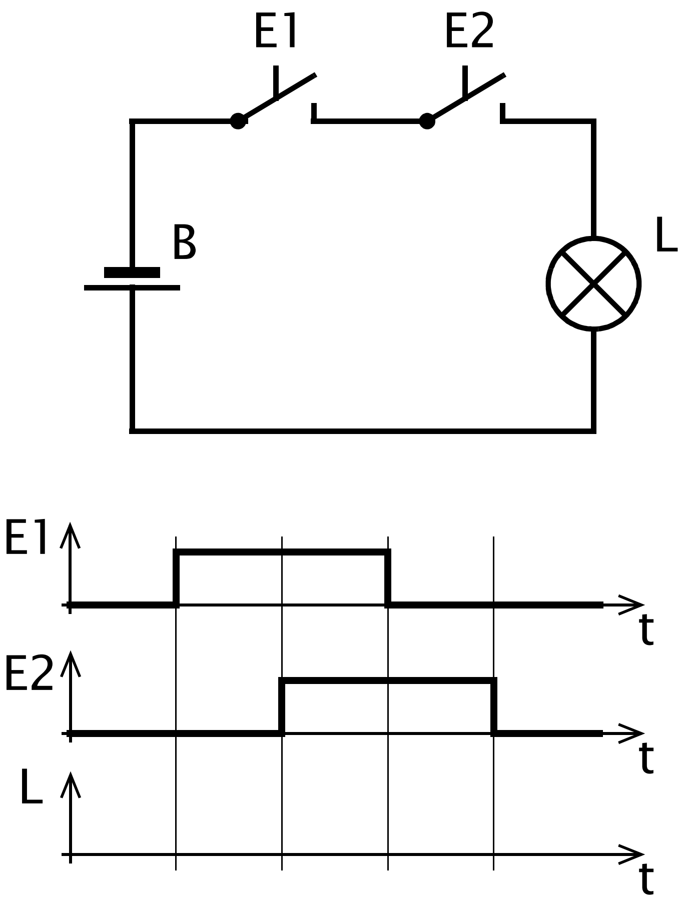
- ODER-Operator
  - Boolesche Summe, Disjunktion
  - ist definiert durch
    - zwei Eingangsvariablen, die nur die Werte 0 oder 1 annehmen können,
    - eine Ausgangsvariable, die den Wert 1 dann annimmt, wenn **mindestens eine** Eingangsvariable den Wert 1 hat.
    - Wahrheitstabelle des ODER-Operators:

E1	E2	E1+E2
0	0	0
0	1	1
1	0	1
1	1	1

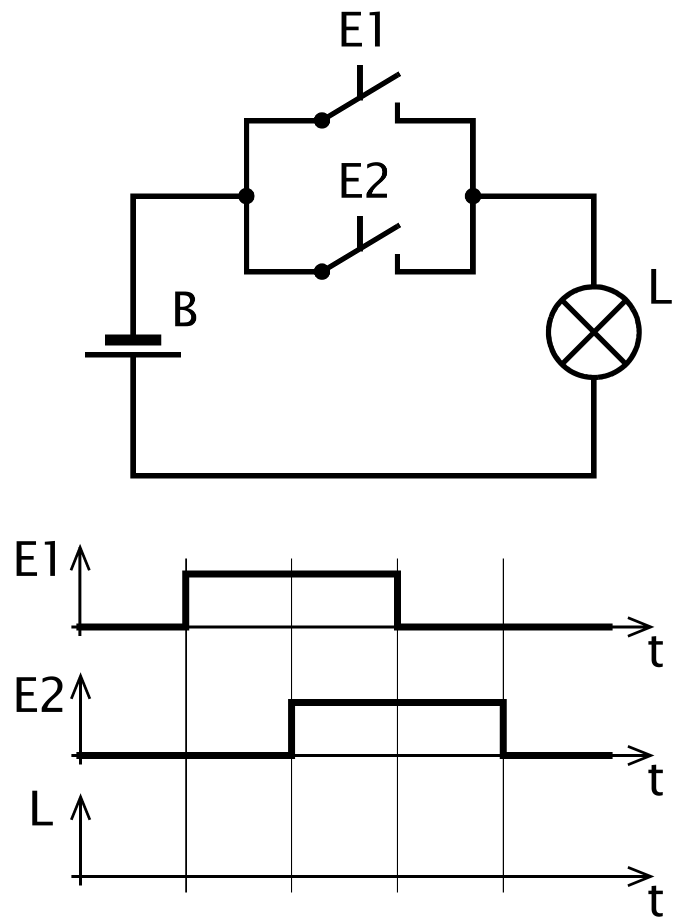
  - Symbolik für die Notation: + (|, $\vee$, or, oder)
    Beispiel: $X+Y$

- Technische Realisierungen mit Schaltern

UND-Operator



ODER-Operator



- Rekonstruktion einer Funktionstabelle als einer booleschen Funktion

$f(a, b, c) = a \cdot b + (b + c)'$

$f(0, 0, 0) = 0 \cdot 0 + (0 + 0)' = 0 + 0' = 0 + 1 = 1$

$f(0, 0, 1) = 0 \cdot 0 + (0 + 1)' = 0 + 1' = 0 + 0 = 0$

$f(0, 1, 0) =$

$f(0, 1, 1) =$

$f(1, 0, 0) =$

$f(1, 0, 1) =$

$f(1, 1, 0) =$

$f(1, 1, 1) =$

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

- Substitutions-/Ersetzungsregeln:
  - jede Variable in einem booleschen Ausdruck darf durch eine neue (d.h. in diesem Ausdruck noch nicht vorkommende) boolesche Variable ersetzt werden.
  - jeder (Teil-)Term eines booleschen Ausdrucks darf durch eine neue boolesche Variable ersetzt werden, um so den gesamten Ausdruck zu vereinfachen und letztendlich auf einen bekannten oder einfacher handhabbaren Ausdruck zurückzuführen.
  - jeder (Teil-)Term eines booleschen Ausdrucks darf durch einen anderen (äquivalenten) Ausdruck ersetzt werden, wenn er den gleichen Funktionswert wie der ersetzte Ausdruck hat, ohne den Funktionswert des gesamten booleschen Ausdruckes zu verändern.

- Kommutativitätsaxiom
  - Vertauschung der Reihenfolge der Variablen

    A1: $a + b = b + a$

    A2: $a \cdot b = b \cdot a$

- Assoziativitätsaxiom
  - Vertauschung der Reihenfolge gleichrangiger Operatoren

    A3: $a + b + c = (a + b) + c = a + (b + c)$

    A4: $a \cdot b \cdot c = (a \cdot b) \cdot c = a \cdot (b \cdot c)$

- Distributivitätsaxiom
  - Auflösung von Klammerausdrücken unter Berücksichtigung der Rangfolge von Operatoren

    A5: $a \cdot (b + c) = a \cdot b + a \cdot c$

    A6: $a + (b \cdot c) = (a + b) \cdot (a + c)$

- Identitätsaxiom
  - 0 ist ein neutrales Element für den ODER-Operator
    A7: $a + 0 = 0 + a = a$
  - 1 ist ein neutrales Element für den UND-Operator
    A8: $a \cdot 1 = 1 \cdot a = a$

- Komplementär-Axiom
  - ODER-Verknüpfung mit einem inversen Element ergibt 1
    A9: $a + a' = a' + a = 1$
  - UND-Verknüpfung mit einem inversen Element ergibt 0
    A10: $a \cdot a' = a' \cdot a = 0$

- Axiome lassen sich mit Hilfe von Wahrheitstabellen beweisen

  Beispiel: das Distributivität-Axiom A6 ist zu beweisen

  $a + (b \cdot c) = (a + b) \cdot (a + c)$

a	b	c	$x = b \cdot c$	$a + x$	$y = a + b$	$z = a + c$	$y \cdot z$
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

Boolesche Algebra

- Die unten stehenden Gleichung ist zu beweisen:

$$(a + b) \cdot (c + d) = a \cdot c + a \cdot d + b \cdot c + b \cdot d$$

- Idempotenzgesetze
  - mehrfach auftretende Variablen dürfen bis auf eine gestrichen werden. Die gleiche Variable darf beliebig oft in eine UND- oder ODER- Verknüpfung hinzugefügt werden.
  - G1: $a + a = a$
  - G2: $a \cdot a = a$

- Dominanzgesetze
  - 1 dominiert eine ODER-Verknüpfung, und 0 eine UND-Verknüpfung
  - G3: $a + 1 = 1 + a = 1$
  - G4: $a \cdot 0 = 0 \cdot a = 0$

- Absorptionsgesetze
  - G5: $a \cdot (a + b) = a$
  - G6: $a + (a \cdot b) = a$

- Doppelte-Negation-Gesetz
  - wird eine boolesche Variable oder ein boolesche Ausdruck zweifach negiert, so heben sich die Negierungen auf.
  - G7: $(a')' = a'' = a$
- erstes Gesetz von De Morgan (NOR-Operation)
  - negiert man eine ODER-Verknüpfung, so ist dies zu einer UND-Verknüpfung äquivalent, in der die einzelnen Variablen negiert sind.
  - G8: $(a + b)' = a' \cdot b'$
  - Negation einer Disjunktion ist die Konjunktion von Negationen
- zweites Gesetz von De Morgan (NAND-Operation)
  - negiert man eine UND-Verknüpfung, so ist dies zu einer ODER-Verknüpfung äquivalent, in der die einzelnen Variablen negiert sind.
  - G9: $(a \cdot b)' = a' + b'$
  - Negation einer Konjunktion ist die Disjunktion von Negationen

- Antivalenz-Operator
  - XOR-Operator: $a \oplus b = a' \cdot b + a \cdot b'$
  - ist definiert durch
    - zwei Eingangsvariablen, die nur die Werte 0 oder 1 annehmen können,
    - eine Ausgangsvariable, die den Wert 1 dann annimmt, wenn **genau eine** der Eingangsvariablen den Wert 1 hat.
    - Wahrheitstabelle des XOR-Operators:

E1	E2	$E1 \oplus E2$
0	0	0
0	1	1
1	0	1
1	1	0

  - Symbolik für die Notation: $\oplus$ ($\overline{\vee}$, $\neq$, xor)
    Beispiel: $X \oplus Y$

- Äquivalenz-Operatoren
  - XNOR-Operator: $(a \equiv b) = (a \oplus b)' = a' \cdot b' + a \cdot b$
  - ist definiert durch
    - zwei Eingangsvariablen, die nur die Werte 0 oder 1 annehmen können,
    - eine Ausgangsvariable, die den Wert 1 dann annimmt, wenn **beide** Eingangsvariablen den gleichen Wert haben, egal ob 0 oder 1.
    - Wahrheitstabelle des XNOR-Operators:

E1	E2	E1 $\equiv$ E2
0	0	1
0	1	0
1	0	0
1	1	1

  - Symbolik für die Notation: $\equiv$ ($\leftrightarrow$, =, xnor)
    Beispiel: $X \equiv Y$

- Beweisbeispiel:
  - das Absorptionsgesetz G5: $a \cdot (a + b) = a$ ist zu beweisen

a	b		
0	0		
0	1		
1	0		
1	1		

G5: $a \cdot (a + b) = a$

- Vereinfachung boolescher Terme
  - Für die algebraische Vereinfachung boolescher Terme gibt es kein allgemein gültiges Rezept, dafür aber eine Reihe heuristischer Regeln.

1. Regel: Ausdrücke ohne Klamern versucht man mit den Absorptionsgesetzen zu vereinfachen.

$$f(x, y, z) = x \cdot y + x \cdot z + y \cdot z + y + z$$

2. Regel: Beim Auftreten von komplexen Klammerausdrücken versucht man zunächst die Klammerinhalte zu vereinfachen.

$$f(x, y) = (x \cdot y + x' \cdot y + x' \cdot y')'$$

3. Regel: Geschachtelte Klammerausdrücke werden im allgemeinen von innen nach außen verarbeitet

$$f(a, b, c) = [(a + b) \cdot (a + b') + a' \cdot b']$$

4. Regel: Beim Auftreten von einfachen negierten Termen versucht man, diese zuerst mit den De Morgan-Gesetzen aufzulösen.

$$f(a, b, c) = (a + b')\cdot(a'\cdot b'\cdot c)'$$

5. Regel: Durch den Einsatz komplexer Operatoren XOR und XNOR lassen sich Termen stark vereinfachen.

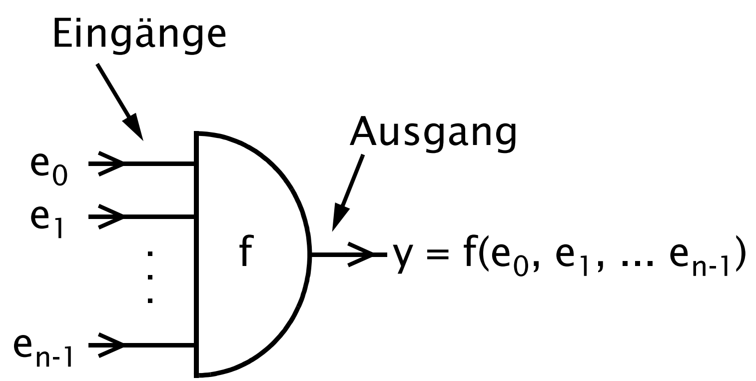
$$f(a, b, c) = a' \cdot b' + a \cdot b' + a' \cdot b \cdot c' + a \cdot b \cdot c$$

6. Regel: Bei einigen Termen führt erst eine geeignete „Expansion“ mit dem Faktor 1=a+a’ zur Anwendung des Absorptionsgesetzes und damit zur Vereinfachung

$$T(p, q, r) = p'\cdot q + p\cdot r' + q\cdot r'$$

Technische Realisierung boolescher Funktionen

- Eine boolesche Funktion kann in eine digitale Schaltung umgeformt werden, so daß die Schaltung den Wert der booleschen Funktion berechnet.
- Das Grundelement einer digitalen Schaltung ist ein (Logik-)Gatter.
- Ein Gatter ist
  - ein elektronisches, aus Transistoren aufgebautes und i.d.R. nicht weiter zerlegbares Bauelement,
  - ausgestattet mit einem, zwei oder mehreren Eingängen $e_0, e_1, \ldots, e_{n-1} \in \{0,1\}$ und genau einem Ausgang $y \in \{0,1\}$,
  - zur Realisierung einer booleschen Funktion $y = f(e_0, e_1, \ldots, e_{n-1})$.
  - Die Funktion f eines Gatters wird durch eine Funktionstabelle festgelegt.

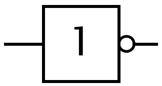
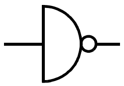
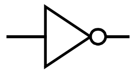
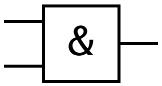
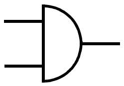
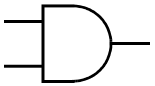
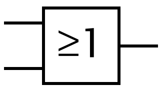
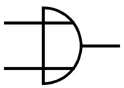
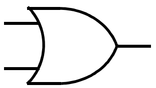
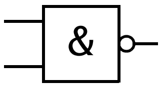
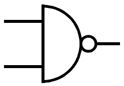
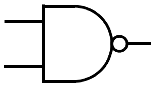
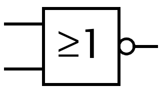
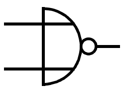
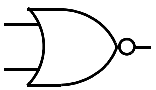
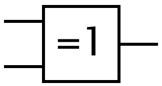
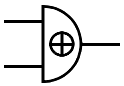
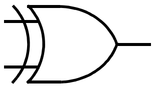
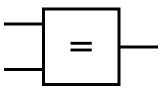
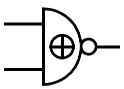
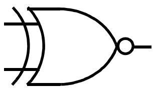


- Eine digitale Schaltung wird aus baumartig verketteten Gattern aufgebaut.

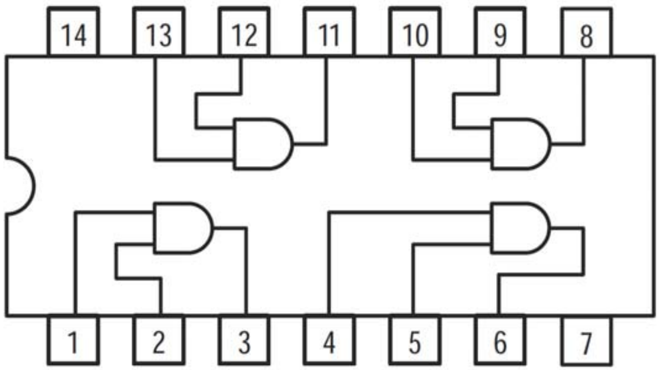
- Schaltnetz (kombinatorische Schaltung)
  - nach DIN IEC 748 Teil 2 ist ein Schaltnetz eine Digitalschaltung, in der es für jede mögliche Kombination von digitalen Signalen an den Eingängen eine – und nur eine – Kombination von digitalen Signalen an den Ausgängen gibt.

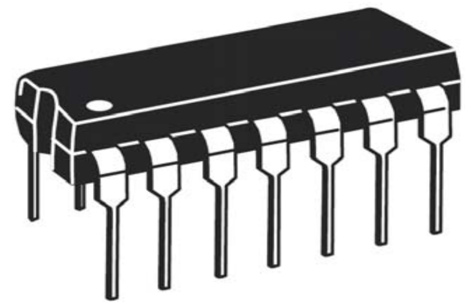
- Schaltnetz (alternative Definitionen)
  - ist eine schaltungstechnische Einheit zum Verarbeiten von Schaltvariablen, deren Wert an den Ausgängen zu einem Zeitpunkt nur von den Werten an den Eingängen zum gleichen Zeitpunkt abhängt.
  - eine Verknüpfung von logischen Grundelementen ohne interne Speicherung von Variablen

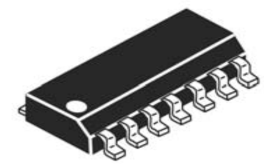
- Boolesche Verküpfungen und Schaltsymbole

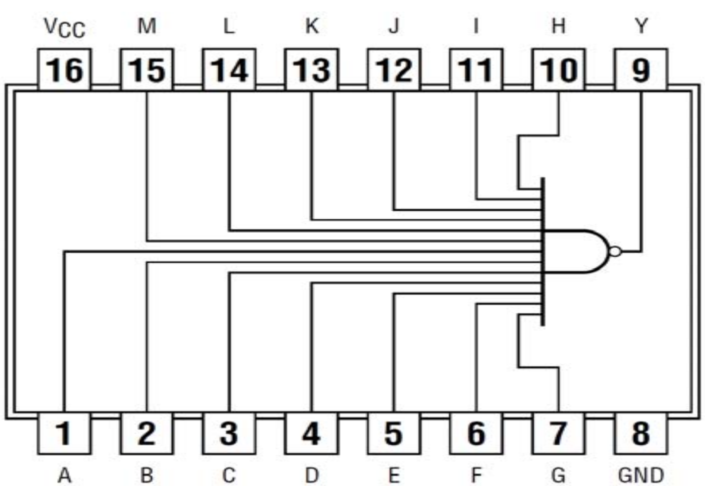
A	**0**	**1**	**0**	**1**			**Schaltsymbole/Gatter**		
B	**0**	**0**	**1**	**1**	**Verknüpfung**	**Funktion**	**IEC 60617**	**DIN 40700**	**ANSI 91-1984**
Y	1	0	–	–	Y = A’	NOT			
Y	0	0	0	1	$Y = A \cdot B$	AND			
Y	0	1	1	1	$Y = A + B$	OR			
Y	1	1	1	0	$Y = (A \cdot B)'$	NAND			
Y	1	0	0	0	$Y = (A + B)'$	NOR			
Y	0	1	1	0	$Y = A \oplus B$	XOR			
Y	1	0	0	1	$Y = (A \oplus B)'$	XNOR			

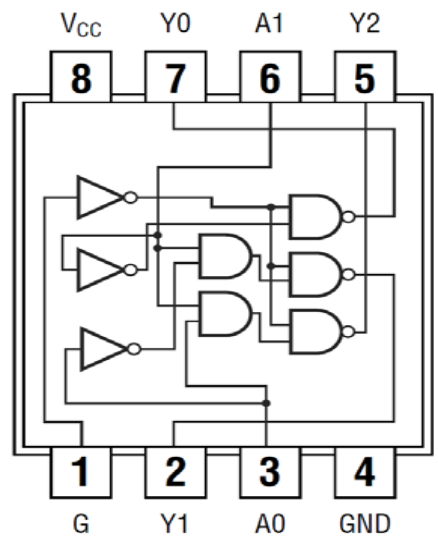
- Integrierte Bausteine











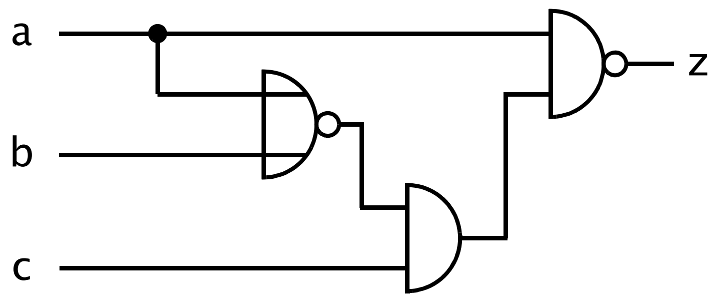
Quellen: Datenbücher von *ON Semiconductor* und *Texas Instruments Inc.*

- Darstellung boolescher Funktionen mit Schaltsymbolen

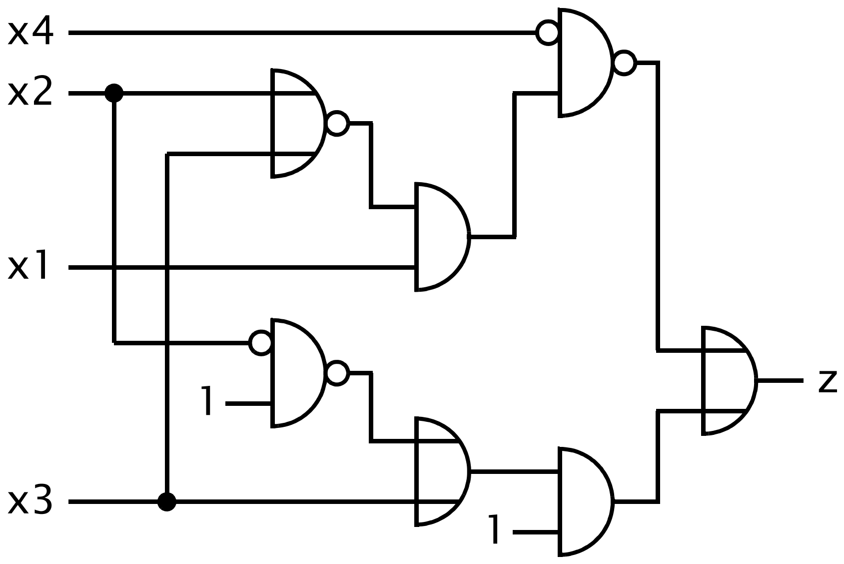
Boolesche Algebra		technische Realisierung
Gleichung	$\Leftrightarrow$	Schaltnetz
Variable	$\Leftrightarrow$	Signal(-leitung)
Operator	$\Leftrightarrow$	Gatter
Funktion	$\Leftrightarrow$	Funktionsblock

$$f = a \cdot b + c \cdot (a + b')$$

- Rekonstruktion boolescher Funktionen aus Schaltnetzen



- Rekonstruktion und Minimierung boolescher Funktionen aus Schaltnetzen



- Realisierung boolescher Funktionen mit NAND-Gattern
  1. Umwandlung einer booleschen Funktion mit Hilfe des Axioms A5 in einen disjunktiven Ausdruck:

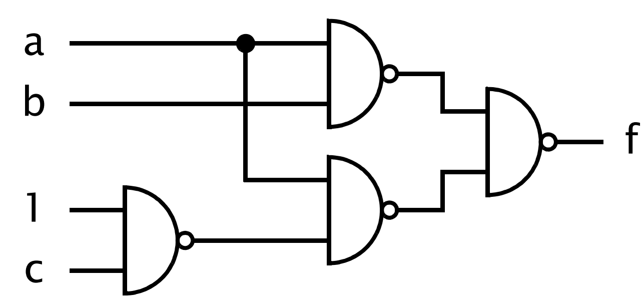
     $f(a, b, c) = a \cdot (b + c')$

  2. Eliminierung der ODER-Operatoren durch die Anwendung des Doppelte-Negation-Gesetzes G7 und anschließend des ersten Gesetzes von De Morgan G8:

     $f(a, b, c) = a \cdot b + a \cdot c'$

  3. Eliminierung negierter Variabalen durch die Anwendung des Identitätsaxioms A8:

     $f(a, b, c) = ((a \cdot b)' \cdot (a \cdot c')')'$



- Realisierung boolescher Funktionen mit NOR-Gattern
  1. Umwandlung einer booleschen Funktion mit Hilfe des Axioms A6 in einen konjunktiven Ausdruck:

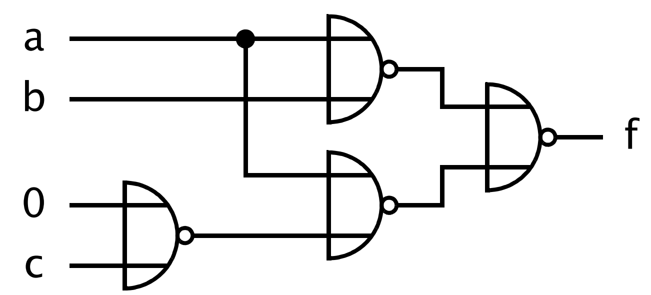
     $f(a, b, c) = a + b \cdot c'$

  2. Eliminierung der UND-Operatoren durch die Anwendung des Doppelte-Negation-Gesetzes G7 und anschließend des zweiten Gesetzes von De Morgan G9:

     $f(a, b, c) = (a + b) \cdot (a + c')$

  3. Eliminierung negierter Variabalen durch die Anwendung des Identitätsaxioms A7:

     $f(a, b, c) = ((a + b)' + (a + c')')'$

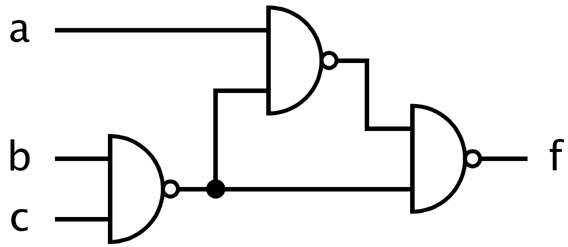


- Realisierung boolescher Funktionen mit NAND-Gattern
  - Beispiel: gesucht wird eine möglichst minimale Realisierung der booleschen Funktion $f(a,b,c) = a \cdot (b' + c') + b \cdot c$ mit NAND-Gattern
  - Lösung:

    $f(a,b,c) = a \cdot (b'+c') + b \cdot c$

    mit $t(b,c) = (b \cdot c)'$

    $f(a,b,c) = ((a \cdot t)' \cdot t)'$



- Äquivalente Darstellungen
  - verbale Beschreibungen
  - boolesche Funktionen
  - Funktions-/Wahrheitstabellen
  - Schaltpläne
  - Impulspläne
  - KV-Diagramme

- Normalformen boolescher Funktionen
  - Eine boolesche Funktion kann durch verschiedene aber äquivalente boolesche Ausdrücke beschrieben werden.
  - Solche Beschreibung ist also nicht eindeutig.
  - Eine Standarddarstellung boolescher Funktionen sind die konjunktive (KNF) und die disjunktive Normalform (DNF).

- Produktterm
  - Konjunktion von Variablen in negierter und nicht negierter Form
- Disjunktive Normalform (DNF)
  - Disjunktion (ODER-Verknüpfung) von Produkttermen
- Minterm
  - Produktterm, in dem jede Variable einer booleschen Funktion genau einmal vorkommt (einfach oder negiert)
- Kanonische Disjunktive Normalform (KDNF)
  - eindeutige Darstellung einer booleschen Funktion als Disjunktion von Mintermen
  - Jeder Minterm einer KDNF entspricht einer Zeile in der Funktion-stabelle, die den Funktionswert 1 liefert.

- Summenterm (Konjuktionsterm)
  - Disjunktion von Variablen in negierter und nicht negierter Form
- Konjunktive Normalform (KNF)
  - Konjunktion (UND-Verknüpfung) von Summentermen
- Maxterm
  - Summenterm, in dem jede Variable einer booleschen Funktion genau einmal vorkommt (einfach oder negiert)
- Kanonische Konjunktive Normalform (KKNF)
  - eindeutige Darstellung einer booleschen Funktion als Konjunktion von Maxtermen
  - Jeder Maxterm einer KKNF entspricht einer Zeile in der Funktionstabelle, die den Funktionswert 0 liefert.

- Kanonische Disjunktive und Konjunktive Normalformen
  - sind eindeutige Darstellungen, d.h. jede boolesche Funktion hat genau eine KDNF und eine KKNF!

Nr	x	y	z	s	Minterme	Maxterme
0	0	0	0	0		$x + y + z$
1	0	0	1	1	$x' \cdot y' \cdot z$	
2	0	1	0	1	$x' \cdot y \cdot z'$	
3	0	1	1	0		$x + y' + z'$
4	1	0	0	1	$x \cdot y' \cdot z'$	
5	1	0	1	0		$x' + y + z'$
6	1	1	0	0		$x' + y' + z$
7	1	1	1	1	$x \cdot y \cdot z$	

KDNF: $s(x, y, z) =$

KKNF: $s(x, y, z) =$

- Kanonische Disjunktive und Konjunktive Normalformen
  - kompakte Darstellung in der binären oder dezimalen Codierung (Hinweis: auf die Reihenfolge der Variablen in Termen achten)

$$f(a, b, c) = \underset{1\,0\,1}{a \cdot b' \cdot c} + \underset{1\,1\,0}{a \cdot b \cdot c'} + \underset{0\,1\,0}{a' \cdot b \cdot c'} + \underset{0\,0\,1}{a' \cdot b' \cdot c}$$

$f(a, b, c) = \Sigma((101)_2, (110)_2, (010)_2, (001)_2)$

$f(a, b, c) = \Sigma(5, 6, 2, 1)$ hier gilt $f(a, b, c) = 1$

$$g(a, b, c) = \underset{(1\quad 1\quad 0)}{(a' + b' + c)} \cdot \underset{(0\quad 1\quad 1)}{(a + b' + c')} \cdot \underset{(0\quad 0\quad 1)}{(a + b + c')}$$

$g(a, b, c) = \Pi((110)_2, (011)_2, (001)_2)$

$g(a, b, c) = \Pi(6, 3, 1)$ hier gilt $g(a, b, c) = 0!$

- Umwandlung von DNF in KDNF

  gesucht wird die KDNF für die Funktion $f(x, y, z) = x' \cdot (y \cdot z')'$

  1. Schritt: negierte Ausdrücke beseitigen (G8, G9)
     alle Klammer beseitigen (A5)

     $f = x' \cdot (y \cdot z')'$

  2. Schritt: Produktterme auf Minterme expandieren (A8, A9, A5)

     $f = x' \cdot y' + x' \cdot z$

  3. Schritt: mehrfach vorkommende Minterme streichen (G1)

     $f = x' \cdot y' \cdot z + x' \cdot y' \cdot z' + x' \cdot y \cdot z + x' \cdot y' \cdot z$

     $f(x, y, z) = \Sigma(0, 1, 3)$

- Umwandlung von KNF in KKNF

  gesucht wird die KKNF für die Funktion $f(x, y, z) = (x + y)' + z$

  1. Schritt: negierte Ausdrücke beseitigen (G8, G9), Klammer einführen (A6)

     $f = (x + y)' + z$

  2. Schritt: Summenterme auf Maxterme expandieren (A7, A10, A6)

     $f = (x' + z) \cdot (y' + z)$

  3. Schritt: mehrfach vorkommende Maxterme streichen (G2)

     $f = (x' + y + z) \cdot (x' + y' + z) \cdot (x + y' + z) \cdot (x' + y' + z)$

     $f(x, y, z) = \Pi(4, 6, 2)$

Beispiel: gesucht wird die KKNF

$f(a, b, c) = a \oplus (b + c)'$

Beispiel: gesucht wird die KDNF

$f(a, b, c) = a \oplus (b + c)'$

- Jede boolesche Funktion kann in *kanonischer disjunktiver* und *konjunktiver Normalform* dargestellt werden.
- Beide Darstellungen sind äquivalent und ineinander überführbar. (KDNF <-> KKNF)
- In der Praxis ist die (K)DNF übersichlicher als die (K)KNF und hinsichtlich der Vereinfachung von booleschen Ausdrücken leichter zu handhaben.
- Anwendungen von KDNF und KKNF:
  - Schaltnetzsynthese und -analyse
  - Schaltungen auf Basis von LUT (Look-Up-Table)
  - Zweistufige Schaltnetze mit PLAs, PALs
  - Schaltnetze mit Multiplexern

- Anwendung der booleschen Algebra:
  - Expansion von Normalformen auf Kanonische Formen DNF/KNF $\rightarrow$ KDNF/KKNF
  - Umwandlung von (K)DNF in (K)KNF und umgekehrt
  - Umsetzung von DNF/KNF auf Realisierung mit NANDs/NORs
  - in der Schaltungsanalyse bei der Rekonstruktion einer Funktion aus einer Schaltung
  - algebraische Minimierung boolescher Funktionen

- Minimierung boolescher Funktionen:
  - algorithmisch schwer beschreibbar
  - heuristik-basierende Vereinfachung
  - manchmal suboptimale Ergebnisse
  - große Erfahrung erforderlich