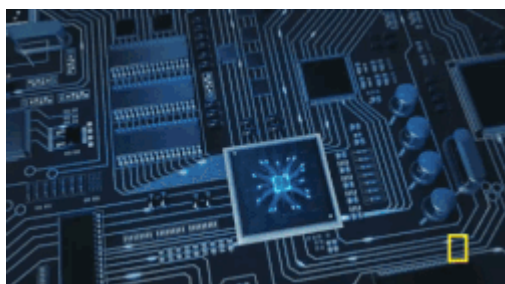


# Minimierung von Boolesche Funktionen

Parameter	Kursinformationen
Veranstaltung:	Digitale Systeme / Eingebettete Systeme
Semester:	Wintersemester 2025/26
Hochschule:	Technische Universität Freiberg
Inhalte:	Verfahren zur Minimierung Boolescher Funktionen, Karnaugh-Veitch-Diagramme
Link auf GitHub:	<a href="https://github.com/TUBAF-lfl-LiaScript/VL_EingebetteteSysteme/blob/master/03_Minimierung.md">https://github.com/TUBAF-lfl-LiaScript/VL_EingebetteteSysteme/blob/master/03_Minimierung.md</a>
Autoren:	Sebastian Zug & André Dietrich & Fabian Bär

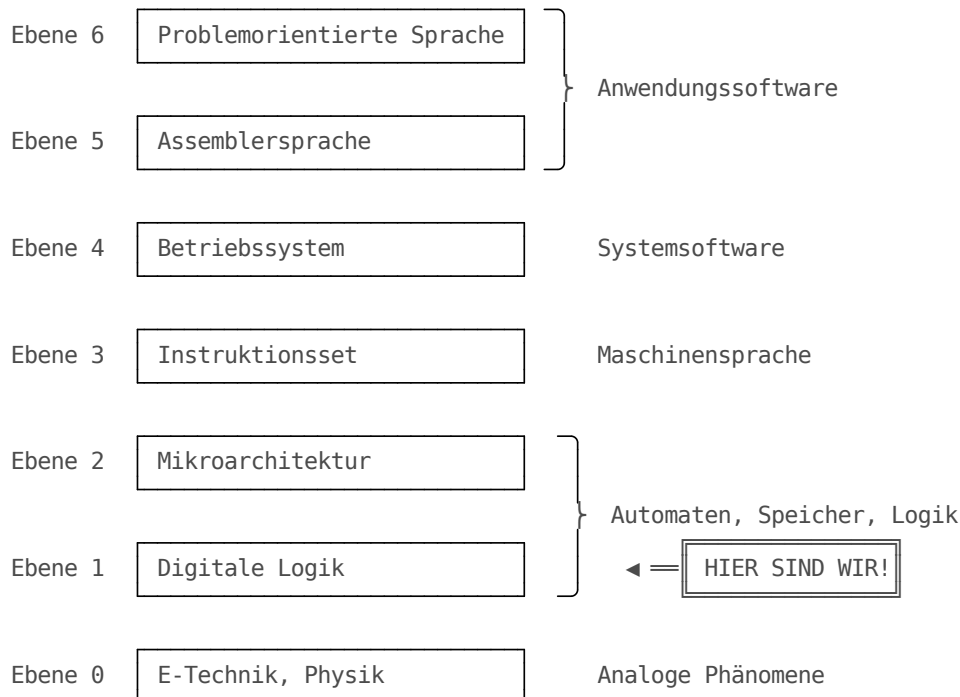


---

## Fragen an die Veranstaltung

- Nennen Sie 3 Beispiele, wie eine Schaltfunktion technisch umgesetzt werden kann.
  - Welcher Unterschied besteht zwischen der DNF und der KDNF?
  - Welcher Unterschied besteht zwischen der DNF und der KNF?
  - Geben Sie das De-Morgansche Gesetz wieder.
  - Welchem Grundkonzept folgt das Diagramm von Karnaugh-Veitch?
  - Welche Areale können im Diagramm zusammengefasst werden?
  - Worin unterscheidet sich das Vorgehen mit dem Karnaugh-Veitch-Diagramm für Min- und Maxterme?
-

### Abstraktionsebenen



## Reflexion Ihrer Fragen / Rückmeldungen

Format	Informatik Studierende	Nicht-Informatik Studierende
Verbesserungsvorschlag	0	2
Fragen	1	0
generelle Hinweise	0	1

Was hatten wir in der vergangenen Woche gelernt?

## Ausgangspunkt

In der letzten Vorlesung haben Sie die Boolesche Algebra kennengelernt und gesehen, wie man Schaltfunktionen durch algebraische Umformungen vereinfachen kann. Erinnern Sie sich?

### Rückblick: Algebraische Minimierung (Vorlesung 02)

Wir haben diese komplexe Funktion schrittweise vereinfacht:

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

... 6 Umformungsschritte später ...

$$= x_1 \cdot x_2 + x_1 \cdot x_3$$

Das war erfolgreich, aber mühsam! Sie mussten die richtigen Gesetze in der richtigen Reihenfolge anwenden. Und bei komplexeren Funktionen wird es schnell unübersichtlich.

#### Die Herausforderungen der algebraischen Methode:

- ✗ Keine systematische Vorgehensweise
- ✗ Fehleranfällig bei vielen Variablen
- ✗ Schwer zu erkennen, ob das Ergebnis minimal ist
- ✗ Zeitaufwändig bei 4+ Variablen

Genau hier setzen die heutigen Verfahren an! Wir lernen systematische Methoden kennen, die immer zum minimalen Ergebnis führen.

#### Heute: Systematische Minimierungsverfahren

**Das Ziel bleibt gleich:** Weniger Gatter = Geringere Kosten + Höhere Geschwindigkeit + Niedrigerer Energieverbrauch

**Der Weg ändert sich:** Von trial-and-error zu garantiert minimalem Ergebnis!

#### Drei Ansätze, die wir kennenlernen:

1. **Graphisch:** Karnaugh-Veitch-Diagramme (2-4 Variablen)
2. **Algorithmisch:** Quine-McCluskey-Verfahren (beliebig viele Variablen)
3. **Software-gestützt:** Moderne Tools für komplexe Schaltungen

## Herausforderung

Beginnen wir mit einem konkreten Beispiel, das die Grenzen der algebraischen Methode deutlich macht.

**Startpunkt für heute:** Können wir beweisen, dass zwei unterschiedlich aussehende Funktionen äquivalent sind?

**Die Aufgabe:** Beweisen Sie, dass diese beiden Funktionen äquivalent sind!

$$f(x_1, x_2, x_3, x_4) = x_3\bar{x}_1 + x_4\bar{x}_3\bar{x}_2 + \bar{x}_4x_3x_2\bar{x}_1 + x_4x_3\bar{x}_2x_1$$

$$g(x_1, x_2, x_3, x_4) = x_3\bar{x}_1 + x_4\bar{x}_2$$

Versuchen wir es zunächst mit algebraischen Umformungen, wie in der letzten Vorlesung. Mmmmh, scheinbar keine weitere Vereinfachung möglich. Sollten die beiden Gleichungen nicht äquivalent sein? Worin liegt unser Denkfehler?

### Versuch 1: Algebraische Vereinfachung


$$\begin{aligned}
 f(x_1, x_2, x_3, x_4) &= x_3\bar{x}_1 + x_4\bar{x}_3\bar{x}_2 + \bar{x}_4x_3x_2\bar{x}_1 + x_4x_3\bar{x}_2x_1 \\
 &= x_3\bar{x}_1 + \bar{x}_4x_3x_2\bar{x}_1 + x_4x_3\bar{x}_2x_1 + x_4\bar{x}_3\bar{x}_2 && \text{(Kommutativgesetz)} \\
 &= x_3\bar{x}_1 + x_4x_3\bar{x}_2x_1 + x_4\bar{x}_3\bar{x}_2 && \text{(Absorptionsgesetz)} \\
 &= \dots \text{ und jetzt?} && \text{✗ Festgefahren!}
 \end{aligned}$$

**✗ Problem erkannt:** Die bisherigen algebraischen Methoden führen uns hier in eine Sackgasse!  
**Aber:** Die Funktionen SIND tatsächlich äquivalent - wir müssen es nur beweisen können!

Alternativ können wir die Äquivalenz über Wahrheitstabellen überprüfen. Das funktioniert zwar nur bis zu einer gewissen Größe, ist aber systematisch! Die Wahrheitstabelle zeigt: Die rot markierten Zeilen werden von unserem minimierten Term  $x_4\bar{x}_2$  abgedeckt! Aber das manuelle Erstellen von Wahrheitstabellen ist mühsam.

### Versuch 2: Nachweis über Wahrheitstabelle

$x_4$	$x_3$	$x_2$	$x_1$	$f$	
0	0	0	0	0	
0	0	0	1	0	
0	0	1	0	0	
0	0	1	1	0	
0	1	0	0	1	$x_3\bar{x}_1$
0	1	0	1	0	
0	1	1	0	1	$\bar{x}_4x_3x_2\bar{x}_1$ $, x_3\bar{x}_1$
0	1	1	1	0	
1	0	0	0	1	$x_4\bar{x}_3\bar{x}_2$
1	0	0	1	1	$x_4\bar{x}_3\bar{x}_2$
1	0	1	0	0	
1	0	1	1	0	
1	1	0	0	1	$x_3\bar{x}_1$
1	1	0	1	1	$x_4x_3\bar{x}_2x_1$
1	1	1	0	1	$x_3\bar{x}_1$
1	1	1	1	0	

 **Erkenntnis:** Die Wahrheitstabelle beweist die Äquivalenz! **Aber:** Bei 4 Variablen = 16 Zeilen, bei 5 Variablen = 32 Zeilen, bei 6 Variablen = 64 Zeilen... **Lösung:** Wir brauchen systematische graphische oder algorithmische Verfahren!

**Die dramatische Reduktion:**





Wir sind von einer komplexen Funktion:

- **9 UND-Verknüpfungen** (teilweise 4 Eingänge)
- **3 ODER-Verknüpfungen** (4 Eingänge)





zu einer minimalen Form gekommen:

- **2 UND-Verknüpfungen** (2 Eingänge)
- **1 ODER-Verknüpfung** (2 Eingänge)

**Algebraische Minimierung funktioniert, aber:**

-  Erfordert viel Erfahrung und Intuition
-  Fehleranfällig bei vielen Schritten
-  Keine Garantie, das Minimum zu finden
-  Nicht skalierbar für komplexe Schaltungen

**Wir brauchen systematische Verfahren, die:**

-  Immer zum minimalen Ergebnis führen
-  Mechanisch anwendbar sind
-  Auch für Anfänger funktionieren
-  Für beliebig viele Variablen skalieren

**Die Lösung: Drei komplementäre Ansätze**

Methode	Eignung	Vorteile	Nachteile
Karnaugh-Veitch-Diagramm	2-4 Variablen	Visuell, intuitiv, schnell	Nicht für >4 Variablen
Quine-McCluskey	Beliebig viele Variablen	Algorithmisch, garantiert minimal	Manuell aufwändig
Software-Tools	Komplexe Schaltungen	Automatisch, skaliert beliebig	Blackbox-Charakter

Beginnen wir mit dem visuell einprägsamsten Verfahren: dem Karnaugh-Veitch-Diagramm!

## Aufstellen der Normalform

Format	Bedeutung
Wahrheitstafel	eindeutig
allgemeine Boolesche Funktion	uneindeutig
Normalformdarstellung einer Booleschen Funktion	eindeutig
Schaltnetz	uneindeutig

Für die Darstellung der Normalform benötigen wir zunächst weitere Begriffsdefinitionen.

### Produktterm

- Konjunktion (UND-Verknüpfung) von Variablen (ggf. negiert)
- Beispiele:  $x \cdot y$ ,  $x \cdot \overline{y}$ ,  $x \cdot y \cdot \overline{z}$

### Summenterm

- Disjunktion (ODER-Verknüpfung) von Variablen (ggf. negiert)
- Beispiele:  $x + y$ ,  $x + \overline{y}$ ,  $x + y + z$

### Minterm

- Produktterm, in dem jede Variable einer booleschen Funktion genau einmal vorkommt (einfach oder negiert)
- Beispiel:  $x \cdot y \cdot \overline{z}$  ist ein Minterm der Funktion  $f(x, y, z)$

### Maxterm

- Summenterm, in dem jede Variable einer booleschen Funktion genau einmal vorkommt (einfach oder negiert)
- Beispiel:  $x + y + z$  ist ein Maxterm der Funktion  $f(x, y, z)$

Und nun in der Kombination ....

### Disjunktive Normalform (DNF, Summe von Produkt-Mintermen)

- Disjunktion von Produkttermen
- Beispiel:  $(x \cdot y) + (x \cdot y \cdot z)$

### Konjunktive Normalform

- Konjunktion von Summentermen
- Beispiel:  $w \cdot (x + y) \cdot (x + y + z)$

#### Kanonische Disjunktive Normalform (KDNF)

- eindeutige Darstellung einer booleschen Funktion  $f$  als Disjunktion von Mintermen
- Beispiel:  $(\bar{x} \cdot y \cdot z) + (x \cdot \bar{y} \cdot \bar{z}) + (\bar{x} \cdot y \cdot \bar{z})$  ist KDNF von  $f(x, y, z)$

#### Kanonische Konjunktive Normalform (KKNF)

- eindeutige Darstellung einer booleschen Funktion  $f$  als Konjunktion von Maxtermen
- Beispiel:  $(x + \bar{y}) \cdot (\bar{x} + y) \cdot (\bar{x} + \bar{y})$  ist KKNF von  $f(x, y)$

"KDNF = Disjunktion von Mintermen"

Jede Boolesche Funktion lässt sich als genau eine KDNF darstellen!

Bildung der KDNF für  $n$ -stellige Boolesche Funktion  $f$ :

- für jede Zeile der Wahrheitstabelle mit  $f(x_1, x_2, \dots, x_n) = 1$  wird ein Minterm aufgestellt
- hierin wird jede Variable  $x_i$  negiert, wenn in der entsprechenden Zeile der Wert der Variablen 0 ist

Am Beispiel der Übungsaufgabe vom Beginn der Vorlesung ergibt sich daraus:



$x_4$	$x_3$	$x_2$	$x_1$	$f$	Minterme
0	0	0	0	0	
0	0	0	1	0	
0	0	1	0	0	
0	0	1	1	0	
0	1	0	0	1	$\overline{x}_4 x_3 \overline{x}_2 \overline{x}_1$
0	1	0	1	0	
0	1	1	0	1	$\overline{x}_4 x_3 x_2 \overline{x}_1$
0	1	1	1	0	
1	0	0	0	1	$x_4 \overline{x}_3 \overline{x}_2 \overline{x}_1$
1	0	0	1	1	$x_4 \overline{x}_3 \overline{x}_2 x_1$
1	0	1	0	0	
1	0	1	1	0	
1	1	0	0	1	$x_4 x_3 \overline{x}_2 \overline{x}_1$
1	1	0	1	1	$x_4 x_3 \overline{x}_2 x_1$
1	1	1	0	1	$x_4 x_3 x_2 \overline{x}_1$
1	1	1	1	0	

Damit ergibt sich die Kanonisch Disjunktiven Normalform obiger Wahrheitstafel zu

$$\begin{aligned}
 f(x_1, x_2, x_3, x_4) = & \bar{x}_4 x_3 \bar{x}_2 \bar{x}_1 + \\
 & \bar{x}_4 x_3 x_2 \bar{x}_1 + \\
 & x_4 \bar{x}_3 \bar{x}_2 \bar{x}_1 + \\
 & x_4 \bar{x}_3 \bar{x}_2 x_1 + \\
 & x_4 x_3 \bar{x}_2 \bar{x}_1 + \\
 & x_4 x_3 \bar{x}_2 x_1 + \\
 & x_4 x_3 x_2 \bar{x}_1
 \end{aligned}$$

Vergleichen Sie die Darstellung mit dem Resultat der Einstiegsübung!

Die Darstellung als KDNF ist abgesehen von Reihenfolge eindeutig!

"KKNF = Konjunktion von Maxtermen"

Jede Boolesche Funktion lässt sich als genau eine KKNF darstellen!

Bildung der KKNF für n -stellige Boolesche Funktion  $f$ :

- für jede Zeile der Wahrheitstabelle mit  $f(x_1, x_2, \dots, x_n) = 0$  wird ein Maxterm aufgestellt
- hierin wird jede Variable  $x_i$  negiert, wenn in der entsprechenden Zeile der Wert der Variablen 1 ist

Am Beispiel der Übungsaufgabe vom Beginn der Vorlesung ergibt sich daraus:

$x_4$	$x_3$	$x_2$	$x_1$	$f$
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Damit ergibt sich die Kanonisch Konjunktive Normalform obiger Wahrheitstafel zu

$$\begin{aligned} f(x_1, x_2, x_3, x_4) = & (x_4 + x_3 + x_2 + x_1) \cdot \\ & (x_4 + x_3 + x_2 + \overline{x}_1) \cdot \\ & (x_4 + x_3 + \overline{x}_2 + x_1) \cdot \\ & (x_4 + x_3 + \overline{x}_2 + \overline{x}_1) \cdot \\ & (x_4 + \overline{x}_3 + x_2 + \overline{x}_1) \cdot \\ & (x_4 + \overline{x}_3 + \overline{x}_2 + \overline{x}_1) \cdot \\ & (\overline{x}_4 + x_3 + \overline{x}_2 + x_1) \cdot \\ & (\overline{x}_4 + x_3 + \overline{x}_2 + \overline{x}_1) \cdot \\ & (\overline{x}_4 + \overline{x}_3 + \overline{x}_2 + \overline{x}_1) \end{aligned}$$

Die Darstellung als KKNF ist abgesehen von Reihenfolge eindeutig!

Satz: Jede KDNF kann in eine KKNF umgewandelt, jede KKNF kann in eine KDNF umgewandelt werden.

Zwei Darstellungen Boolescher Funktionen sind äquivalent, wenn sie durch Umformungen gemäß den Sätzen der Booleschen Algebra auf die gleiche KDNF bzw. KKNF zurückgeführt werden können

Welche Form erweist sich in welcher Situation als günstiger?

Eine KDNF ist günstiger als eine KKNF, wenn nur für wenige Kombinationen der Eingabewerte  $f(x_1, x_2, \dots, x_n) = 1$  gilt. Umgekehrt, wenn nur für wenige Kombinationen der Eingabewerte  $f(x_1, x_2, \dots, x_n) = 0$  gilt, ist die KKNF vorzuziehen.

Für die **OR** Funktion bedeutet das (wenig überraschend):

$x$	$y$	$f$
0	0	0
0	1	1
1	0	1
1	1	1

$$\text{KDNF: } f = \bar{x} \cdot y + x \cdot \bar{y} + x \cdot y$$

$$\text{KKNF: } f = x + y$$

## Minimierung von Schaltfunktionen mit dem KV-Diagramm

- Wann heißt eine Darstellung minimal ?
  - kleinste Anzahl an Gattern ?
  - kleinste Anzahl an Verbindungsleitungen ?
  - kleinste Anzahl an Produkttermen bzw. Summentermen ?

Lösungsmöglichkeiten:

1. geeignetes Umformen mit Gesetzen der Booleschen Algebra
2. Einsatz eines graphischen Verfahrens (z.B. ein Karnaugh-Veith-Diagramm), nur möglich bei Schaltfunktionen mit wenigen Variablen
3. algorithmisches Minimieren (z.B. Verfahren nach Quine-Mc-Cluskey), geeignet auch für Schaltfunktionen mit vielen Variablen

Ausgangspunkt für die Minimierung von Booleschen Ausdrücken ist das neutrale Element einer Operation. Für eine **ODER** Verknüpfung bewirkt eine falsche Aussage und für eine **UND** Operation eine wahre Aussage keine Veränderung des Gesamtausdruckes.

Disjunktive Normalform	Konjunktive Normalform
$  \begin{aligned}  f(x, y) &= x \cdot y + x \cdot \bar{y} \\  &= x \cdot (y + \bar{y}) \\  &= x \cdot (1)  \end{aligned}  $	$  \begin{aligned}  f(x, y) &= (x + y) \cdot (x + \bar{y}) \\  &= x + (y \cdot \bar{y}) \\  &= x + (0)  \end{aligned}  $
Wenn sich in einer disjunktiven Normalform zwei Summanden nur in genau einer komplementären Variablen unterscheiden, so können beide Summanden durch ihren gemeinsamen Teil ersetzt werden!	Wenn sich in einer konjunktiven Normalform zwei Produkte nur in genau einer komplementären Variablen unterscheiden, so können beide Produkte durch ihren gemeinsamen Teil ersetzt werden!

## Karnaugh-Veitch-Diagramme - Konzept

Jede zweistellige Boolesche Funktion lässt sich mit den Kombinationen ihrer Variablen in einer tabellarischen Darstellung visualisieren:

$$A = \bar{x} \cdot \bar{y} + \bar{x} \cdot y + x \cdot \bar{y} + x \cdot y$$

	$\overline{x}$	$x$
$\overline{y}$	$\overline{x} \cdot \overline{y}$	$x \cdot \overline{y}$
$y$	$\overline{x} \cdot y$	$x \cdot y$

Mit dieser verschobenen Wahrheitstafel lässt sich der Fingerabdruck einer booleschen Funktion darstellen.

Beispiel 1:  $f = x \cdot \overline{y} + x \cdot y$

	$\overline{x}$	$x$
$\overline{y}$	0	1
$y$	0	1

Beispiel 2:  $f = x \cdot \overline{y} + \overline{x} \cdot y$

	$\overline{x}$	$x$
$\overline{y}$	0	1
$y$	1	0

Dieses Konzept lässt sich auf Funktionen mit beliebig vielen Variablen übertragen. Wichtig dabei ist, dass benachbarte Kombinationen von Variablen sich nur in **einer** Variable unterscheiden. (Siehe [Gray-Code](#))

Der Gray-Code ist ein stetiger Code, bei dem sich benachbarte Codewörter nur in einer einzigen binären Ziffer unterscheiden.

Dezimalzahl	Binärcode	Gray-Code
0	000	000
1	001	001
2	010	011
3	011	010

	$\bar{x}\bar{y}$	$\bar{x}y$	$xy$	$x\bar{y}$
$\bar{z}$	$\bar{x} \cdot \bar{y} \cdot \bar{z}$	$\bar{x} \cdot y \cdot \bar{z}$	$x \cdot y \cdot \bar{z}$	$x \cdot \bar{y} \cdot \bar{z}$
$z$	$\bar{x} \cdot \bar{y} \cdot z$	$\bar{x} \cdot y \cdot z$	$x \cdot y \cdot z$	$x \cdot \bar{y} \cdot z$

Jeweils nur ein Wechsel von einer Variable pro Zeilen-/Spaltenübergang! Karnaugh-Veitch-Diagramme werden dabei geometrisch als Torus interpretiert!

Vorgehen zur Minimierung der KDNF einer  $n$ -stelligen Funktion  $f$

1. wird für jeden Minterm mit  $f(x_1, \dots, x_n) = 1$  eine 1 im Diagramm eingetragen
2. wird für jeden Maxterm mit  $f(x_1, \dots, x_n) = 0$  eine 0 im Diagramm eingetragen;
3. werden möglichst wenige und möglichst große zusammenhängende Bereiche (Schleifen) aus Einsen markiert, wobei
  - jeder Bereich aus  $2^k$  Elementen (mit  $0 \leq k \leq n$ ) besteht;
  - alle Einsen überdeckt werden müssen;
4. werden die markierten Bereiche nach der Resolutionsregel zu Produkttermen zusammengefasst, die summiert werden.

	$\bar{w}\bar{x}$	$\bar{w}x$	$wx$	$w\bar{x}$
$\bar{y}\bar{z}$	0	1	0	0
$\bar{y}z$	0	1	0	0
$yz$	0	0	0	0
$y\bar{z}$	0	0	0	0

Welcher Ausdruck steht hinter dem markierten Bereich?

$$\begin{aligned}
 f &= \bar{w}x\bar{y}\bar{z} + \bar{w}x\bar{y}z \\
 &= \bar{w}x\bar{y}(\bar{z} + z) \\
 &= \bar{w}x\bar{y} \cdot 1 \\
 &= \bar{w}x\bar{y}
 \end{aligned}$$

Beispiel 1:  $\overline{w} \overline{y}$

	$\overline{w} \overline{x}$	$\overline{w} x$	$w x$	$w \overline{x}$
$\overline{y} \overline{z}$	1	1	0	0
$\overline{y} z$	1	1	0	0
$y z$	0	0	1	0
$y \overline{z}$	0	0	1	0

Beispiel 2:  $w x y$

	$\overline{w} \overline{x}$	$\overline{w} x$	$w x$	$w \overline{x}$
$\overline{y} \overline{z}$	1	1	0	0
$\overline{y} z$	1	1	0	0
$y z$	0	0	1	0
$y \overline{z}$	0	0	1	0

Wählen Sie immer die größtmöglichen Schleifen und vermeiden Sie "Einzelelemente".

Falsch -  $f = \overline{w} \overline{y} + \overline{w} x y z$

	$\overline{w} \overline{x}$	$\overline{w} x$	$w x$	$w \overline{x}$
$\overline{y} \overline{z}$	1	1	0	0
$\overline{y} z$	1	1	0	0
$y z$	0	1	0	0
$y \overline{z}$	0	0	0	0

Richtig -  $f = \overline{w} \overline{y} + \overline{w} x z$



	$\overline{w} \overline{x}$	$\overline{w} x$	$w x$	$w \overline{x}$
$\overline{y} \overline{z}$	1	1	0	0
$\overline{y} z$	1	1	0	0
$y z$	0	1	0	0
$y \overline{z}$	0	0	0	0

Die Uni Marburg bietet eine gute Online Visualisierung des KV-Diagrammes an:

<https://www.mathematik.uni-marburg.de/~thormae/lectures/ti1/code/karnaughmap/>

## Beispiele

	$\overline{w} \overline{x}$	$\overline{w} x$	$w x$	$w \overline{x}$
$\overline{z}$	0	1	0	0
$z$	0	1	0	0

$$f = \overline{w} x$$

	$\overline{w} \overline{x}$	$\overline{w} x$	$w x$	$w \overline{x}$
$\overline{z}$	0	0	0	0
$z$	1	1	1	1

$$f = z$$

	$\overline{w} \overline{x}$	$\overline{w}x$	$wx$	$w\overline{x}$
$\overline{y} \overline{z}$	0	1	1	0
$\overline{y}z$	0	0	0	0
$yz$	0	0	0	0
$y\overline{z}$	0	1	1	0

$$f = x\overline{z}$$

	$\overline{w} \overline{x}$	$\overline{w}x$	$wx$	$w\overline{x}$
$\overline{y} \overline{z}$	1	0	0	1
$\overline{y}z$	0	0	0	0
$yz$	0	0	0	0
$y\overline{z}$	1	0	0	1

$$f = \overline{x} \overline{z}$$

	$\overline{w} \overline{x}$	$\overline{w}x$	$wx$	$w\overline{x}$
$\overline{y} \overline{z}$	1	0	0	1
$\overline{y}z$	0	1	1	1
$yz$	0	0	0	1
$y\overline{z}$	1	0	0	1

Hausaufgabe ...

Und jetzt Sie!

A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

	$\overline{A}\overline{B}$	$\overline{A}B$	$AB$	$A\overline{B}$
$\overline{C}$	1	1	0	1
$C$	1	0	0	0

$$\overline{A}\overline{B} + \overline{A}\overline{C} + \overline{B}\overline{C}$$

### Optimierungskriterien

	$\overline{w}\overline{x}$	$\overline{w}x$	$wx$	$w\overline{x}$
$\overline{y}\overline{z}$	1	1	0	1
$\overline{y}z$	0	1	1	1
$yz$	0	1	1	0
$y\overline{z}$	0	0	0	0

Welche Möglichkeiten haben wir ausgehend von der ersten Gruppe?

Variante 1:  $xz + \overline{x} \overline{y} \overline{z} + \overline{w} x \overline{y} + w \overline{x} \overline{y}$  (10 Operationen)

	$\overline{w} \overline{x}$	$\overline{w} x$	$w x$	$w \overline{x}$
$\overline{y} \overline{z}$	<div>1</div>	<div>1</div>	0	<div>1</div>
$\overline{y} z$	0	<div>1</div>	1	<div>1</div>
$y z$	0	<div>1</div>	1	0
$y \overline{z}$	0	0	0	0

Variante 2:  $xz + \overline{x} \overline{y} \overline{z} + \overline{w} x \overline{y} + w \overline{y} z$  (10 Operationen)

	$\overline{w} \overline{x}$	$\overline{w} x$	$w x$	$w \overline{x}$
$\overline{y} \overline{z}$	<div>1</div>	<div>1</div>	0	<div>1</div>
$\overline{y} z$	0	<div>1</div>	<div>1</div>	<div>1</div>
$y z$	0	<div>1</div>	1	0
$y \overline{z}$	0	0	0	0

Variante 3:  $xz + \overline{w} \overline{y} \overline{z} + w \overline{x} \overline{y}$  (7 Operationen)

	$\overline{w} \overline{x}$	$\overline{w} x$	$w x$	$w \overline{x}$
$\overline{y} \overline{z}$	<div>1</div>	<div>1</div>	0	<div>1</div>
$\overline{y} z$	0	<div>1</div>	1	<div>1</div>
$y z$	0	<div>1</div>	1	0
$y \overline{z}$	0	0	0	0

Überlappung lohnt sich nur, wenn dadurch größere Schleifen entstehen und zum Beispiel eine isolierte 1 vermieden wird.

## Karnaugh-Veitch für Maxterme

Das Karnaugh-Veitch-Diagramm lässt sich analog für Produkte von Summen aufstellen. Im Unterschied zur Minterm-Variante werden hier aber die Nullen erfasst - diese entsprechen ja auch den resultierenden Maxtermen.

Nehmen wir folgende Wahrheitstafel an:

$w$	$x$	$y$	$z$	$A$
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

8 Maxterme vs. 8 Minterme

	$\overline{w} \overline{x}$	$\overline{w} x$	$w x$	$w \overline{x}$
$\overline{y} \overline{z}$	1	0	0	1
$\overline{y} z$	0	1	1	1
$y z$	0	1	1	0
$y \overline{z}$	1	0	0	0

Ablesen der KNF:

- Zusammenfassung der Nullwerte (Summentermen) -  $\overline{w} + \overline{x} + z, x + \overline{z}, w + \overline{x} + y$
- Invertieren der Eingangsvariablen -  $w + x + \overline{z}, \overline{x} + z, \overline{w} + x + \overline{y}$
- Aufstellen der KNF -  $A = (w + x + \overline{z}) \cdot (\overline{x} + z) \cdot (\overline{w} + x + \overline{y})$

Da dies ein wenig durch die *Brust ins Auge* gedacht ist, wird folgendes Vorgehen empfohlen, das die Negation der KDNF nutzt:

- Zusammenfassen der Nullwerte als negierte DNF -  $\overline{A}_{DNF} = \overline{w} \overline{x} z + x \overline{z} + w \overline{x} y$
- Anwendung des de-morganschen Gesetzes -  $A_{DNF} = \overline{\overline{w} \overline{x} z + x \overline{z} + w \overline{x} y} = (w + x + \overline{z})(\overline{x} + z)(\overline{w} + x + \overline{y})$

**Aufgabe:** Weisen Sie die Äquivalenz der Gleichungen für die KNF  $A_{KNF} = (w + x + \overline{z})(\overline{x} + z)(\overline{w} + x + \overline{y})$  und die DNF  $A_{DNF} = \overline{w} \overline{x} \overline{z} + x z + w \overline{x} y$  nach.

## Dont-Care-Einträge in der Wahrheitstafel

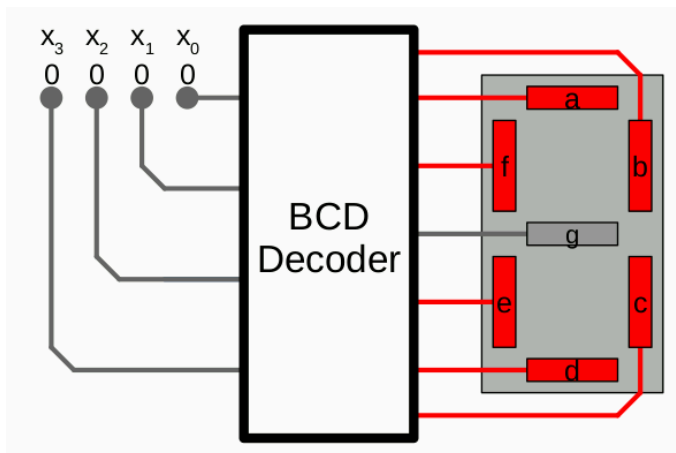
In einigen Fällen bildet die Wahrheitstafel Kombinationen der Eingangswerte ab, die für die Ausgabe gar nicht relevant sind. In diesem Fall spricht man von sogenannten **don't care** Ausgaben. Letztendlich ist uns das Funktionsergebnis für diese Fälle egal. Aus den **don't care** Fällen können entsprechend weitere Minimierungen hergeleitet werden.

Ein sehr anschauliches Anwendungsbeispiel dafür sind Sieben-Segment-Anzeigen, die aus 7 + 1 (Punkt) Leds bestehen und insbesondere zur Darstellung von Zahlenwerten genutzt werden.

<https://cdn-reichelt.de/documents/datenblatt/A500/SA52-11%23KIN.pdf>

Acht Eingänge pro Element würde die Zahl der verfügbaren Pins eines Controllers recht schnell aufbrauchen und ist auch nicht nötig. Wir wollen ja nur maximal die Zahlen von 0-9 darstellen. Diese können wir als 10 Zustände mit ... Leitungen abbilden

Folglich ergeben sich 6 Eingangskombinationen, die für unsere Ausgabe irrelevant sind.



Betrachten wir die Wahrheitstafel für das "d" Element.



$w$	$x$	$y$	$z$	$d$
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	D
1	0	1	1	D
1	1	0	0	D
1	1	0	1	D
1	1	1	0	D
1	1	1	1	D

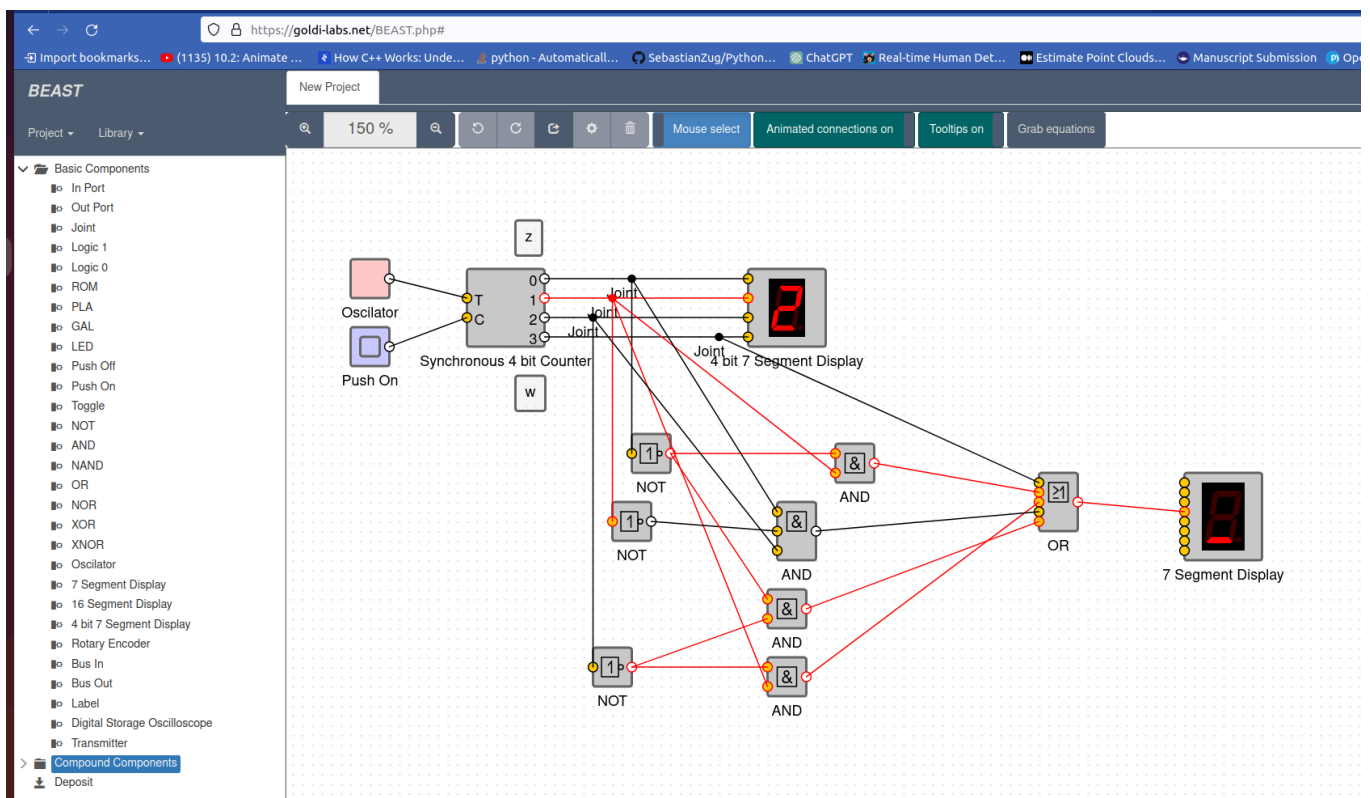
	$\overline{w} \overline{x}$	$\overline{w} x$	$w x$	$w \overline{x}$
$\overline{y} \overline{z}$	1	0	D	1
$\overline{y} z$	0	1	D	1
$y z$	1	0	D	D
$y \overline{z}$	1	1	D	D

Welche Gleichung für d lesen Sie daraus ab?

$$d = w + y\overline{z} + \overline{x}y + x\overline{y}z + \overline{x} \overline{z}$$

Die Korrektheit unserer Lösung können Sie in einer Simulationsumgebung evaluieren.

<https://goldi-labs.net/BEAST.php#>



Die gezeigte Lösung findet sich im Projektrepository.

## Zusammenfassung

Regeln zur Bildung der Schleifen:

- Fangen Sie mit isolierten Zellen an. Die entsprechenden Minterme können nicht mehr vereinfacht werden.
- Falls keine isolierten Zellen existieren, fange bei denen mit den wenigsten gleichwertigen Nachbarzellen an.
- Suche die Schleifen mit der größten Überdeckung von Zellen. Die Schleifen umfassen jeweils  $2^n$  mit ( $n = 0, 1, 2, \dots$ ) benachbarte Zellen. Starten Sie mit den kleinsten Schleifen.
- Überlappungen führen nur dann zu minimaleren Ausdrücken, wenn dadurch größere Schleifen gebildet werden können.
- Die minimale Funktion besteht aus der kleinsten Schleifenmenge, die alle individuell möglichst groß sind.

### Und wie geht es weiter?

Wir brauchen offensichtlich eine Möglichkeit, die Minimierung automatisiert durchzuführen. Hierzu gibt es verschiedene Methoden, die wir in den nächsten Kapiteln kennenlernen werden.

#### Minimize.py

```

1  from sympy.logic import SOPform
2  from sympy import symbols
3  x3, x2, x1, x0 = symbols('x3 x2 x1 x0')
4
5  minterms = [[0, 1, 1, 0],
6              [0, 1, 1, 1],
7              [1, 0, 0, 0],
8              [1, 0, 0, 1],
9              [1, 0, 1, 0],
10             [1, 0, 1, 1],
11             [1, 1, 0, 0],
12             [1, 1, 0, 1],
13             [1, 1, 1, 0]]
14 result = SOPform([x3, x2, x1, x0], minterms)
15 print("Minimized result:")
16 print(result)

```

```

Minimized result:
(x3 & ~x0) | (x3 & ~x1) | (x3 & ~x2) | (x1 & x2 & ~x3)
Minimized result:
(x3 & ~x0) | (x3 & ~x1) | (x3 & ~x2) | (x1 & x2 & ~x3)

```

Die Dokumentation zum Sympy Paket von Python finden Sie unter

<https://docs.sympy.org/latest/modules/logic.html> Sympy verwendet intern verschiedene Solver, darunter z.B. z3, dpll2 oder minisat22.

# Hausaufgaben

- Lösen Sie das Minimierungsproblem der Einstiegsaufgabe mit dem Karnaugh-Veitch-Diagramm.
- Stellen Sie die Wahrheitstafel für ein weiteres Element der Sieben-Segmentanzeige auf. Minimieren Sie den Ausdruck.
- Erstellen Sie mit dem Python-Beispiel eine eigene (willkürliche) Wahrheitstafel und vereinfachen Sie diese mit dem Karnaugh-Veitch-Diagramm und auf analytischem Wege.

Für das Training oder die Probe bei der Erledigung der Hausaufgaben können Sie die Werkzeuge der TU Ilmenau verwenden, die die besprochenen Inhalte wunderbar zusammenfasst.

<https://www.goldi-labs.de/SANE/view3>

The screenshot shows the SANE web application interface. The top navigation bar includes links for Truth Table, All Functions, Karnaugh Map, Normal Forms, Function Indices, QMC Algorithm, and Settings. The main content area is divided into two panels. The left panel, titled 'Karnaugh Map', displays a 4x4 grid for variables x0, x1, and x2. The right panel, titled 'Truth Table', shows a table with columns for input variables (x2, x1, x0), output variable (y0), and the resulting function value (Y).

#	x <sub>2</sub>	x <sub>1</sub>	x <sub>0</sub>	y <sub>0</sub>	H'	Y
0	0	0	0	0		0
1	0	0	1	0		0
2	0	1	0	1		1
3	0	1	1	0		0
4	1	0	0	0		0
5	1	0	1	0		0
6	1	1	0	0		0
7	1	1	1	1		1