Vereinfachung mit Hilfe von Karnaugh-Tafeln

Übungen Digitales Design

1 | KAR - Karnaugh-Tafel

1.1 Darstellung von Monomen

Stellen Sie die folgende Monome in einer Karnaugh-Tafel mit 4 Variablen dar.

$y_1 = \overline{b}a$ (1)

$y_2 = \overline{d}\overline{a}$ (2)

$y_3 = \overline{d}cb$ (3)

$y_4 = dba$ (4)

$y_5 = \overline{c}\overline{b}\overline{a}$ (5)

$y_6 = dcb\overline{a}$ (6)

kar/karnaugh-01

1.2 Monome

Geben Sie die algebraische Form der Monomen der folgenden Karnaugh-Tafeln.

y_1

	C		D
1	1	0	0
1	1	0	0
0	0	0	0
0	0	0	0

A

B

y_3

	C		D
0	1	0	0
0	1	0	0
0	0	0	0
0	0	0	0

A

B

y_2

	C		D
1	0	0	1
0	0	0	0
0	0	0	0
1	0	0	1

A

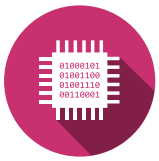
B

y_4

	C		D
0	1	0	0
0	0	0	0
0	0	0	0
0	1	0	0

A

B



kar/karnaugh-02

1.3 Darstellung von Polynomen

Stellen Sie die folgende Polynome in einer Karnaugh-Tafel mit 4 Variablen dar.

$$y_1 = \overline{b} + ac \quad (7)$$

$$y_2 = \overline{d} + \overline{a} + bc \quad (8)$$

$$y_3 = \overline{d}cb + d\overline{c}\overline{b} \quad (9)$$

$$y_4 = db + ab \quad (10)$$

$$y_5 = \overline{c}\overline{b}\overline{a} + \overline{c}\overline{b} \quad (11)$$

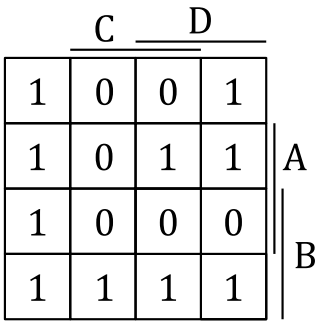
$$y_6 = dcb\overline{a} + \overline{d}cb\overline{a} \quad (12)$$

kar/karnaugh-03

2 KAR - Vereinfachung in der From von Produktsumme

2.1 Karnaugh-Tafel mit 4 Variablen

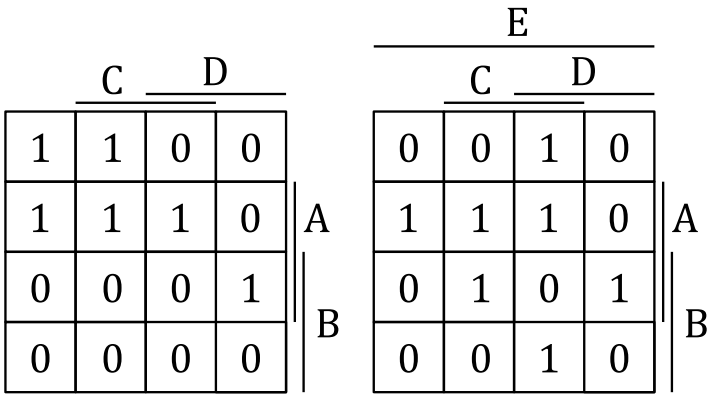
Bestimmen Sie die minimale Polynomialform der in der Karnaugh-Tafel der folgenden Abbildung dargestellten Funktion.



kar/productsum-01

2.2 Karnaugh-Tafel mit 5 Variablen

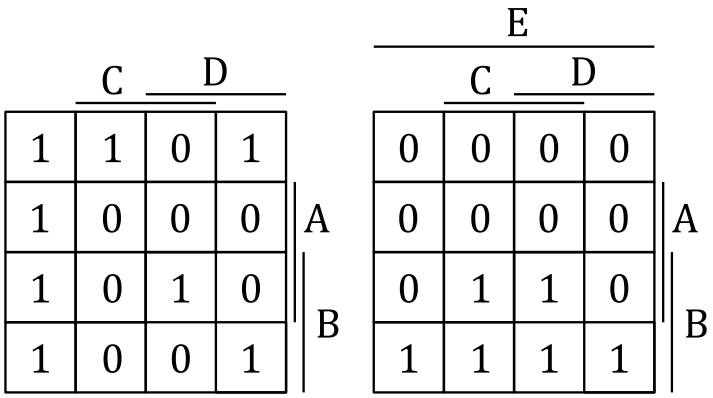
Bestimmen Sie die minimale Polynomialform der in der Karnaugh-Tafel der folgenden Abbildung dargestellten Funktion.



kar/productsum-02

2.3 Karnaugh-Tafel mit 5 Variablen

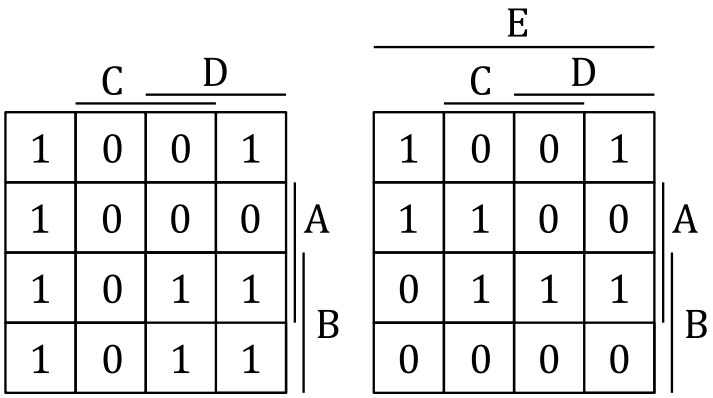
Bestimmen Sie die minimale Polynomialform der in der Karnaugh-Tafel der folgenden Abbildung dargestellten Funktion.



kar/productsum-03

2.4 Karnaugh-Tafel mit 5 Variablen

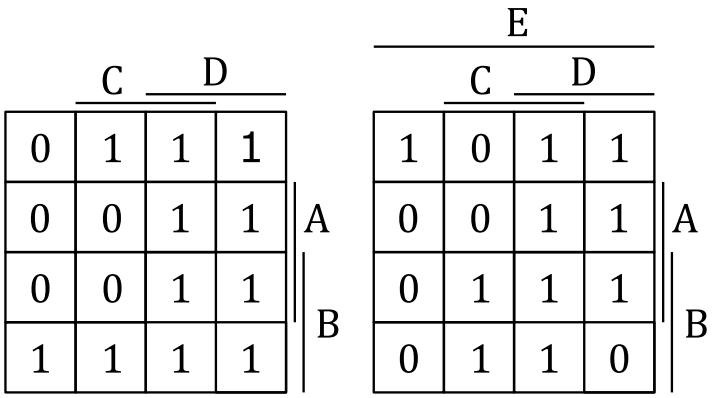
Bestimmen Sie die minimale Polynomialform der in der Karnaugh-Tafel der folgenden Abbildung dargestellten Funktion.



kar/productsum-04

2.5 Karnaugh-Tafel mit 5 Variablen

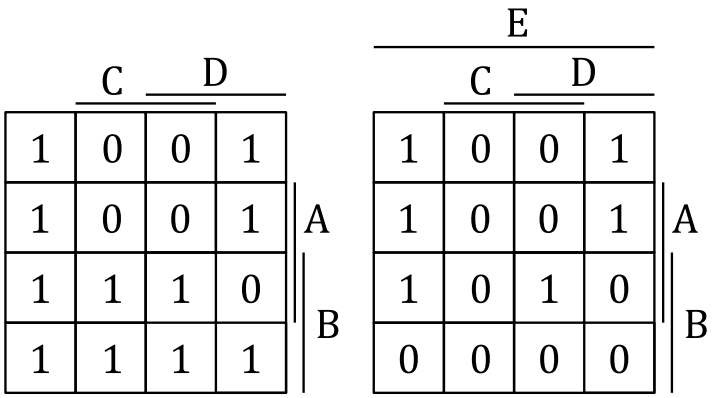
Bestimmen Sie die minimale Polynomialform der in der Karnaugh-Tafel der folgenden Abbildung dargestellten Funktion.



kar/productsum-05

2.6 Karnaugh-Tafel mit 5 Variablen

Bestimmen Sie die minimale Polynomialform der in der Karnaugh-Tafel der folgenden Abbildung dargestellten Funktion. s



kar/productsum-06

2.7 Minimale Polynomialform

Bestimmen Sie die minimale Polynomialform der Funktion

$$y = \overline{x_3\ x_2\ \overline{x_0} + \overline{x_2}\ \overline{x_1}\ \overline{x_0} + x_2\ \overline{x_1}\ x_0} \tag{13}$$

kar/productsum-07

2.8 Inverse Funktion

Sei die Funktion gegeben durch die Gleichung

$$y = \overline{a}\ \overline{c} + a\ \overline{b}\ \overline{e} + b\ \overline{c}\ \overline{d} + \overline{a}\ \overline{b}\ e \tag{14}$$

Geben Sie die die minimale Polynomialform der inversen Funktion $\overline{y}$.

kar/productsum-08

2.9 Minimale Polynomialform

Für die Funktion, welche in der nebenstehenden Tafel angegeben ist, bestimmen Sie, welche Form die kleinste Anzahl an Termen hat: diejenige der Funktion Y oder diejenige der Funktion $\overline{Y}$.

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

kar/productsum-09

2.10 Funktion von 5 Variablen

Bestimmen Sie die minimale Polynomialform der Mehrheitsfunktion mit 5 Eingängen:

- die Funktion ergibt eine **`'0'`**, wenn mehr als die Hälfte der Eingänge auf **`'0'`** sind.
- die Funktion ergibt eine **`'1'`**, wenn mehr als die Hälfte der Eingänge auf **`'1'`** sind.

kar/productsum-10

2.11 Unvollständig definierte Funktion

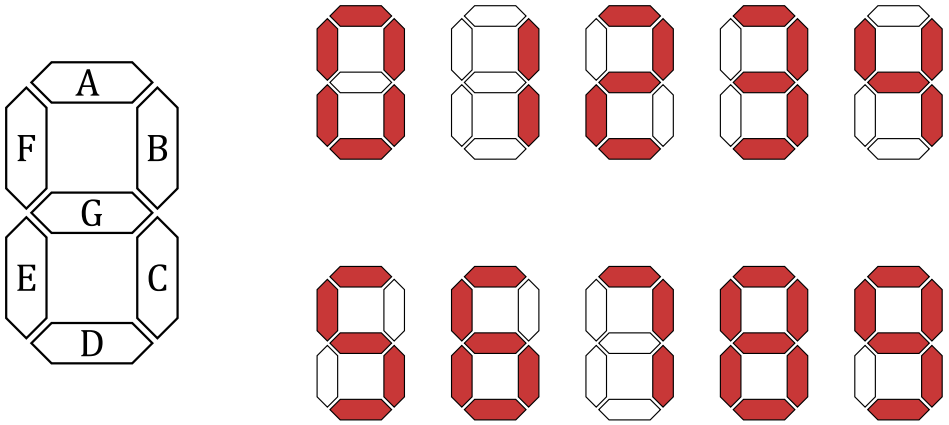
Bestimmen Sie die minimale Polynomialform der Mehrheitsfunktion mit 4 Eingängen:

- die Funktion ergibt eine **`'0'`**, wenn mehr als die Hälfte der Eingänge auf **`'0'`** sind.
- die Funktion ergibt eine **`'1'`**, wenn mehr als die Hälfte der Eingänge auf **`'1'`** sind,
- wenn gleich viele Eingänge auf **`'0'`** und auf **`'1'`** sind, dann kann das Resultat irgendwelchen Wert nehmen.

kar/productsum-11

2.12 Unvollständig definierte Funktion

Eine 7-Segment-Anzeige wird angewandt, um eine Dezimalziffer anzugeben.



Erstellen Sie eine kombinatorische Logikschaltung, welche eine auf 4 Bit codierte Binärzahl zu den 7 Steuersignalen zur Einleuchtung der 7 Segmenten umwandelt.
In diesem System kann die Eingangszahl nur die Binärwerte aufnehmen, welche den Dezimalziffern von 0_d bis 9_d entsprechen.

kar/productsum-12

3 KAR - Vereinfachung von XOR-Funktionen

3.1 Darstellung von XOR-Funktionen

Stellen sie in jeweils eine Karnaugh-Tafel von 4 Variablen die folgenden Funktionen dar:

$y_1 = a \oplus b$
$y_2 = a \oplus b \oplus c$
$y_3 = a \oplus b \oplus c \oplus d$
$y_4 = \overline{a \oplus b}$
$y_5 = \overline{a} \oplus b$
$y_6 = a \oplus \overline{b}$
$y_7 = d\overline{b} \oplus \overline{d}b$
$y_8 = d \oplus b$
$y_9 = \overline{b} \oplus \overline{d}$

kar/xor-01

3.2 Minimale Polynomialform

Bestimmen Sie die minimale Polynomialform der Funktion

$$y = x_1 \oplus \overline{x_1}x_0 \oplus x_2\overline{x_1} \oplus x_3x_2\overline{x_0} \tag{15}$$

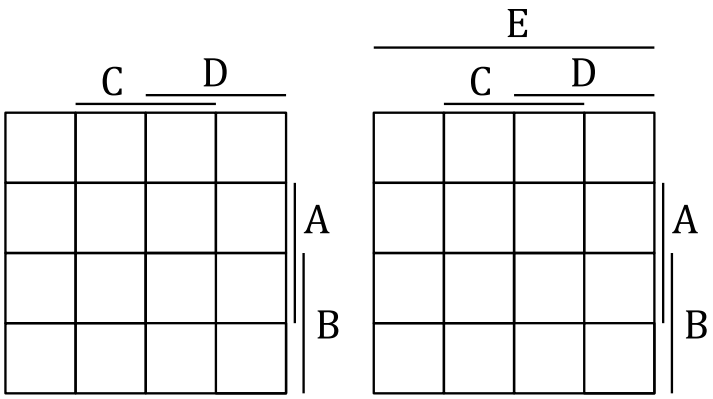
kar/xor-02

3.3 Minimale Polynomialform

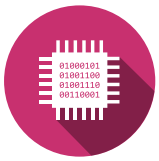
Bestimmen Sie die minimale Polynomialform der Funktion

$$y = e \oplus \overline{d} \oplus dc \oplus d\overline{b} \oplus \overline{c}a \tag{16}$$

Wenn Sie mit einer Karnaugh-Tafel arbeiten, so wählen Sie dieselbe Anordnung der Variablen wie in der Tafel der folgenden Abbildung.



kar/xor-03



3.4 Darstellung in der Form von XOR von Produkten

Schreiben Sie die Gleichung der Funktion, welche in der folgenden Karnaugh-Tafel angegeben ist, in der Form von XOR von Produkten.

	C		D	
1	1	0	0	
0	1	1	0	A
1	1	0	0	
0	1	1	0	B

kar/xor-04

3.5 Darstellung in der Form von XOR von Produkten

Schreiben Sie die folgende Funktion in der Form von XOR von Produkten um.

$$y = x_1\ x_0 + \overline{x_2}\ x_1 + \overline{x_2}\ x_0 + \overline{x_3}\ x_2\ \overline{x_1} \tag{17}$$

kar/xor-05

3.6 Addierer

Zeichnen Sie das Schema eines 2-Bit Addierers nur mit Hilfe von AND- und XOR Gattern.
Der Addierer erstellt die Summe von 2 Zahlen, die auf 2 Bits codiert sind. Er gibt ein Resultat auf 3 Bits.
Minimieren Sie in erster Priorität die Verzögerung zwischen Ein- und Ausgänge. Betrachten Sie dabei, dass alle Gatter dieselbe Verzögerung vorweisen. Minimieren Sie als nächstes die Anzahl an Logikgattern.

kar/xor-06

4 KAR - Funktionen mit einer grossen Anzahl an Eingängen

4.1 Zahlenvergleich

Erstellen Sie eine Schaltung, welche angibt, ob eine Binärzahl A grösser als eine Binärzahl B ist. Die Zahlen sind auf 8 Bit codiert. Sie entsprechen positive, ganze Zahlen.

kar/manyinputs-01

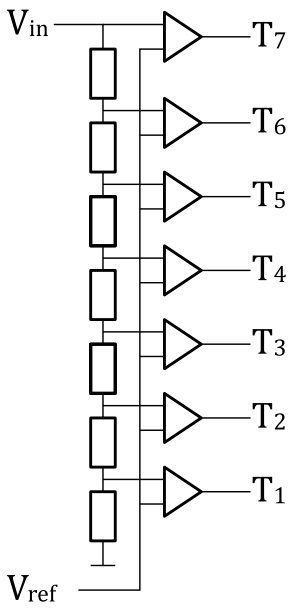
4.2 Binäraddierer

Erstellen Sie eine Schaltung, welche die Summe von zwei 8-Bit erzeugt. Das Resultat soll auch auf 8 Bit gegeben sein.
Achten Sie darauf, die Anzahl an Logikgattern zu minimieren.

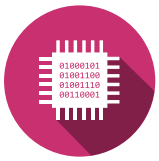
kar/manyinputs-02

4.3 Thermometer-Code zu Binärcode Umwandlung

Ein 3-Bit analog/digital Wandler funktionniert wie folgt: Die Eingangsspannung wird durch eine Kette von 7 Widerständen geteilt. Jede Zwischenspannung ist zu einer Referenzspannung verglichen. Der so erzeugte Binärcode wird als Thermometer-Code bezeichnet.
Erstellen Sie eine Schaltung, welche diesen Thermometer-Code zu einem Binärcode umwandelt.



kar/manyinputs-03



4.4 Übermittlung anhand der Priorität

Erzeugen Sie eine logische, kombinatorische Schaltung, welche von allen Eingangssignalen nur dasjenige mit der höchsten Priorität weiterleitet.
Das System beträgt 8 Eingänge, I_1 aufI_8, und 8 Ausgänge, O_1 auf O_8. Der Eingang I_8 hat die höchste Priorität.
Die folgende Tafel gibt einige Funktionsbeispiele:

$I_1 ... I_8$	$O_1 ... O_8$
00000000	00000000
00010000	00010000
11000000	01000000
11010000	00010000
11010010	00000010

kar/manyinputs-04

4.5 Logik für Zähler ohne Rückgang auf Null

Zum Entwurf eines 16-Bit Zählers, erstellen Sie eine kombinatorische Schaltung, welche den nächsten Wert des Zählers, y, als Funktion des gegenwärtigen Wertes, x, erzeugt.
Die zu erzeugende Funktion ist $y = x + 1$ solange seinen maximalen Wert, $x = \mathrm{FFFF}_h$ (alle Bits auf 1), nicht erreicht hat. In diesem Speziellfall erzeugt die Schaltung den Ausgangswert $y = \mathrm{FFFF}_h$, um einen Rückgang auf Null zu vermeiden.

kar/manyinputs-05

4.6 Addierer mit Sättigung

Schlagen Sie eine Schaltung eines Addierers vor, dessen Eingänge sowohl wie der Ausgang auf 16 Bit codiert sind, und wo die Überschreitung des Maximalwertes mit Sättigung behandelt wird. Die Operation findet auf positive Zahlen statt.
Die Sättigung ist wie folgt definiert: wenn die Summe der 2 Zahlen den Maximalwert $y = \mathrm{FFFF}_h$ (alle Bits auf 1) überschreitet, so erzeugt die Schaltung diesen Maximalwert am Ausgang.

kar/manyinputs-06

4.7 BCD-codierte Zahlen

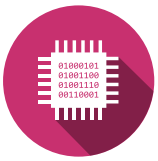
Schlagen Sie eine Schaltung vor, welche die Summe von zwei BCD-codierte Zahlen (Binary Coded Decimal) erzeugt.
Die 2 Zahlen sind auf 3 BCD-Ziffern codiert, was 12 Bits entspricht. Das Resultat ist auf 4 BCD-Ziffern zu erzeugen, somit auf 16 Bits.

kar/manyinputs-07

4.8 Mehrheitsfunktion mit 7 Eingängen

Mit Hilfe von Invertern, UND-, ODER- und Exklusiv-ODER-Gatter, zeichnen Sie das Schema der Schaltung, welche die Mehrheit von 7 Eingängen bestimmt.

kar/manyinputs-08



4.9 Arithmetische und logische Einheit

Zeichnen Sie das Schema einer Arithmetischen und logischen Einheit (Arithmetic and Logic Unit, ALU) eines Mikroprozessors.
Die ALU arbeitet auf 8 Bits. Die möglichen Operationen sind:

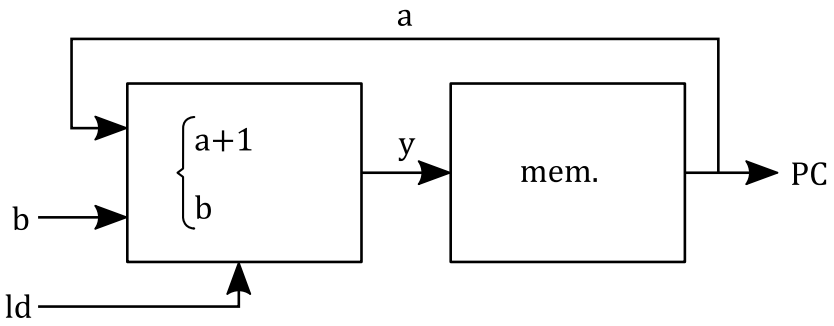
Kontrolle	Operation	
00	Addition	$y = a + b$
01	Substraktion	$y = a - b$
10	UND	$y = a$ AND b
11	ODER	$y = a$ OR b

Dies logische Operationen werden Bit für Bit durchgeführt, Beispiel: $y_i = a_i$ AND $b_i, \forall i$.

kar/manyinputs-09

4.10 Logik für Programmzähler

Für den Programmzähler (Program Counter, PC) eines Mikroprozessors braucht man einen Block, welcher den Zähler entweder inkrementiert (Suche des nächsten Befehls) oder mit einen neuen Wert lädt (Sprung).

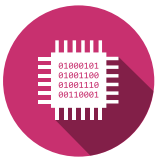


Entwerfen Sie eine kombinatorische Schaltung, welche berechnet,

$$y = \begin{cases} \text{ld} = 0 \Rightarrow a + 1 \\ \text{ld} = 1 \Rightarrow b \end{cases} \tag{18}$$

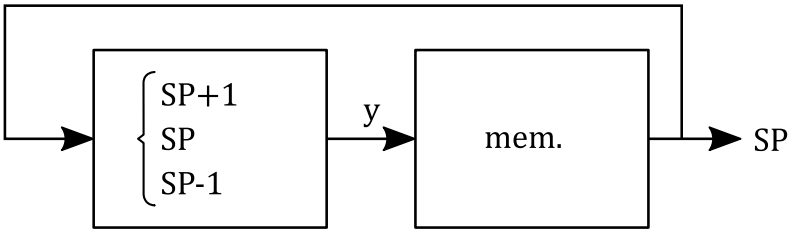
wobei die Zahlen a, b und y auf 16 Bits kodiert sind. wobei die Zahlen a, b und y auf 16 Bits kodiert sind.

kar/manyinputs-10



4.11 Logik für Stackpointer

Für den Stackpointer (Stack Pointer, SP) eines Mikroprozessors braucht man einen Block, welcher den Pointer entweder inkrementiert (ein Wert wird aufgestapelt), dekrementiert (ein Wert wird abgestapelt) oder gleich bleibt (keine Operation mit der Stapel).



Der Stackpointer ist auf 16 bits codiert.

kar/manyinputs-11