Grundlagen der Rechnerarchitektur

[CS3100.010]

Wintersemester 2014/15

Heiko Falk

Institut für Eingebettete Systeme/Echtzeitsysteme
Ingenieurwissenschaften und Informatik
Universität Ulm

Kapitel 2

Kombinatorische Logik

Inhalte der Vorlesung

1. Einführung
2. **Kombinatorische Logik**
3. Sequentielle Logik
4. Technologische Grundlagen
5. Rechnerarithmetik
6. Grundlagen der Rechnerarchitektur
7. Speicher-Hardware
8. Ein-/Ausgabe

Inhalte des Kapitels (1)

2. **Kombinatorische Logik**
   - Einleitung
     - Digitale Schaltungen
     - Logische Schaltungen und Gatter
     - Boolesche Algebra
   - Schaltfunktionen
     - Beispiele für wichtige Schaltfunktionen
     - Darstellung von Schaltfunktionen
       - Wahrheitstabellen
       - Normalformen (KDNF, KKNF)
     - Hauptsatz der Schaltalgebra
   - ...

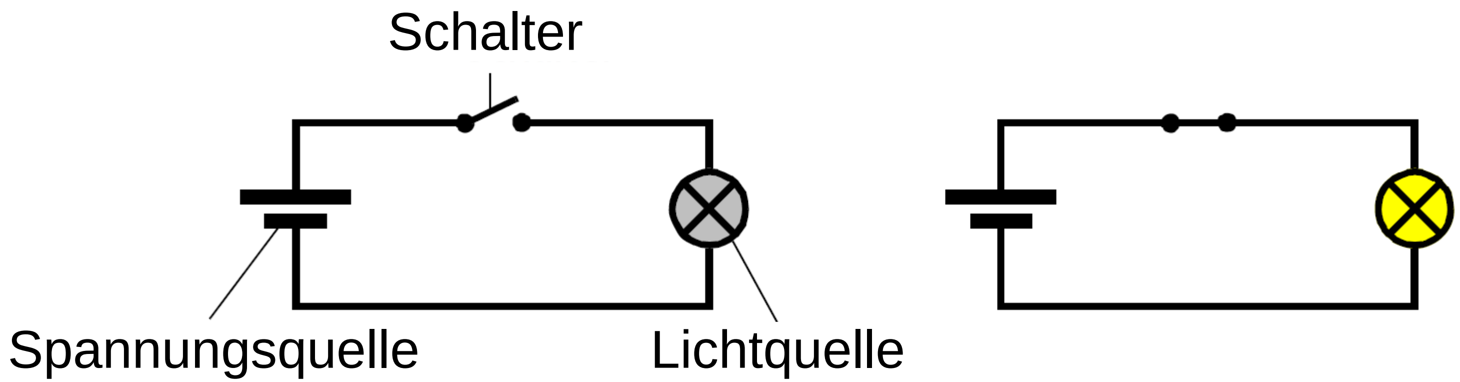
Inhalte des Kapitels (2)

2. Kombinatorische Logik

- …
- Synthese von Schaltungen
  - Beispiele
    - ODER-Funktion, Eingabemelder, Siebensegmentanzeige
  - Äquivalenz von Schaltfunktionen
  - Minimierung von Schaltungen
    - Grundlagen
    - Vorgehensweise
    - Karnaugh-Veitch Diagramme
    - Beispiele
- Schaltnetze / *Combinational Networks*
  - Vorgehensweise der Synthese für Schaltnetze
  - Typische Schaltnetze

Digitale elektrische Systeme

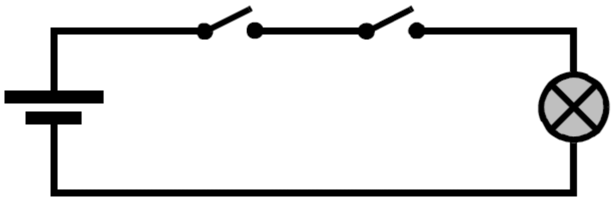
Einfaches Szenario: Lampenschaltung



- Strom kann bei geschlossenem Stromkreis fließen
- Zwei Zustände:
  - Strom fließt – Lampe leuchtet
  - Strom fließt nicht – Lampe bleibt dunkel

Einfache Schaltungen (1)

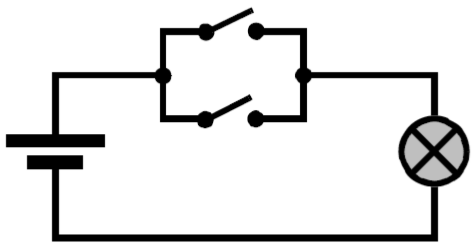
Einfaches Szenario: Zwei Schalter in Reihe



– Lampe leuchtet nur dann, wenn der erste ***und*** der zweite Schalter geschlossen sind

Einfache Schaltungen (2)

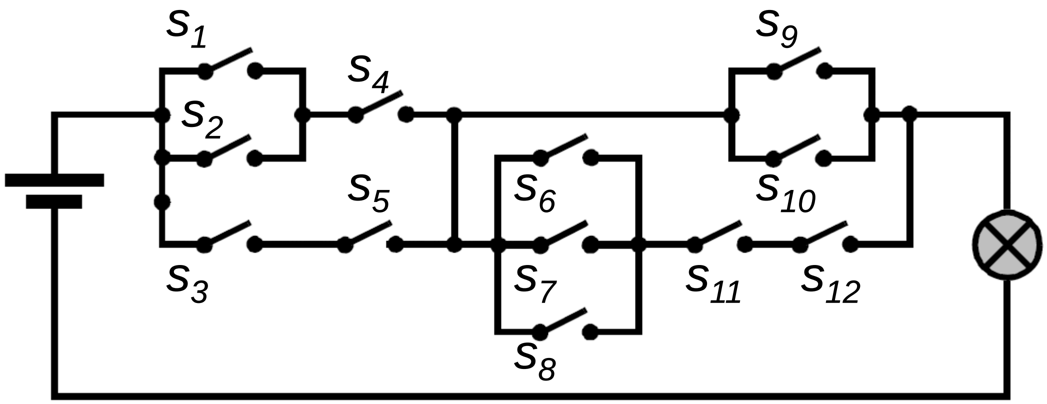
Einfaches Szenario: Zwei parallele Schalter



– Lampe leuchtet nur dann, wenn der erste ***<u>oder</u>*** der zweite Schalter geschlossen sind

Komplexe Schaltung

Komplexes Szenario: Viele Schalter



– Wann leuchtet die Lampe?

Realisierungsmöglichkeiten

Bisher

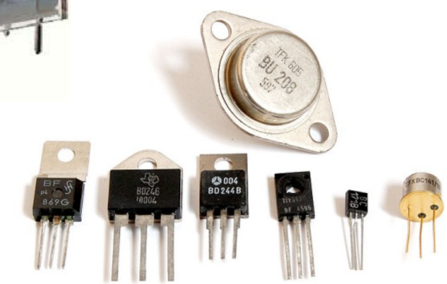
– Stromkreis, Schalter und Lampe

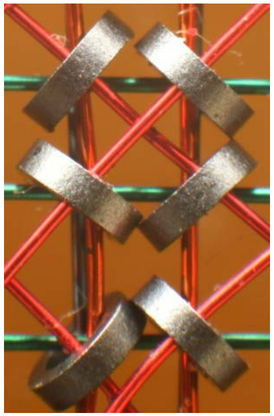
Unterschiedlichste Technologien

– Mechanisch, optoelektronisch, elektromechanisch, ***elektronisch***
– Flüssigkeitslogik *(Fluidic Logics)*

Repräsentation der zwei Zustände

– Spannung, Strom, Ladung
– Perforation
– Magnetisierung
– Licht
– Flüssigkeitsströmung
– Bio ...





[de.wikipedia.org]

Logische Schaltungen (1)

Betrachtung aus dem Blickwinkel der Logik

- Zwei Zustände
  - an – aus
  - Stromfluss – kein Stromfluss
  - wahr – falsch
  - *true* – *false*
  - 1 – 0

Realisierung mit elektronischen Bausteinen

- Zwei Zustände, z.B. Stromfluss – kein Stromfluss

Logische Schaltungen (2)

Elementare Funktionen über die beiden Zustände

– Realisiert mit Funktionsbausteinen (Gatter)

– Eingänge und Ausgänge

Komplexere Funktionen durch mehrere zusammengeschaltete Gatter

☞ **Logische Funktion tritt in den Vordergrund**

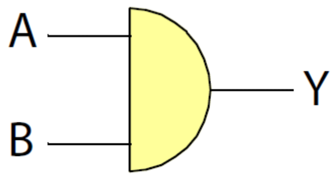
– Elektronik wird weniger wichtig

☞ **Betrachtung elementarer Funktionen / Gatter**

UND-Gatter

Implementierung der *Konjunktion*

– Zwei Parameter

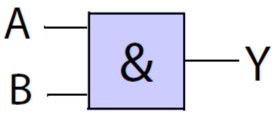


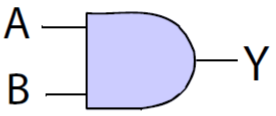
$Y = A \cdot B$

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

Wahrheitstafel

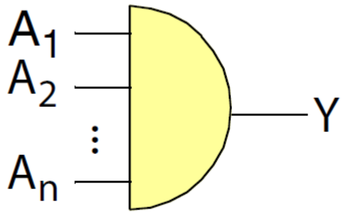
Alternative Schreibweisen





$Y = A \wedge B$
$Y = AB$
$Y = A\&B$

– Mehrere Parameter

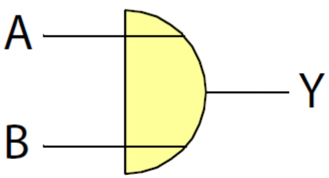


$Y = A_1 \cdot A_2 \cdot \ldots \cdot A_n$

ODER-Gatter

Implementierung der *Disjunktion*

– Zwei Parameter

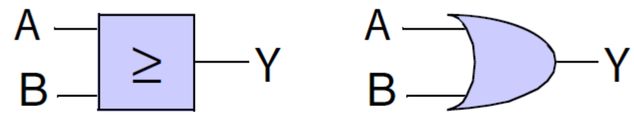


$$Y = A + B$$

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

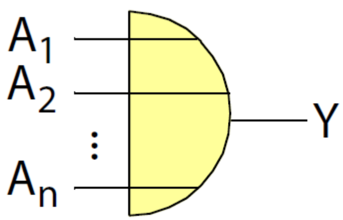
Wahrheitstafel

Alternative Schreibweisen



$$Y = A \vee B$$

– Mehrere Parameter

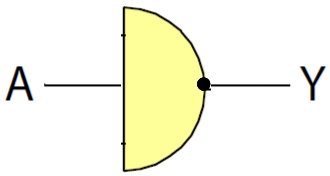


$$Y = A_1 + A_2 + \ldots + A_n$$

NICHT-Gatter

Implementierung der *Negation*

– Ein Parameter

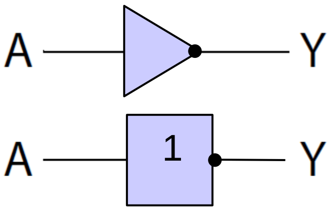


$Y = \overline{A}$

A	Y
0	1
1	0

Wahrheitstafel

Alternative Schreibweisen



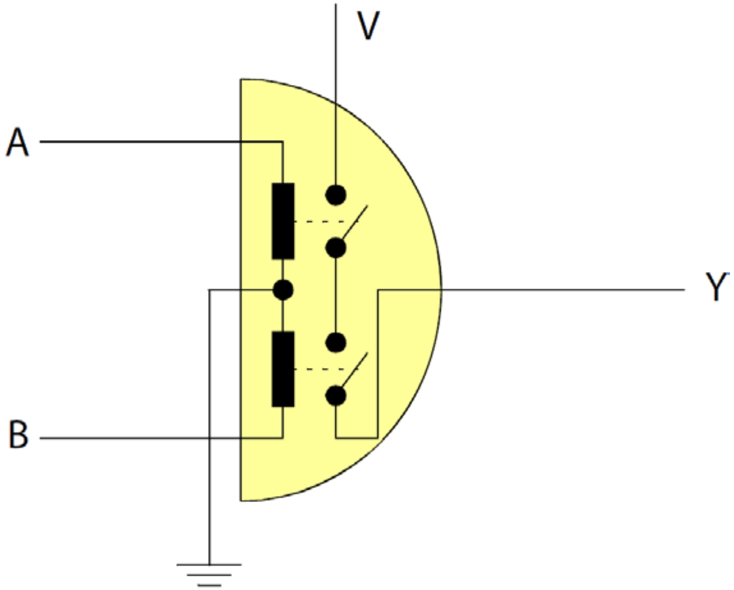
$Y = \neg A$

$Y = A'$

Mögliche Realisierung (1)

Implementierung einer Gatterfunktion

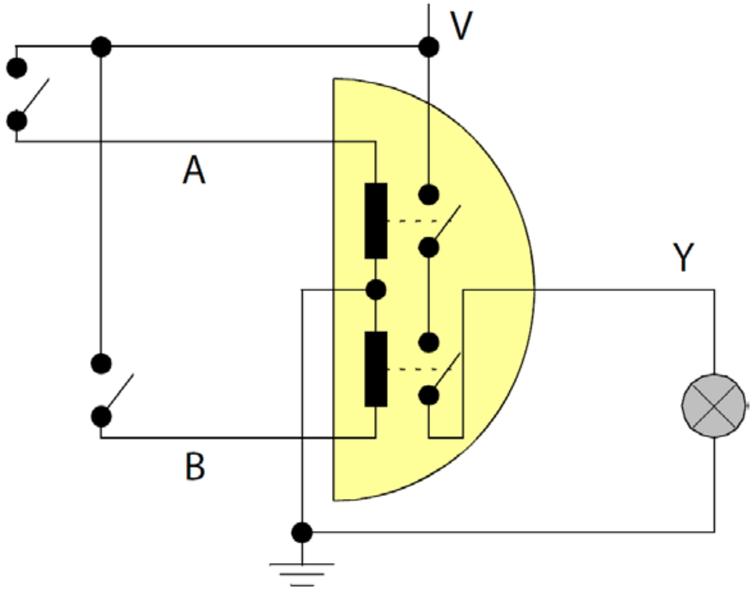
- Z.B. elektrische Implementierung mit Relais
  - 1 – Stromfluss möglich
  - 0 – kein Stromfluss möglich
- Möglicher Aufbau eines UND-Gatters



Mögliche Realisierung (2)

Implementierung einer Gatterfunktion

- Z.B. elektrische Implementierung mit Relais
  - 1 – Stromfluss möglich
  - 0 – kein Stromfluss möglich
- Möglicher Aufbau eines UND-Gatters

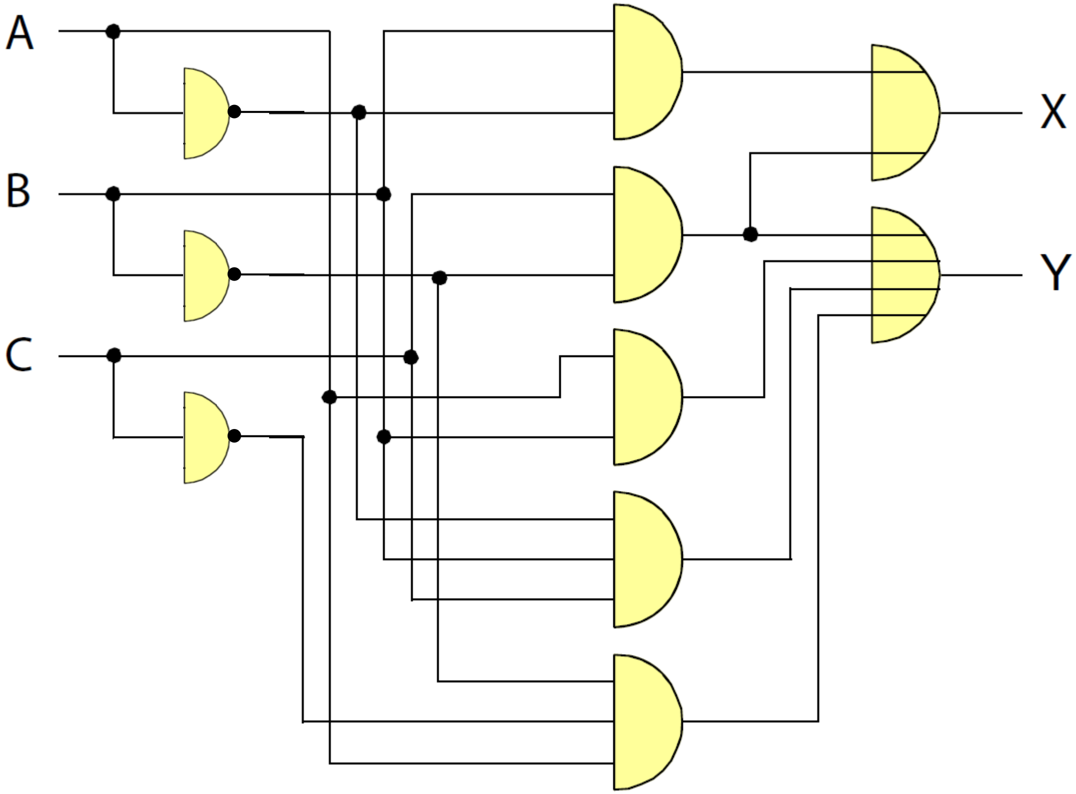


Logische Schaltungen (1)

Zusammenschalten mehrerer elementarer Gatter zu einer Schaltung

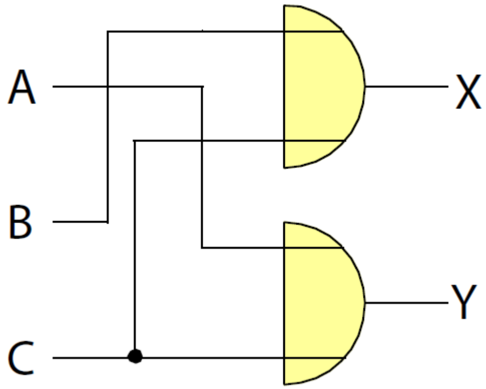
– Eingänge und Ausgänge

Beispiel 1:



Logische Schaltungen (2)

Beispiel 2:



Problemkreise

– Synthese von Schaltungen aus Wahrheitstabelle
– Nachweis der Äquivalenz von Schaltungen
– Minimaler Aufbau von Schaltungen (Kostenersparnis)

Boolesche Algebra

Von George Boole 1854 entwickelte Algebra

- Zwei Werte: 0 und 1
- Drei Operationen: + und * und $\overline{}$
- Vier Axiome/Rechengesetze (nach Huntington, 1904):
  1. Kommutativität

     $A + B = B + A$ $\qquad$ $A * B = B * A$
  2. Neutrales Element

     $0 + A = A$ $\qquad$ $1 * A = A$
  3. Distributivität

     $(A + B) * C = (A * C) + (B * C)$

     $(A * B) + C = (A + C) * (B + C)$
  4. Komplementäres Element

     $A + \overline{A} = 1$ $\qquad$ $A * \overline{A} = 0$

Wichtige Sätze (1)

Aus den Axiomen beweisbare Sätze

– Abgeschlossenheit

Boolesche Operationen liefern nur Boolesche Werte als Ergebnis

– Assoziativität

$A + (B + C) = (A + B) + C$ $\qquad$ $A * (B * C) = (A * B) * C$

– Idempotenz

$A + A = A$ $\qquad$ $A * A = A$

– Absorption

$A + (A * B) = A$ $\qquad$ $A * (A + B) = A$

Wichtige Sätze (2)

Aus den Axiomen beweisbare Sätze

– Doppeltes Komplement

$\overline{(\overline{A})} = A$

– Komplementäre Werte

$\overline{0} = 1$ $\overline{1} = 0$

– Satz von De Morgan

$\overline{(A + B)} = \overline{A} * \overline{B}$ $\overline{(A * B)} = \overline{A} + \overline{B}$

– Dualität

Für jede aus den Axiomen ableitbare Aussage gibt es eine duale Aussage, die durch Tausch der Operationen + und * sowie durch Tausch der Werte 0 und 1 entsteht.

Schaltalgebra

Boolesche Algebra für logische Schaltungen

- Operation UND (*)
- Operation ODER (+)
- Operation NICHT ($\overline{\ }$)
- Zwei Zustände 0 und 1 entsprechen den Werten

Gesetze der Booleschen Algebra helfen bei

- Synthese von Schaltungen aus Wahrheitstabellen
- Nachweis der Äquivalenz von Schaltungen
- Minimierung von Schaltungen

Roter Faden

2. Kombinatorische Logik

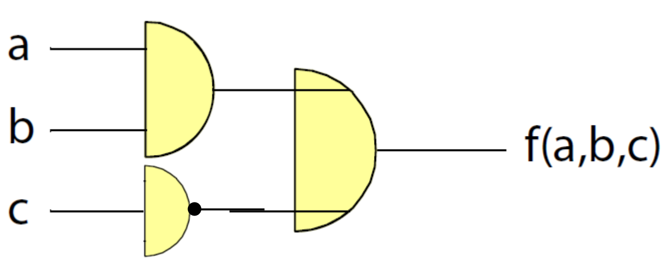
- Einleitung
  - Digitale Schaltungen
  - Logische Schaltungen und Gatter
  - Boolesche Algebra
- Schaltfunktionen
- Synthese von Schaltungen
- Schaltnetze / *Combinational Networks*

Schaltfunktionen

Zweiwertige Funktion über zweiwertige Variablen

- Eindeutige Zuordnungsvorschrift
  - Mögliche Wertekombinationen der Eingabevariablen
  - Je ein Ergebniswert
- Bei *n* Eingabevariablen ergeben sich 2^n Kombinationsmöglichkeiten für mögliche Eingabewerte
- Darstellung durch Boolesche Ausdrücke
  - Variablen und Operationen aus der Booleschen Algebra

Beispiel: $f(a, b, c) = a * b + \overline{c}$



Wahrheitstabellen

Tabellarische Darstellung der Zuordnung

Beispiel:

a	b	c	f(a,b,c)
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

– Es gibt $2^{(2^n)}$ verschiedene *n*-stellige Schaltfunktionen
    – 256 dreistellige Schaltfunktionen

Einstellige Schaltfunktionen

Vier einstellige Schaltfunktionen

Name		Null-funktion	Negation	Identitäts-funktion	Eins-funktion
Wahr-heits-tabelle	x	$f(x)$	$f(x)$	$f(x)$	$f(x)$
	0	0	1	0	1
	1	0	0	1	1
Ausdruck		$f(x) = 0$	$f(x) = \overline{x}$	$f(x) = x$	$f(x) = 1$

Zweistellige Schaltfunktionen (1)

Sechzehn zweistellige Schaltfunktionen

Name				Konjunktion		
Wahrheitstabelle	x	y	$f(x, y)$	$f(x, y)$	$f(x, y)$	$f(x, y)$
	0	0	0	0	0	0
	0	1	0	0	0	0
	1	0	0	0	1	1
	1	1	0	1	0	1
Ausdruck			$f(x, y) = 0$	$f(x, y) = x * y$	$f(x, y) = x * \overline{y}$	$f(x, y) = x$

– Konjunktion: UND-Funktion

Zweistellige Schaltfunktionen (2)

Sechzehn zweistellige Schaltfunktionen

Name					exklusives Oder	Disjunktion
Wahr-heits-tabelle	x	y	$f(x, y)$	$f(x, y)$	$f(x, y)$	$f(x, y)$
	0	0	0	0	0	0
	0	1	1	1	1	1
	1	0	0	0	1	1
	1	1	0	1	0	1
Ausdruck			$f(x, y) = \overline{x} * y$	$f(x, y) = y$	$f(x, y) = \overline{x} * y + x * \overline{y}$	$f(x, y) = x + y$

– Exklusives Oder: Antivalenz, Addition Modulo 2, $f(x, y) = x \oplus y$

– Disjunktion: ODER-Funktion

Zweistellige Schaltfunktionen (3)

Sechzehn zweistellige Schaltfunktionen

Name			Peirce-Funktion	Äquivalenz	
Wahr-heits-tabelle	x	y	$f(x, y)$	$f(x, y)$	$f(x, y)$
	0	0	1	1	1
	0	1	0	0	0
	1	0	0	0	1
	1	1	0	1	0
Ausdruck			$f(x, y) = \overline{x} * \overline{y}$	$f(x, y) = x * y + \overline{x} * \overline{y}$	$f(x, y) = \overline{y}$

– Peirce-Funktion: Negiertes Oder, NOR

– Äquivalenz: $f(x, y) = x \equiv y$

Zweistellige Schaltfunktionen (4)

Sechzehn zweistellige Schaltfunktionen

Name			Implikation		Implikation	Sheffer-Funktion	Eins-Funktion
Wahrheitstabelle	x	y	$f(x, y)$	$f(x, y)$	$f(x, y)$	$f(x, y)$	$f(x, y)$
	0	0	1	1	1	1	1
	0	1	0	1	1	1	1
	1	0	1	0	0	1	1
	1	1	1	0	1	0	1
Ausdruck			$f(x, y) = x + \overline{y}$	$f(x, y) = \overline{x}$	$f(x, y) = \overline{x} + y$	$f(x, y) = \overline{x * y}$	$f(x, y) = 1$

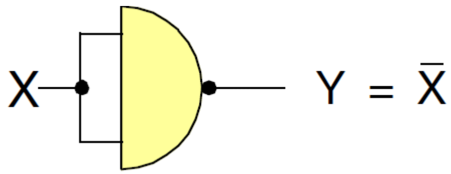
- Implikation: $f(x, y) = y \Rightarrow x$ bzw. $f(x, y) = x \Rightarrow y$
- Sheffer-Funktion: negiertes UND, NAND

Sheffer-Funktion (1)

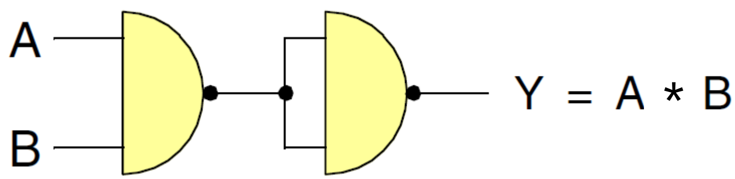
Eine Funktion zur Darstellung aller anderen

– Sheffer-Funktion / NAND

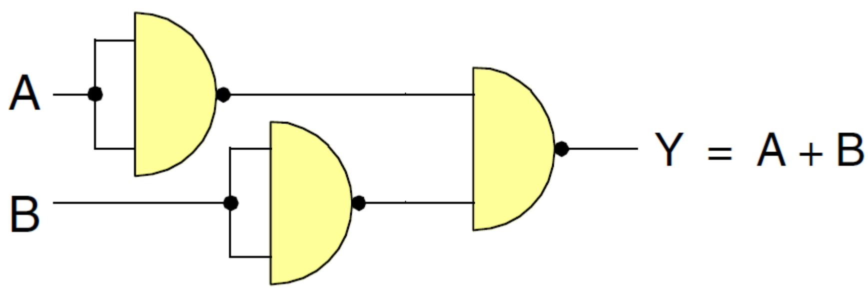
– NICHT-Funktion



– UND-Funktion



– ODER-Funktion



Sheffer-Funktion (2)

Die drei grundlegenden Funktionen UND, ODER, NICHT sind darstellbar

☞ Alle Schaltfunktionen sind alleine mit NAND darstellbar

Neben Sheffer-Funktion auch Pierce-Funktion NOR

– Alle grundlegenden Funktionen ebenfalls mit NOR darstellbar

NAND leicht in Hardware realisierbar

Darstellung von Schaltfunktionen

Mögliche Darstellungen

– Wahrheitstabelle
– Boolescher Ausdruck
– Schaltung mit elementaren Gattern (z.B. UND, ODER, NICHT)

Problem

– Boolescher Ausdruck und Schaltung sind nicht eindeutig
☞ Mehrere Ausdrücke (Schaltungen) repräsentieren die gleiche Schaltfunktion

Normalformen (1)

Normierte Ausdrücke zur Darstellung von Schaltfunktionen

Begriffe

- Produktterm
  - Einfache Variable (evtl. negiert)
  - Konjunktion einfacher Variablen (jeweils evtl. negiert)
  - Beispiele: x, $\overline{x}$, $x * y * \overline{z}$

- Summenterm
  - Analog zu Produktterm, jedoch Disjunktion
  - Beispiele: x, $\overline{x}$, $x + y + \overline{z}$

Normalformen (2)

Weitere Begriffe

- Minterm
  - Produktterm, in dem jede Variable einer Schaltfunktion genau einmal vorkommt (einfach oder negiert)
  - Beispiel:
    $\overline{x_1} * x_2 * \overline{x_3} * x_4$ ist Minterm für Funktion $f(x_1, x_2, x_3, x_4)$

- Maxterm
  - Summenterm, in dem jede Variable einer Schaltfunktion genau einmal vorkommt (einfach oder negiert)
  - Beispiel:
    $\overline{x_1} + x_2 + \overline{x_3} + x_4$ ist Maxterm für Funktion $f(x_1, x_2, x_3, x_4)$

Normalformen (3)

Noch mehr Begriffe

- Disjunktive Normalform (DNF)
  - Disjunktion von Produkttermen
  - Beispiel:
    $x + x * \overline{y} + y * \overline{z}$
- Konjunktive Normalform (KNF)
  - Konjunktion von Summentermen
  - Beispiel:
    $x * (x + \overline{y}) * (y + \overline{z})$

Normalformen (4)

Noch mehr aber besonders wichtige Begriffe

- Kanonische disjunktive Normalform (KDNF, DKF)
  - Disjunktion einer Menge von Mintermen mit gleichen Variablen
  - Beispiel: KDNF zur Funktion $f(x_1, x_2, x_3, x_4)$

    $$x_1 * \overline{x_2} * x_3 * x_4 + \overline{x_1} * x_2 * \overline{x_3} * x_4 + \overline{x_1} * \overline{x_2} * x_3 * \overline{x_4}$$

- Kanonische konjunktive Normalform (KKNF, KKF)
  - Konjunktion einer Menge von Maxtermen mit gleichen Variablen
  - Beispiel: KKNF zur Funktion $f(x_1, x_2, x_3, x_4)$

    $$(x_1 + \overline{x_2} + x_3 + x_4) * (\overline{x_1} + x_2 + \overline{x_3} + x_4) * (\overline{x_1} + \overline{x_2} + x_3 + \overline{x_4})$$

Hauptsatz der Schaltalgebra (1)

Jede Schaltfunktion lässt sich als genau eine KDNF darstellen

- Für jedes $f(\vec{x}) = 1$ aus der Wahrheitstabelle bilde man einen Minterm für die KDNF.
- Eine Variable x_i wird invertiert, wenn die Variable für diesen Eintrag in der Wahrheitstabelle 0 ist, ansonsten einfach verwendet.
- Beispiel:

x_1	x_2	x_3	x_4	$f(x_1,x_2,x_3,x_4)$
		...		
0	1	0	1	1

$$\rightarrow \quad \overline{x_1} \cdot x_2 \cdot \overline{x_3} \cdot x_4$$

- Darstellung durch KDNF bis auf Vertauschungen eindeutig (Kommutativität)

Hauptsatz der Schaltalgebra (2)

Jede Schaltfunktion lässt sich als genau eine KKNF darstellen

– Für jedes $f(\vec{x}) = 0$ aus der Wahrheitstabelle bilde man einen Maxterm für die KKNF.

– Eine Variable x_i wird invertiert, wenn die Variable für diesen Eintrag in der Wahrheitstabelle 1 ist, ansonsten einfach verwendet.

– Beispiel:

x_1	x_2	x_3	x_4	$f(x_1,x_2,x_3,x_4)$
		...		
0	1	0	1	0

$\rightarrow$ $x_1 + \overline{x_2} + x_3 + \overline{x_4}$

– Darstellung durch KKNF bis auf Vertauschungen eindeutig (Kommutativität)

Hauptsatz der Schaltalgebra (3)

Überführung der kanonischen Normalformen ineinander

– Wegen Dualität gilt:

$$\mathrm{KDNF}(\ f(\vec{x})\ ) = \overline{\mathrm{KKNF}(\ \overline{f(\vec{x})}\ )}$$

und

$$\mathrm{KKNF}(\ f(\vec{x})\ ) = \overline{\mathrm{KDNF}(\ \overline{f(\vec{x})}\ )}$$

Roter Faden

2. Kombinatorische Logik

- Einleitung
- Schaltfunktionen
  - Beispiele für wichtige Schaltfunktionen
  - Darstellung von Schaltfunktionen
    - Wahrheitstabellen
    - Normalformen (KDNF, KKNF)
  - Hauptsatz der Schaltalgebra
- Synthese von Schaltungen
- Schaltnetze / *Combinational Networks*

Synthese von Schaltungen

Vorgehen

- Aufstellen der Wahrheitstabelle
- Bilden der KDNF (oder KKNF)
- Aufbau der dazugehörigen Schaltung

Beispiel: ODER-Funktion

Wahrheitstabelle für die zweistellige ODER-Funktion

x_1	x_2	$f(x_1, x_2)$
0	0	0
0	1	1
1	0	1
1	1	1

Bildung der KDNF

– Suchen der Stellen mit $f(x_1, x_2) = 1$

– Summieren der entsprechenden Minterme:

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

Problem

– Schaltung wird in aller Regel nicht minimal sein

Einschub: Binärzahlen

Darstellung ganzer Zahlen mit Hilfe von binären Zuständen

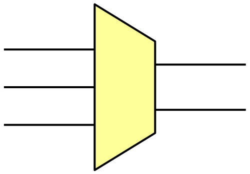
- Darstellung mit 0 und 1 aber mehreren Stellen
- Z.B. zweistellige Binärzahlen
  - 4 Möglichkeiten: 00, 01, 10, 11
- Z.B. dreistellige Binärzahlen
  - 8 Möglichkeiten: 000, 001, 010, 011, 100, 101, 110, 111

Wert einer Binärzahl

- Beispiel: $(011)_2$
  - Stufenzahlen aus dem Binärsystem: Zweierpotenzen, 2n
  - Berechnung:

    $(011)_2 = 0 * 2^2 + 1 * 2^1 + 1 * 2^0 = 3_{10}$

Beispiel: Eingabemelder / 1-aus-*n*-Codierer (1)



Verhalten für *n* = 3: 3 Eingänge und 2 Ausgänge
- Höchstens einer der Eingänge darf 1 sein
- Ergebnis ist die Binärzahl, die die Nummer des Eingangs darstellt

Wahrheitstabelle

x_3	x_2	x_1	$f_2(x_1, x_2, x_3)$	$f_1(x_1, x_2, x_3)$
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	*	*
1	0	0	1	1
1	0	1	*	*
1	1	0	*	*
1	1	1	*	*

Ausgabe des 1-aus-3-Codierers ist Ergebnis zweier Schaltfunktionen f_2 und f_1.

Beispiel: Eingabemelder / 1-aus-*n*-Codierer (2)

Uninteressante Funktionswerte

– Einige Eingabewerte können nicht vorkommen / werden ausgeschlossen
– *„Don't care"* Ergebnisse
– Mit „*" in Wahrheitstabelle gekennzeichnet

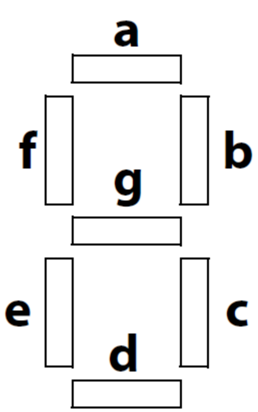
Bildung der KDNF für beide Schaltfunktionen

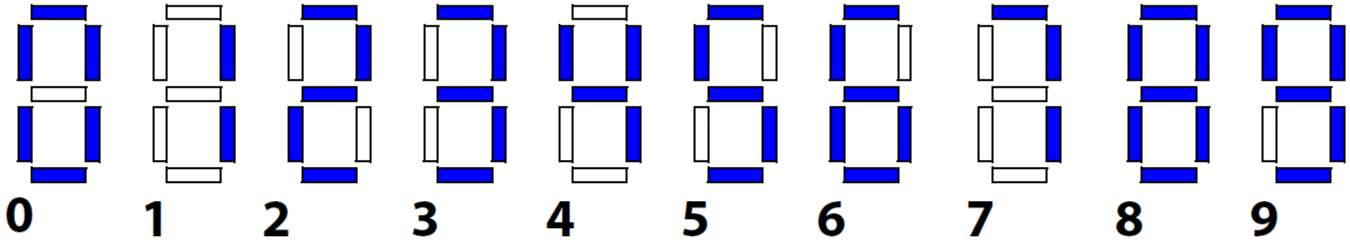
– $f_2(x_1, x_2, x_3) = \overline{x_1} * x_2 * \overline{x_3} + \overline{x_1} * \overline{x_2} * x_3$
– $f_1(x_1, x_2, x_3) = x_1 * \overline{x_2} * \overline{x_3} + \overline{x_1} * \overline{x_2} * x_3$

– Uninteressante Funktionsergebnisse werden nicht berücksichtigt (d.h. bei KDNF wie Null-Ergebnisse behandelt)

Beispiel: Siebensegmentanzeige (1)

Typische Anzeige für Ziffern (bspw. in Digitaluhren)





- Schaltfunktionen zur Ansteuerung der 7 verschiedenen Segmente
- Eingabe pro Schaltfunktion / Segment: Binär codierte Zahl bzw. Ziffer
- Ausgabe pro Schaltfunktion / Segment: Bit ob Segment an oder aus

Im folgenden gesucht: Schaltfunktion zur Ansteuerung von Segment d

Beispiel: Siebensegmentanzeige (2)

Wahrheitstabelle zur Ansteuerung des Segments d

x_3	x_2	x_1	x_0	$d(x_3, x_2, x_1, x_0)$
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	*
...				...
1	1	1	1	*

Beispiel: Siebensegmentanzeige (3)

Aufstellung der KDNF

- Nur 1-Werte betrachten
- *Don't care*-Werte werden ignoriert
- $d(x_3, x_2, x_1, x_0) = \overline{x_3} * \overline{x_2} * \overline{x_1} * \overline{x_0} + \overline{x_3} * \overline{x_2} * x_1 * \overline{x_0} + \overline{x_3} * \overline{x_2} * x_1 * x_0 +$
  $\overline{x_3} * x_2 * \overline{x_1} * x_0 + \overline{x_3} * x_2 * x_1 * \overline{x_0} + x_3 * \overline{x_2} * \overline{x_1} * \overline{x_0} +$
  $x_3 * \overline{x_2} * \overline{x_1} * x_0$

KDNF sicherlich nicht minimal

- Ungeeignet zur Übertragung in eine kostengünstige Schaltung

Äquivalenz von Schaltfunktionen

Wegen der Eindeutigkeit der KDNF- bzw. KKNF-Darstellung gilt

- Zwei Schaltfunktionen sind äquivalent, wenn sie sich auf die selbe KDNF oder KKNF zurückführen lassen
  - Bis auf Vertauschungen bzgl. des Kommutativitätsaxioms

Umformungen nach den Gesetzen der Booleschen Algebra

- Erhaltung der Schaltfunktion

Nutzen

- Z.B. Minimierung von Schaltfunktionen

Minimierung

Suche nach einer minimalen Darstellung einer Schaltfunktion

Größenbegriff notwendig

– Menge der benötigten Gatter
– Anzahl der Variablen
– Anzahl der notwendigen ICs
– Anzahl der notwendigen Kontakte
– Größe der benötigten Chipfläche in mm^2
– ...

☞ **Größenbegriff von den (Herstellungs-) Kosten bestimmt**

☞ **Größenbegriff hier**

– Anzahl der Booleschen Operationen

Grundlage der Minimierung (1)

Gesetze der Booleschen Algebra

– Insbesondere

$$A * B + A * \overline{B} = A$$

– Beweis

$A * B + A * \overline{B} = A * (B + \overline{B})$ wg. (Kommutativität u.) Distributivität

$A * (B + \overline{B}) = A * 1$ wg. komplementärem Element

$A * 1 = A$ wg. neutralem Element

Grundlage der Minimierung (2)

Beispiel: ODER-Funktion

– KDNF: $f(x_1, x_2) = \overline{x_1} * x_2 + x_1 * \overline{x_2} + x_1 * x_2$

– Umwandlung

$$
\begin{aligned}
f(x_1, x_2) &= \overline{x_1} * x_2 + x_1 * \overline{x_2} + x_1 * x_2 \\
&= \overline{x_1} * x_2 + x_1 * (\overline{x_2} + x_2) \\
&= \overline{x_1} * x_2 + x_1 * 1 \\
&= \overline{x_1} * x_2 + x_1 \\
&= \overline{x_1} * x_2 + x_1 + x_1 * x_2 \\
&= (\overline{x_1} + x_1) * x_2 + x_1 \\
&= x_2 + x_1
\end{aligned}
$$

– Distributivität

– Neutrales Element

– Absorption

Vorgehensweise

Manuelle Minimierung

– Umformen (z.B. der KDNF) nach den Regeln der Booleschen Algebra

Algorithmisches Verfahren

– Verfahren nach Quine/McCluskey
– Kann durch ein Programm automatisiert angewendet werden
– Geeignet für Schaltfunktionen mit vielen Variablen

Graphische Verfahren

– Händlerscher Kreisgraph
– Karnaugh-Veitch Diagramme
– Geeignet für Schaltfunktionen mit wenigen Variablen

Karnaugh-Veitch Diagramme (1)

Ausgangspunkt KDNF (oder KKNF)

- Rechteckschema
- Je ein Feld für jeden möglichen Minterm (Maxterm)
- Anordnung der Felder, so dass benachbarte Felder bzw. Minterme zusammenfassbar sind

Diagramm für zweistellige Schaltfunktion

- Funktion: $f(x_1, x_2)$
- KV-Diagramm

	$\overline{x_1}$	x_1
$\overline{x_2}$	$\overline{x_1} \cdot \overline{x_2}$	$x_1 \cdot \overline{x_2}$
x_2	$\overline{x_1} \cdot x_2$	$x_1 \cdot x_2$

Karnaugh-Veitch Diagramme (2)

Diagrammaufbau

– Jede Variable x_i halbiert das Diagramm in zwei zusammenhängende Teile
  – Erster Teil für $\overline{x_i}$
  – Zweiter Teil für x_i

– Variable x_1

	$\overline{x_1}$	x_1
$\overline{x_2}$	$\overline{x_1} \cdot \overline{x_2}$	$x_1 \cdot \overline{x_2}$
x_2	$\overline{x_1} \cdot x_2$	$x_1 \cdot x_2$

Variable x_2

	$\overline{x_1}$	x_1
$\overline{x_2}$	$\overline{x_1} \cdot \overline{x_2}$	$x_1 \cdot \overline{x_2}$
x_2	$\overline{x_1} \cdot x_2$	$x_1 \cdot x_2$

– Benachbarte Felder unterscheiden sich nur um das „Vorzeichen“ ***einer*** Variablen in den beiden Mintermen

Beispiel: ODER-Funktion (1)

Aufstellen der KDNF

– $f(x_1, x_2) = \overline{x_1} * x_2 + x_1 * \overline{x_2} + x_1 * x_2$

Eintragung in das Diagramm

– Eintragung einer 1, wenn Minterm benötigt wird

– Eintragung einer 0, wenn Minterm nicht benötigt wird

	$\overline{x_1}$	x_1
$\overline{x_2}$	0	1
x_2	1	1

☞ Eintragungen im KV-Diagramm auch direkt aus Wahrheitstabelle möglich

Beispiel: ODER-Funktion (2)

Markierung möglichst weniger und möglichst großer zusammenhängender Bereiche, die im KV-Diagramm 1en enthalten

- Nur zusammenhängende rechteckige Bereiche mit 2^n Elementen erlaubt
- Im KV-Diagramm markierte Bereiche dürfen sich durchaus überlappen
- Alle 1-Felder im KV-Diagramm müssen schließlich markiert sein

	$\overline{x_1}$	x_1
$\overline{x_2}$	0	1
x_2	1	1

– Markierte Bereiche ergeben Produktterme, die summiert werden:

$f(x_1, x_2) = x_1 + x_2$

  – Produktterme ergeben sich aus denjenigen Variablen, die entweder nur negiert oder nur nicht-negiert im markierten Bereich vorkommen

Beispiel: ODER-Funktion (3)

Alternative Markierung

– Markierung nicht so groß wie möglich, aber alle 1en markiert

	$\overline{x_1}$	x_1
$\overline{x_2}$	0	1
x_2	1	1

– Markierte Bereiche ergeben Produktterme, die summiert werden:

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

Funktion korrekt, jedoch nicht minimal

Beispiel: Eingabemelder / 1-aus-*n*-Codierer (1)

Dreistellige Schaltfunktionen

– Karnaugh-Veitch Diagramm

	$\overline{x_2}$	x_2		$\overline{x_2}$
	$\overline{x_1}$		x_1	
$\overline{x_3}$	$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3}$	$\overline{x_1} \cdot x_2 \cdot \overline{x_3}$	$x_1 \cdot x_2 \cdot \overline{x_3}$	$x_1 \cdot \overline{x_2} \cdot \overline{x_3}$
x_3	$\overline{x_1} \cdot \overline{x_2} \cdot x_3$	$\overline{x_1} \cdot x_2 \cdot x_3$	$x_1 \cdot x_2 \cdot x_3$	$x_1 \cdot \overline{x_2} \cdot x_3$

Beispiel: Eingabemelder / 1-aus-*n*-Codierer (2)

Halbierungen des Diagramms

– Variable x_1

$\overline{x_1}$	$\overline{x_1}$	x_1	x_1
$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3}$	$\overline{x_1} \cdot x_2 \cdot \overline{x_3}$	$x_1 \cdot x_2 \cdot \overline{x_3}$	$x_1 \cdot \overline{x_2} \cdot \overline{x_3}$
$\overline{x_1} \cdot \overline{x_2} \cdot x_3$	$\overline{x_1} \cdot x_2 \cdot x_3$	$x_1 \cdot x_2 \cdot x_3$	$x_1 \cdot \overline{x_2} \cdot x_3$

– Variable x_2

$\overline{x_2}$	x_2	x_2	$\overline{x_2}$
$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3}$	$\overline{x_1} \cdot x_2 \cdot \overline{x_3}$	$x_1 \cdot x_2 \cdot \overline{x_3}$	$x_1 \cdot \overline{x_2} \cdot \overline{x_3}$
$\overline{x_1} \cdot \overline{x_2} \cdot x_3$	$\overline{x_1} \cdot x_2 \cdot x_3$	$x_1 \cdot x_2 \cdot x_3$	$x_1 \cdot \overline{x_2} \cdot x_3$

– Wichtig: die Bereiche für $\overline{x_2}$ gehören zusammen

– Vorstellung: Diagramm ist an den Rändern zusammengeklebt

Beispiel: Eingabemelder / 1-aus-*n*-Codierer (3)

Halbierungen des Diagramms

– Variable x_3

$\overline{x_3}$	$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3}$	$\overline{x_1} \cdot x_2 \cdot \overline{x_3}$	$x_1 \cdot x_2 \cdot \overline{x_3}$	$x_1 \cdot \overline{x_2} \cdot \overline{x_3}$
x_3	$\overline{x_1} \cdot \overline{x_2} \cdot x_3$	$\overline{x_1} \cdot x_2 \cdot x_3$	$x_1 \cdot x_2 \cdot x_3$	$x_1 \cdot \overline{x_2} \cdot x_3$

Beispiel: Eingabemelder / 1-aus-*n*-Codierer (4)

Belegen des KV-Diagramms aus der Wahrheitstabelle

– Funktion f_2 aus Folie 44

– Eintragung der *Don’t care*-Werte

	$\overline{x_2}$	x_2	x_2	$\overline{x_2}$
	$\overline{x_1}$	$\overline{x_1}$	x_1	x_1
$\overline{x_3}$	0	1	d	0
x_3	1	d	d	d

– *Don’t care*-Werte können mitmarkiert werden oder nicht

– Ziel: möglichst große Bereiche markieren

– Markierte *Don’t care*-Werte werden später zu 1, andere zu 0

Beispiel: Eingabemelder / 1-aus-*n*-Codierer (5)

Markierungen für f_2

– Zwei Bereiche

	$\overline{x_2}$	x_2	x_2	$\overline{x_2}$
	$\overline{x_1}$	$\overline{x_1}$	x_1	x_1
$\overline{x_3}$	0	1	d	0
x_3	1	d	d	d

– Markierte Bereiche ergeben Produktterme, die summiert werden:

$f_2(x_1, x_2, x_3) = x_2 + x_3$

Beispiel: Eine fiktive Funktion

Gegeben sei folgende Belegung aus der Wahrheitstabelle

– Gesucht ist die beste Markierung

	$\overline{x_2}$, $\overline{x_1}$	x_2, $\overline{x_1}$	x_2, x_1	$\overline{x_2}$, x_1
$\overline{x_3}$	1	0	d	1
x_3	1	1	d	d

– Markierte Bereiche ergeben Produktterme, die summiert werden:

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

Beispiel: Eine andere fiktive Funktion (1)

Gegeben sei eine weitere Belegung aus einer Wahrheitstabelle

– Gesucht ist die beste Markierung

	$\overline{x_2}$	x_2	x_2	$\overline{x_2}$
	$\overline{x_1}$	$\overline{x_1}$	x_1	x_1
$\overline{x_3}$	1	0	0	1
x_3	0	0	1	1

– Frage: minimale DNF?

  – DNF: $f(x_1, x_2, x_3) = \overline{x_1} * \overline{x_2} * \overline{x_3} + x_1 * x_3 + x_1 * \overline{x_2}$

  – Sind die größten Bereiche markiert?

Beispiel: Eine andere fiktive Funktion (2)

Gegeben sei eine weitere Belegung aus einer Wahrheitstabelle

– Gesucht ist die beste Markierung

	$\bar{x}_2$	x_2	x_2	$\bar{x}_2$
	$\bar{x}_1$	$\bar{x}_1$	x_1	x_1
$\bar{x}_3$	1	0	0	1
x_3	0	0	1	1

– Markierung kann vergrößert werden

  – DNF: $f(x_1, x_2, x_3) = \bar{x}_2 * \bar{x}_3 + x_1 * x_3 + x_1 * \bar{x}_2$

– Frage: DNF minimal?

  – Minimale Anzahl von Markierungen?

  ☞ Überflüssige Markierung kann entfallen

Beispiel: Eine andere fiktive Funktion (3)

Gegeben sei eine weitere Belegung aus einer Wahrheitstabelle

– Gesucht ist die beste Markierung

	$\overline{x_2}$	x_2	x_2	$\overline{x_2}$
	$\overline{x_1}$	$\overline{x_1}$	x_1	x_1
$\overline{x_3}$	1	0	0	1
x_3	0	0	1	1

– Minimale DNF gefunden

  – DNF: $f(x_1, x_2, x_3) = \overline{x_2} * \overline{x_3} + x_1 * x_3$

Vierstellige Funktionen (1)

Karnaugh-Veitch Diagramm für vierstellige Schaltfunktion

		$\overline{x_2}$	x_2	x_2	$\overline{x_2}$
		$\overline{x_1}$	$\overline{x_1}$	x_1	x_1
$\overline{x_4}$	$\overline{x_3}$	$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot \overline{x_4}$	$\overline{x_1} \cdot x_2 \cdot \overline{x_3} \cdot \overline{x_4}$	$x_1 \cdot x_2 \cdot \overline{x_3} \cdot \overline{x_4}$	$x_1 \cdot \overline{x_2} \cdot \overline{x_3} \cdot \overline{x_4}$
x_4	$\overline{x_3}$	$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot x_4$	$\overline{x_1} \cdot x_2 \cdot \overline{x_3} \cdot x_4$	$x_1 \cdot x_2 \cdot \overline{x_3} \cdot x_4$	$x_1 \cdot \overline{x_2} \cdot \overline{x_3} \cdot x_4$
x_4	x_3	$\overline{x_1} \cdot \overline{x_2} \cdot x_3 \cdot x_4$	$\overline{x_1} \cdot x_2 \cdot x_3 \cdot x_4$	$x_1 \cdot x_2 \cdot x_3 \cdot x_4$	$x_1 \cdot \overline{x_2} \cdot x_3 \cdot x_4$
$\overline{x_4}$	x_3	$\overline{x_1} \cdot \overline{x_2} \cdot x_3 \cdot \overline{x_4}$	$\overline{x_1} \cdot x_2 \cdot x_3 \cdot \overline{x_4}$	$x_1 \cdot x_2 \cdot x_3 \cdot \overline{x_4}$	$x_1 \cdot \overline{x_2} \cdot x_3 \cdot \overline{x_4}$

Vierstellige Funktionen (2)

Halbierungen für vierstellige Schaltfunktion

$\overline{x_1}$		x_1	
$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot \overline{x_4}$	$\overline{x_1} \cdot x_2 \cdot \overline{x_3} \cdot \overline{x_4}$	$x_1 \cdot x_2 \cdot \overline{x_3} \cdot \overline{x_4}$	$x_1 \cdot \overline{x_2} \cdot \overline{x_3} \cdot \overline{x_4}$
$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot x_4$	$\overline{x_1} \cdot x_2 \cdot \overline{x_3} \cdot x_4$	$x_1 \cdot x_2 \cdot \overline{x_3} \cdot x_4$	$x_1 \cdot \overline{x_2} \cdot \overline{x_3} \cdot x_4$
$\overline{x_1} \cdot \overline{x_2} \cdot x_3 \cdot x_4$	$\overline{x_1} \cdot x_2 \cdot x_3 \cdot x_4$	$x_1 \cdot x_2 \cdot x_3 \cdot x_4$	$x_1 \cdot \overline{x_2} \cdot x_3 \cdot x_4$
$\overline{x_1} \cdot \overline{x_2} \cdot x_3 \cdot \overline{x_4}$	$\overline{x_1} \cdot x_2 \cdot x_3 \cdot \overline{x_4}$	$x_1 \cdot x_2 \cdot x_3 \cdot \overline{x_4}$	$x_1 \cdot \overline{x_2} \cdot x_3 \cdot \overline{x_4}$

$\overline{x_2}$	x_2		$\overline{x_2}$
$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot \overline{x_4}$	$\overline{x_1} \cdot x_2 \cdot \overline{x_3} \cdot \overline{x_4}$	$x_1 \cdot x_2 \cdot \overline{x_3} \cdot \overline{x_4}$	$x_1 \cdot \overline{x_2} \cdot \overline{x_3} \cdot \overline{x_4}$
$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot x_4$	$\overline{x_1} \cdot x_2 \cdot \overline{x_3} \cdot x_4$	$x_1 \cdot x_2 \cdot \overline{x_3} \cdot x_4$	$x_1 \cdot \overline{x_2} \cdot \overline{x_3} \cdot x_4$
$\overline{x_1} \cdot \overline{x_2} \cdot x_3 \cdot x_4$	$\overline{x_1} \cdot x_2 \cdot x_3 \cdot x_4$	$x_1 \cdot x_2 \cdot x_3 \cdot x_4$	$x_1 \cdot \overline{x_2} \cdot x_3 \cdot x_4$
$\overline{x_1} \cdot \overline{x_2} \cdot x_3 \cdot \overline{x_4}$	$\overline{x_1} \cdot x_2 \cdot x_3 \cdot \overline{x_4}$	$x_1 \cdot x_2 \cdot x_3 \cdot \overline{x_4}$	$x_1 \cdot \overline{x_2} \cdot x_3 \cdot \overline{x_4}$

Vierstellige Funktionen (3)

Halbierungen für vierstellige Schaltfunktion

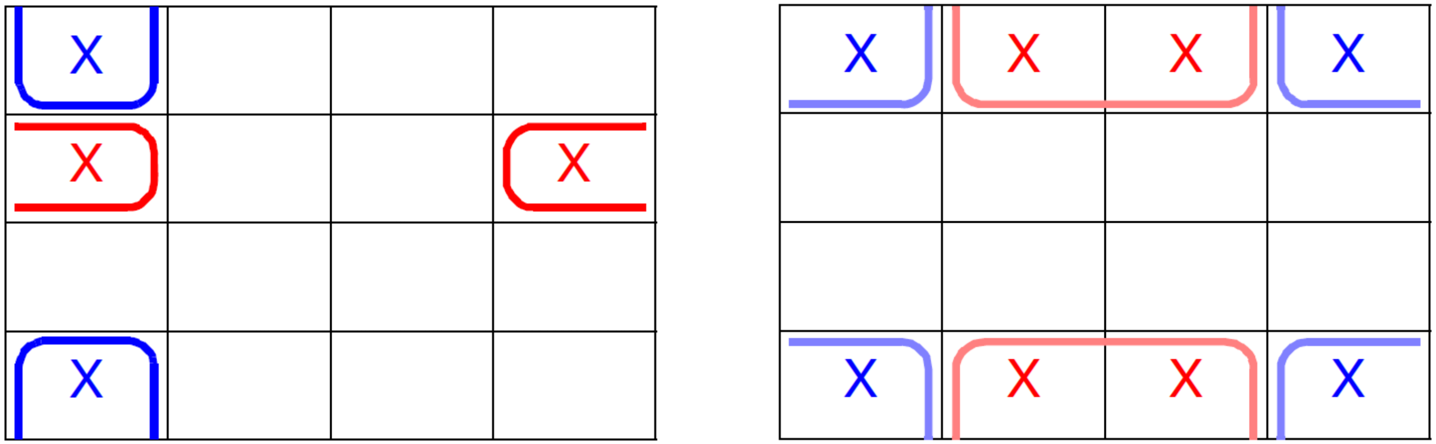
$\overline{x_3}$	$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot \overline{x_4}$	$\overline{x_1} \cdot x_2 \cdot \overline{x_3} \cdot \overline{x_4}$	$x_1 \cdot x_2 \cdot \overline{x_3} \cdot \overline{x_4}$	$x_1 \cdot \overline{x_2} \cdot \overline{x_3} \cdot \overline{x_4}$
	$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot x_4$	$\overline{x_1} \cdot x_2 \cdot \overline{x_3} \cdot x_4$	$x_1 \cdot x_2 \cdot \overline{x_3} \cdot x_4$	$x_1 \cdot \overline{x_2} \cdot \overline{x_3} \cdot x_4$
x_3	$\overline{x_1} \cdot \overline{x_2} \cdot x_3 \cdot x_4$	$\overline{x_1} \cdot x_2 \cdot x_3 \cdot x_4$	$x_1 \cdot x_2 \cdot x_3 \cdot x_4$	$x_1 \cdot \overline{x_2} \cdot x_3 \cdot x_4$
	$\overline{x_1} \cdot \overline{x_2} \cdot x_3 \cdot \overline{x_4}$	$\overline{x_1} \cdot x_2 \cdot x_3 \cdot \overline{x_4}$	$x_1 \cdot x_2 \cdot x_3 \cdot \overline{x_4}$	$x_1 \cdot \overline{x_2} \cdot x_3 \cdot \overline{x_4}$

$\overline{x_4}$	$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot \overline{x_4}$	$\overline{x_1} \cdot x_2 \cdot \overline{x_3} \cdot \overline{x_4}$	$x_1 \cdot x_2 \cdot \overline{x_3} \cdot \overline{x_4}$	$x_1 \cdot \overline{x_2} \cdot \overline{x_3} \cdot \overline{x_4}$
x_4	$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot x_4$	$\overline{x_1} \cdot x_2 \cdot \overline{x_3} \cdot x_4$	$x_1 \cdot x_2 \cdot \overline{x_3} \cdot x_4$	$x_1 \cdot \overline{x_2} \cdot \overline{x_3} \cdot x_4$
	$\overline{x_1} \cdot \overline{x_2} \cdot x_3 \cdot x_4$	$\overline{x_1} \cdot x_2 \cdot x_3 \cdot x_4$	$x_1 \cdot x_2 \cdot x_3 \cdot x_4$	$x_1 \cdot \overline{x_2} \cdot x_3 \cdot x_4$
$\overline{x_4}$	$\overline{x_1} \cdot \overline{x_2} \cdot x_3 \cdot \overline{x_4}$	$\overline{x_1} \cdot x_2 \cdot x_3 \cdot \overline{x_4}$	$x_1 \cdot x_2 \cdot x_3 \cdot \overline{x_4}$	$x_1 \cdot \overline{x_2} \cdot x_3 \cdot \overline{x_4}$

Vierstellige Funktionen (4)

Markierungen

– Insbesondere sind folgende Markierungen möglich



– Vorstellung: Diagramm ist an den Seiten jeweils zusammengeklebt

Beispiel: 2x2-Multiplizierer (1)

Binärer Multiplizierer für 2 mal 2 Eingänge

- Binärdarstellung von Zahlen von 0 bis 3 bzw. 0 bis 15
- Zwei Eingänge a_1 und a_0
- Zwei Eingänge b_1 und b_0
- Vier Ausgänge y_3, y_2, y_1 und y_0

Beispiel: 2x2-Multiplizierer (2)

	$a_1 = x_1$	$a_0 = x_2$	$b_1 = x_3$	$b_0 = x_4$	y_3	y_2	y_1	y_0
0 x 0 = 0	0	0	0	0	0	0	0	0
0 x 1 = 0	0	0	0	1	0	0	0	0
0 x 2 = 0	0	0	1	0	0	0	0	0
0 x 3 = 0	0	0	1	1	0	0	0	0
1 x 0 = 0	0	1	0	0	0	0	0	0
1 x 1 = 1	0	1	0	1	0	0	0	1
1 x 2 = 2	0	1	1	0	0	0	1	0
1 x 3 = 3	0	1	1	1	0	0	1	1
2 x 0 = 0	1	0	0	0	0	0	0	0
2 x 1 = 2	1	0	0	1	0	0	1	0
2 x 2 = 4	1	0	1	0	0	1	0	0
2 x 3 = 6	1	0	1	1	0	1	1	0
3 x 0 = 0	1	1	0	0	0	0	0	0
3 x 1 = 3	1	1	0	1	0	0	1	1
3 x 2 = 6	1	1	1	0	0	1	1	0
3 x 3 = 9	1	1	1	1	1	0	0	1

Beispiel: 2x2-Multiplizierer (3)

Karnaugh-Veitch Diagramm für y_0

	$\overline{x_2}$, $\overline{x_1}$	x_2, $\overline{x_1}$	x_2, x_1	$\overline{x_2}$, x_1
$\overline{x_4}$, $\overline{x_3}$	0	0	0	0
x_4, $\overline{x_3}$	0	1	1	0
x_4, x_3	0	1	1	0
$\overline{x_4}$, x_3	0	0	0	0

– Markierte Bereiche:

$y_0 = x_2 * x_4$

Beispiel: 2x2-Multiplizierer (4)

Karnaugh-Veitch Diagramm für y_1

		$\overline{x_2}$	x_2	x_2	$\overline{x_2}$
		$\overline{x_1}$	$\overline{x_1}$	x_1	x_1
$\overline{x_4}$	$\overline{x_3}$	0	0	0	0
x_4	$\overline{x_3}$	0	0	1	1
x_4	x_3	0	1	0	1
$\overline{x_4}$	x_3	0	1	1	0

– Markierte Bereiche:

$$y_1 = x_2 * x_3 * \overline{x_4} + \overline{x_1} * x_2 * x_3 + x_1 * \overline{x_3} * x_4 + x_1 * \overline{x_2} * x_4$$

Beispiel: 2x2-Multiplizierer (5)

Karnaugh-Veitch Diagramm für y_2

		$\overline{x_2}$ / $\overline{x_1}$	x_2 / $\overline{x_1}$	x_2 / x_1	$\overline{x_2}$ / x_1
$\overline{x_4}$	$\overline{x_3}$	0	0	0	0
x_4	$\overline{x_3}$	0	0	0	0
x_4	x_3	0	0	0	1
$\overline{x_4}$	x_3	0	0	1	1

– Markierte Bereiche:

$$y_2 = x_1 * \overline{x_2} * x_3 + x_1 * x_3 * \overline{x_4}$$

Beispiel: 2x2-Multiplizierer (6)

Karnaugh-Veitch Diagramm für y_3

		$\overline{x_2}$, $\overline{x_1}$	x_2, $\overline{x_1}$	x_2, x_1	$\overline{x_2}$, x_1
$\overline{x_4}$	$\overline{x_3}$	0	0	0	0
x_4	$\overline{x_3}$	0	0	0	0
x_4	x_3	0	0	$\boxed{1}$	0
$\overline{x_4}$	x_3	0	0	0	0

– Markierte Bereiche:

$y_3 = x_1 * x_2 * x_3 * x_4$

Zusammenfassung

Markierungsregeln

- Rechteckige Bereiche mit 2^n Elementen markieren
  - Achtung: Diagramm gilt als oben/unten bzw. links/rechts „zusammengeklebt“
- Alle 1-Werte müssen markiert werden
- Möglichst große Bereiche markieren
- Möglichst wenige markierte Bereiche benutzen

Roter Faden

2. Kombinatorische Logik

- Einleitung
- Schaltfunktionen
- Synthese von Schaltungen
  - Beispiele
    - ODER-Funktion, Eingabemelder, Siebensegmentanzeige
  - Äquivalenz von Schaltfunktionen
  - Minimierung von Schaltungen
    - Grundlagen
    - Vorgehensweise
    - Karnaugh-Veitch Diagramme
    - Beispiele
- Schaltnetze / *Combinational Networks*

Schaltnetze (1)

Mehrere Schaltfunktionen *(Combinational Networks)*

– Sind von gleichen Eingangsvariablen abhängig

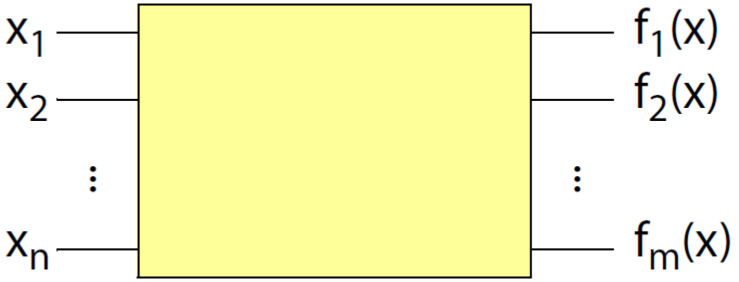
$f_1(x_1, x_2, \ldots, x_n)$

$f_2(x_1, x_2, \ldots, x_n)$

…

$f_m(x_1, x_2, \ldots, x_n)$

– Entspricht Schaltung mit mehreren Ausgängen



☞ Kombinatorische Logik

Schaltnetze (2)

Gerichteter, azyklischer Graph

– Gatter sowie Ein- und Ausgänge sind Knoten
– Verbindungsleitungen sind Kanten (gerichtet von Eingängen zu Ausgängen)

Aufbau von Schaltnetzen

– Einstufige (nur eine Gatterebene)
– Zweistufige (zwei Gatterebenen)
– Mehrstufige

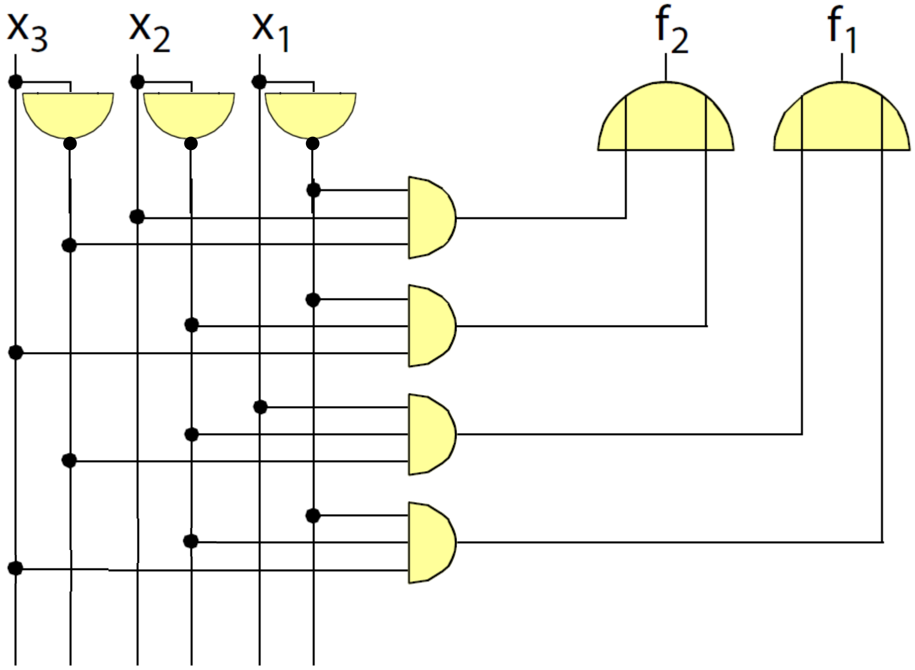
Folgerung aus der Darstellung durch kanonische Normalformen

☞ Jedes Schaltnetz ist zweistufig realisierbar, wenn
  – alle Signale einfach und negiert vorliegen, und
  – Gatter mit ausreichender Anzahl von Eingängen vorliegen

Schaltnetze (3)

Begründung

– Bezug zur KDNF (oder KKNF)
– Alle Variablen werden einfach oder negiert benutzt
– Zunächst Minterme: ein UND-Gatter pro Minterm (erste Stufe)
– Summe der Minterme: ein ODER-Gatter für alle Minterme



Beispiel
Eingabemelder

Schaltnetze (4)

Anzahl der notwendigen Gatter bei *n* Eingängen

- Max. 2^n UND-Gatter pro Schaltfunktion mit bis zu *n* Eingängen (KDNF)
- Ein ODER-Gatter mit bis zu 2^n Eingängen

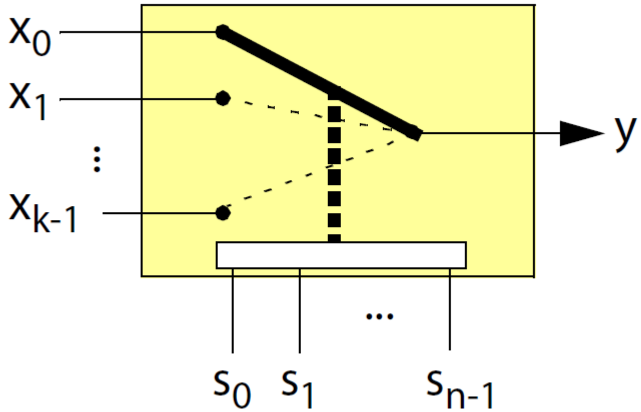
Minimierung

- Reduziert Gatteranzahl und Eingangsanzahl pro Gatter
- Minimierung parallel für mehrere Schaltfunktionen des Schaltnetzes
  - Verwendung der selben Gatter für verschiedene Schaltfunktionen
- Z.B. Karnaugh-Veitch Diagramme für mehrere Schaltfunktionen des Netzes

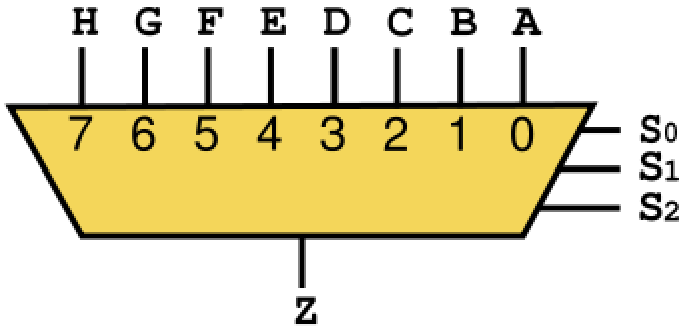
Typische Schaltnetze: 1-aus-*k*-Multiplexer (1)

Steuerleitungen weisen viele Eingabeleitungen einem Ausgang zu

- n Steuerleitungen $s_0, s_1, \ldots, s_{n\text{-}1}$ (Eingänge)
- $k = 2^n$ Eingänge $x_0, x_1, \ldots, x_{k\text{-}1}$
- Ein Ausgang y
- Es gilt: $y = x_i$ für $(s_{n\text{-}1}, \ldots, s_1, s_0)_2 = i$ (Zahlendarstellung im Binärsystem)



Multiplexer (MUX)



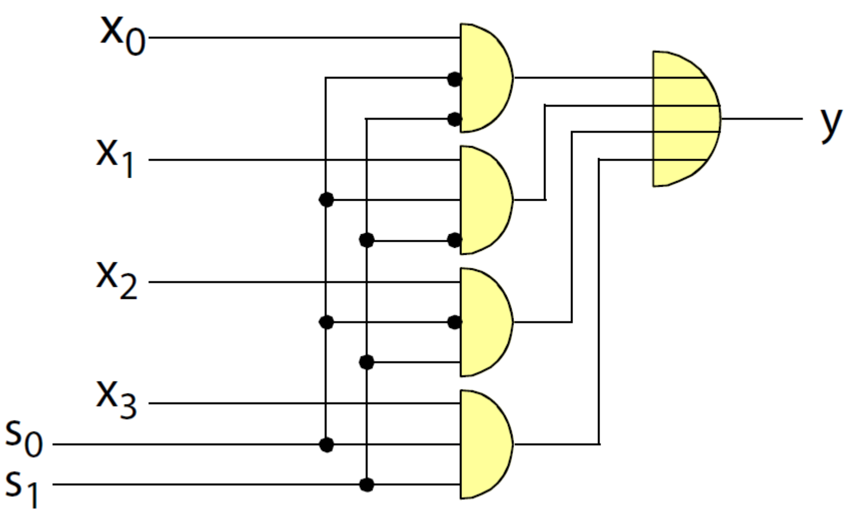
[en.wikipedia.org]

Typische Schaltnetze: 1-aus-*k*-Multiplexer (2)

Realisierung

– Für *n* = 2 als DNF

$$y = \overline{s_1} \cdot \overline{s_0} \cdot x_0 + \overline{s_1} \cdot s_0 \cdot x_1 + s_1 \cdot \overline{s_0} \cdot x_2 + s_1 \cdot s_0 \cdot x_3$$



Hinweis: Ein dicker Punkt an einem Gatter-*Ein*gang x steht für das Inverse, d.h. $\overline{x}$

Einsatz

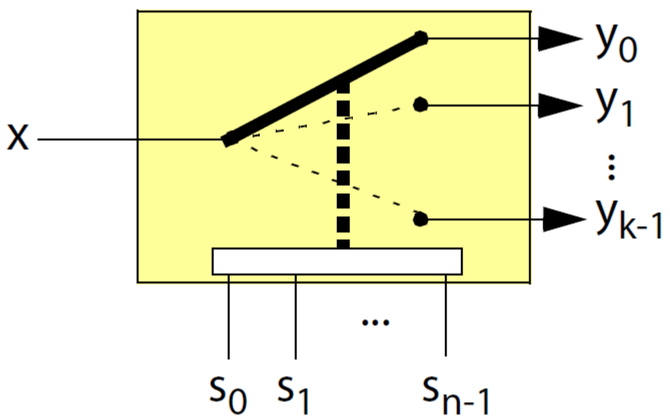
– Anzeige und Auswahl verschiedener Datenquellen

Z.B. Auslesen von Daten aus Speicherzellen

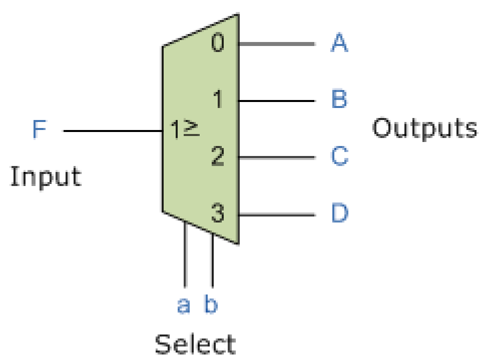
Typische Schaltnetze: 1-zu-*k*-Demultiplexer (1)

Steuerleitungen weisen eine Eingabeleitung vielen Ausgängen zu

- n Steuerleitungen $s_0, s_1, \ldots, s_{n\text{-}1}$ (Eingänge)
- Ein Eingang x
- $k = 2^n$ Ausgänge $y_0, y_1, \ldots, y_{k\text{-}1}$
- Es gilt: $y_i = x$ für $(s_{n\text{-}1}, \ldots, s_1, s_0)_2 = i$ (Zahlendarstellung im Binärsystem)



Demultiplexer (DEMUX)



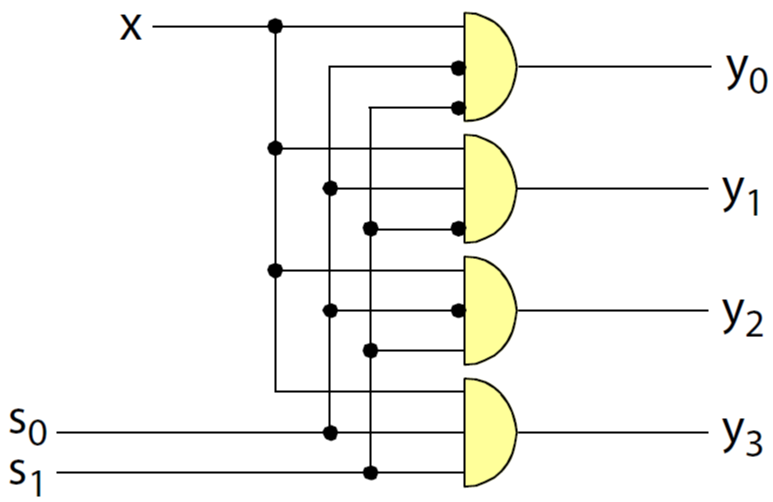
[electronics-tutorials.ws]

Typische Schaltnetze: 1-zu-*k*-Demultiplexer (2)

Realisierung

– Für n = 2 als DNF

$$y_0 = \overline{s_1} \cdot \overline{s_0} \cdot x,\; y_1 = \overline{s_1} \cdot s_0 \cdot x,\; y_2 = s_1 \cdot \overline{s_0} \cdot x,\; y_3 = s_1 \cdot s_0 \cdot x$$



Einsatz

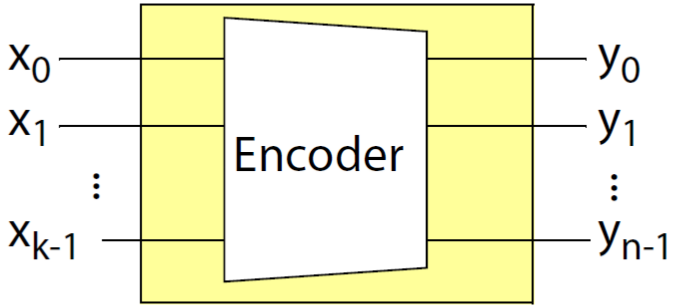
– Zuordnung und Auswahl verschiedener Datensenken

Z.B. Speichern von Daten in Speicherzellen

Typische Schaltnetze: *k*-zu-*n*-Codierer (1)

Nummer eines Eingangs wird ausgegeben

- $k = 2^n$ Eingänge $x_0, x_1, \ldots, x_{k-1}$
- Immer genau eine Eingangsleitung auf 1:
  $\exists\, i$ mit $x_i = 1$ und $\forall\, j \neq i$: $x_j = 0$
- n Ausgänge $y_0, y_1, \ldots, y_{n-1}$
- Es gilt: $(y_{n-1}, \ldots, y_1, y_0)_2 = i$
  (Zahlendarstellung im Binärsystem)



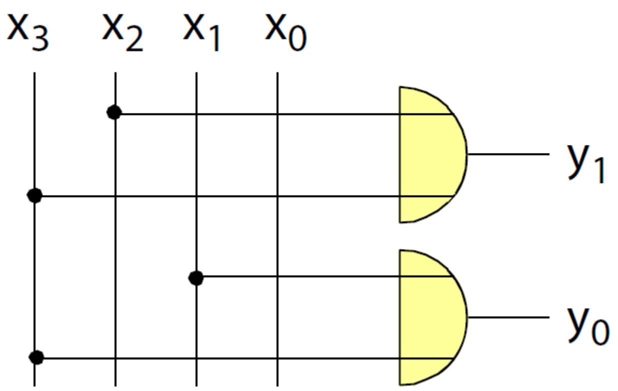
Codierer

Typische Schaltnetze: *k*-zu-*n*-Codierer (2)

Realisierung

– Für n = 2, k = 4 als DNF

$$y_0 = x_1 + x_3, \quad y_1 = x_2 + x_3$$



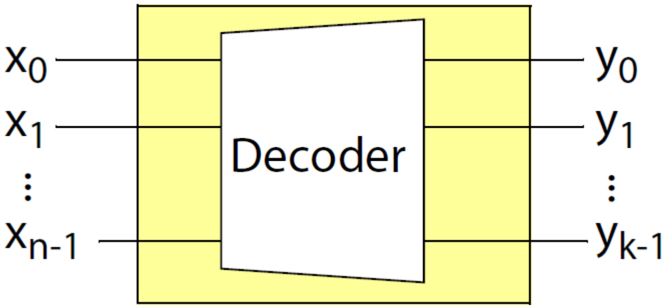
Einsatz

– Z.B. Signalisierung eines *Interrupt*-Eingangs

Typische Schaltnetze: *n*-zu-*k*-Decodierer (1)

Eingänge selektieren genau einen von vielen Ausgängen

– n Eingänge $x_0, x_1, \ldots, x_{n\text{-}1}$
– $k = 2^n$ Ausgänge $y_0, y_1, \ldots, y_{k\text{-}1}$
– Es gilt: $y_i = 1$ und $\forall\, j \neq i$: $y_j = 0$ mit $(x_{n\text{-}1}, \ldots, x_1, x_0)_2 = i$ (Zahlendarstellung im Binärsystem)



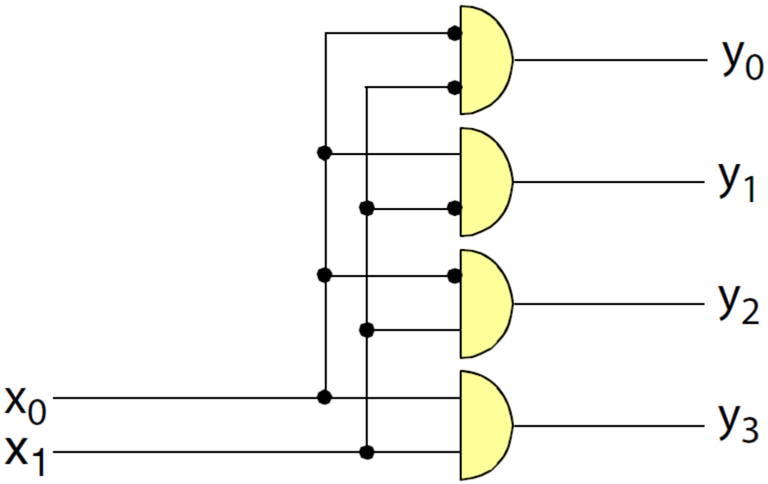
Decodierer

Typische Schaltnetze: *n*-zu-*k*-Decodierer (2)

Realisierung

– Für $n = 2$, $k = 4$ als DNF

$$y_0 = \overline{x_0} \cdot \overline{x_1},\quad y_1 = x_0 \cdot \overline{x_1},\quad y_2 = \overline{x_0} \cdot x_1,\quad y_3 = x_0 \cdot x_1$$



Einsatz

– Z.B. Dekodierung eines Maschinenbefehls

Zusammenfassung (1)

Einleitung

- Elektrische Zustände einer Schaltung
  - Strom fließt oder fließt nicht
  - Ladung ist vorhanden oder nicht
- Logische Zustände
  - wahr oder falsch
  - 1 oder 0
- Realisierung digitaler logischer Schaltungen mit elementaren Bausteinen: UND-, ODER und NICHT-Gatter
- Direkter Zusammenhang zwischen digitalen logischen Schaltungen und Boolescher Algebra: Rechenregeln der Booleschen Algebra können auf logische Schaltungen angewendet werden

Zusammenfassung (2)

Schaltfunktionen

- Wahrheitstabellen: Tabellarische Zuordnung von logischen Ausgabewerten zu allen möglichen Kombinationen von Eingabewerten
- Normalformen: Normierte Ausdrücke zur Darstellung von Schaltfunktionen
  - Kanonische disjunktive Normalform (KDNF): Disjunktion einer Menge von Mintermen
  - Kanonische konjunktive Normalform (KKNF): Konjunktion einer Menge von Maxtermen
- Hauptsatz der Schaltalgebra: Jede Schaltfunktion lässt sich als genau eine KDNF bzw. KKNF darstellen; diese Darstellung ist eindeutig

Zusammenfassung (3)

Synthese von Schaltungen

- Vorgehensweise: Aufstellen der Wahrheitstabelle, Bilden der KDNF/KKNF, Aufbau der Schaltung
- Schaltungs-Minimierung: Suche nach einer minimalen Darstellung einer Schaltfunktion
- Graphisches Verfahren: Karnaugh-Veitch Diagramme
  - Rechteckschema, je ein Feld für jeden möglichen Minterm (Maxterm)
  - Anordnung, so dass benachbarte Felder zusammenfassbar
  - Markierung möglichst weniger und möglichst großer zusammenhängender Bereiche mit Einsen im KV-Diagramm
  - Überführung der markierten Bereiche in Schaltung
  - Unbestimmte Werte (*Don't care*) werden mitmarkiert oder nicht

Zusammenfassung (4)

Schaltnetze / *Combinational Networks*

- Schaltnetz: Schaltung mit mehreren Ausgängen; mehrere Schaltfunktionen hängen von gleichen Eingangsvariablen ab
- Jedes Schaltnetz ist zweistufig realisierbar
- Typische Schaltnetze
  - 1-aus-*k*-Multiplexer
  - 1-zu-*k*-Demultiplexer
  - *k*-zu-*n*-Codierer
  - *n*-zu-*k*-Decodierer