Vortrag für Grundlagen der technischen Informatik

Thema: Schaltnetze

Ein Vortrag von:

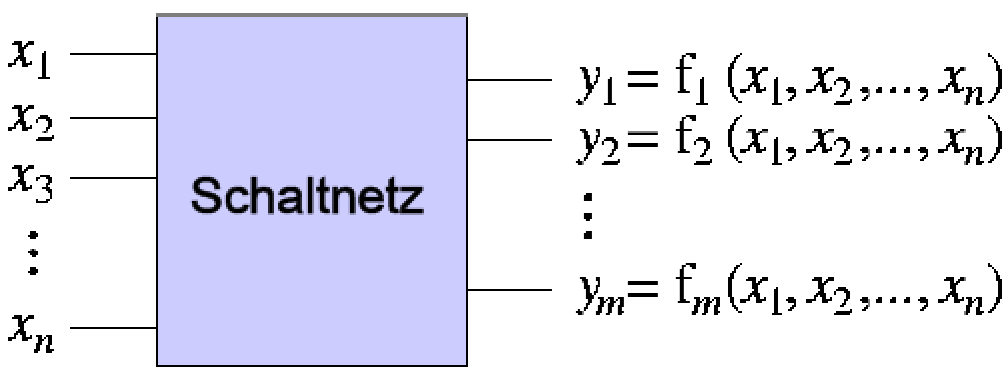
Benjamin Krause
Max Josef Ender
Alexander Dockhorn
Jannis Becke

Inhalt

- Schaltnetze
- Dekodierer
- Kodierer
- Demultiplexer
- Multiplexer
- Übungsaufgaben

Schaltnetze allgemein

- Ein **Schaltnetz** ist eine schaltungstechnische Realisierung einer Schaltfunktion
  $f: \{0,1\}^n \rightarrow \{0,1\}^m$ mit $n, m \geq 1$
- f ist zerlegbar in m Boolesche Funktionen mit den gleichen n Eingangsvariablen :
  $f_1(x_1, x_2, \ldots, x_n), \quad f_2(x_1, x_2, \ldots, x_n), \ldots, \quad f_m(x_1, x_2, \ldots, x_n)$

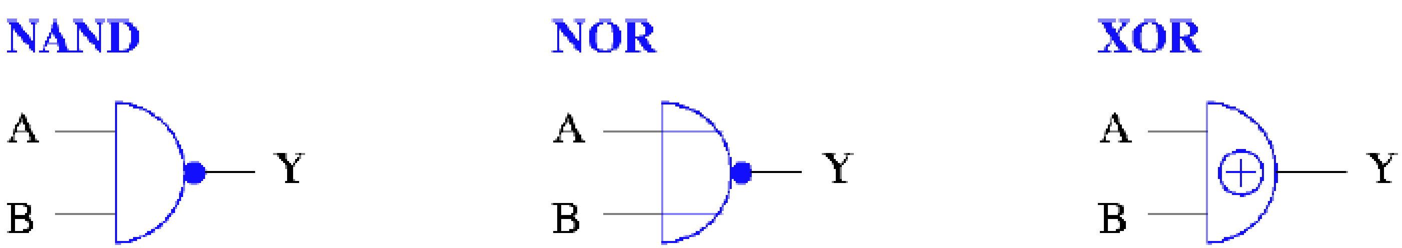


Schaltnetze bezeichnet man auch als **kombinatorische Logik**.

Schaltnetze allgemein

- es existieren einstufige (eine Gatterebene), zweistufige (zwei Gatterebenen) aber auch mehrstufige Schaltnetze
- jedes Schaltnetz:
  - ist als **gerichteter, azyklischer Graph** darstellbar
  - kann durch ein zweistufiges Schaltnetz realisiert werden
  - kann nur aus NAND- und NOR-Gattern aufgebaut werden

Gängige Gatter-Bausteine:



=> Stellen Sie ein OR durch ein NAND dar!

zweistufige Schaltnetze

- jede Schaltfunktion kann als **KDNF** bzw. **KKNF** formuliert werden
- bei Realisierung einer **KDNF** werden benötigt:
  - je Minterm ein UND-Gatter
  - ein ODER-Gatter zur Disjunktion aller Minterme
- bei Realisierung einer **KKNF** werden benötigt:
  - je Maxterm ein ODER-Gatter
  - ein UND-Gatter zur Konjunktion aller Maxterme
- Minimierung der Booleschen Funktionen zur **DNF** bzw. **KNF** reduziert Anzahl und Größe der benötigten Gatter signifikant

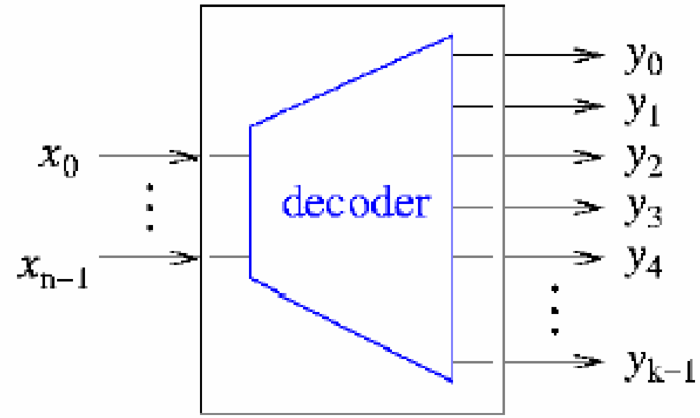
zweistufige Schaltnetze

- aus der Darstellung als KNF resultiert, dass **jede** Schaltfunktion $f\,(x_1, x_2, \ldots, x_n)$ durch ein **zweistufiges Schaltnetz** realisierbar ist, wenn:
  - alle Eingangssignale x_i sowohl einfach als auch negiert vorliegen
  - Gatter mit einer ausreichenden Größe zur Verfügung stehen

=> Jede Funktion kann durch zweistufiges Schaltnetz realisiert werden!

n-zu-k Dekodierer

- **n-zu-k Dekodierer:**
  - n Eingänge $x_0, x_1, \ldots, x_{n-1}$
  - $k = 2^n$ Ausgänge $y_0, y_1, \ldots, y_{k-1}$
  - wenn $(x_{n-1}, \ldots, x_1, x_0)_2 = i$, dann ist $y_i = 1$ und $y_{j \neq i} = 0$



Für jede Eingangskombination wird **genau ein** Ausgang aktiviert.

Zum Beispiel benötigt für Balkenanzeigen (Tankfüllanzeige, Akkustand, …)

Bsp. zu Dekodierern

2-zu-4 Dekodierer:

Eingänge		Ausgang			
x_0	x_1	y_0	y_1	y_2	y_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

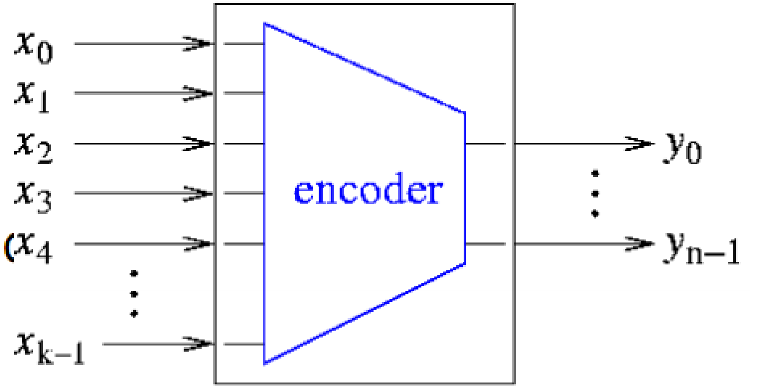
$$y_0 = \neg x_0 \wedge \neg x_1$$
$$y_1 = \neg x_0 \wedge x_1$$
$$y_2 = x_0 \wedge \neg x_1$$
$$y_3 = x_0 \wedge x_1$$

Wie man leicht sieht:
Für jede Variante von x_0 und x_1 wird genau **ein** bestimmtes $\mathbf{y}_n$ ausgewählt.

k-zu-n Kodierer

- **k-zu-n Kodierer:**
  - n Ausgänge $y_0, y_1, \ldots, y_{n-1}$
  - $k = 2^n$ Eingänge $x_0, x_1, \ldots, x_{k-1}$
  - nur genau eine Eingangsleitung $x_i = 1, x_{j \neq i} = 0$

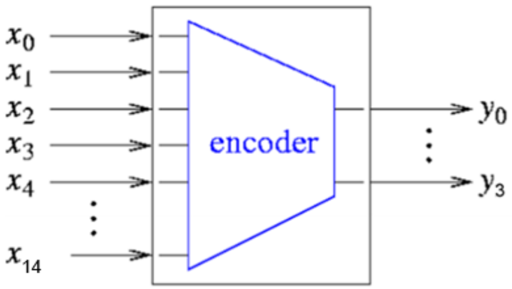
$(y_{n-1}, \ldots, y_1, y_0)_2 = i$



Jeder Eingangsleitung wird genau **eine Kombination** der Ausgangsleitungen zugeordnet.

Bsp. zu Kodierern

Kodieren der Tastenanschläge eines einfachen Taschenrechners:



Tasten 0-9 = 10 Eingänge
+ die Tasten „+“ , „-“ , „*“ , „/“ , „=“ = 5 Eingänge
insgesamt = 15 Eingänge

16 Binärzahlen reichen, um Tastendrücke darzustellen, da gleichzeitige Eingaben irrelevant sind.
Wir benötigen mindestens 4 Stellen um 16 Binärzahlen darzustellen ($2^4 = 16$).

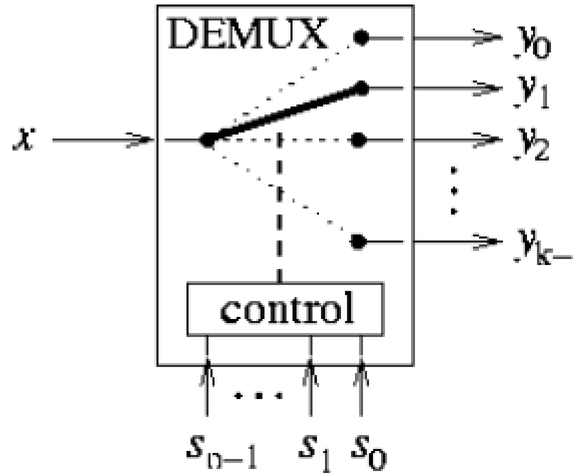
Auszug aus der Wertetabelle:

x0	x1	x2	x3	x4	...	x15	y1	y2	y3	y4
1	0	0	0	0	...	0	0	0	0	0
0	1	0	0	0	...	0	0	0	0	1
0	0	1	0	0	...	0	0	0	1	0
0	0	0	1	0	...	0	0	0	1	1
0	0	0	0	1	...	0	0	1	0	0
0	0	0	0	0	...	1	1	1	1	1

- Umwandlung eines einzelnen Inputs in eine Binärzahl
- Vorteil weniger Datentransfer
- Gleichzeitige Tastendrücke werden hier nicht kodiert

1-zu-k Demultiplexer

- **1-zu-k Demultiplexer:**
  - n Steuerleitungen $s_{n-1}, \ldots, s_1, s_0$
  - ein Eingang x
  - $k = 2^n$ Ausgänge $y_0, y_1, \ldots, y_{k-1}$
  - $y_i = x$ für $(s_{n-1}, \ldots, s_1, s_0)_2 = i$



Die Belegung des Eingangs wird zu **genau einem** Ausgang geleitet.
Der Ausgang wird über die Steuerleitungen ausgewählt.

- Zeitdemultiplexer - zum erhöhen der Bandbreite von Übertragungen (*TDM = Time Devision Multiplexing*)
- digitaler Demultiplexer - parallele Aufteilung von Datenströmen auf mehrere digitale Ausgänge, bzw. zum Lenken der Datenströme in Mikroprozessorsystemen

Bsp. zu Demultiplexer

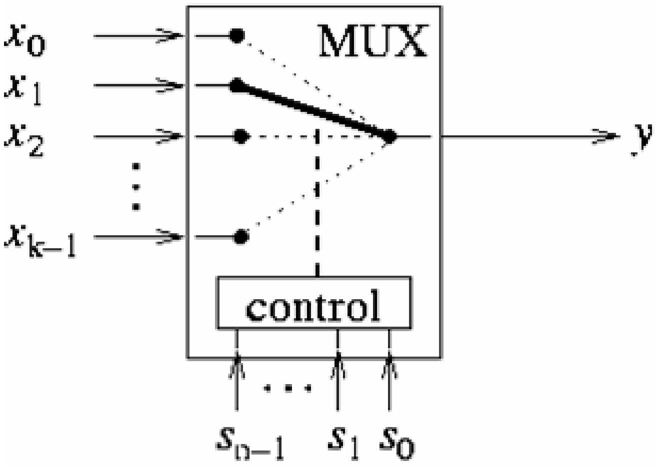
1-zu-4 Demultiplexer:

s_0	s_1	x	y_0	y_1	y_2	y3	
0	0	0	0	0	0	0	y_0
0	0	1	1	0	0	0	
0	1	0	0	0	0	0	y_1
0	1	1	0	1	0	0	
1	0	0	0	0	0	0	y_2
1	0	1	0	0	1	0	
1	1	0	0	0	0	0	y_3
1	1	1	0	0	0	1	

- Gegenstück zum Multiplexer
- Umschaltung kann statisch erfolgen oder periodisch/zyklisch (dann aber zeitliche Abstimmung erforderlich)

1-aus-k Multiplexer

- **1-aus-k Multiplexer:**
  - n Steuerleitungen: $s_{n-1}, \ldots , s_1, s_0$
  - $k = 2^n$ Eingänge: $x_0, x_1, \ldots , x_{k-1}$
  - ein Ausgang: y
  - $y = x_i$ für $(s_{n-1}, \ldots , s_1, s_0)_2 = i$

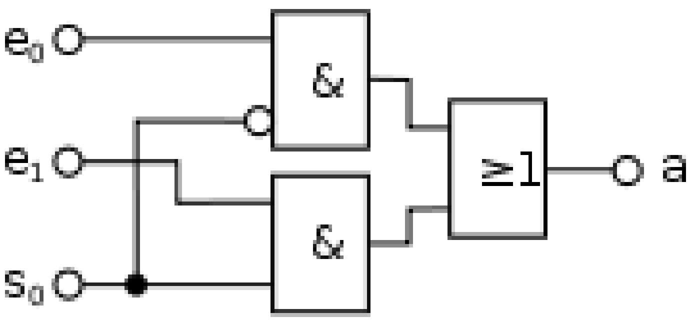


Genau **ein** durch die Belegung der Selektionseingänge gewählter Eingang wird zum Ausgang propagiert.

Bsp. Zu Multiplexern

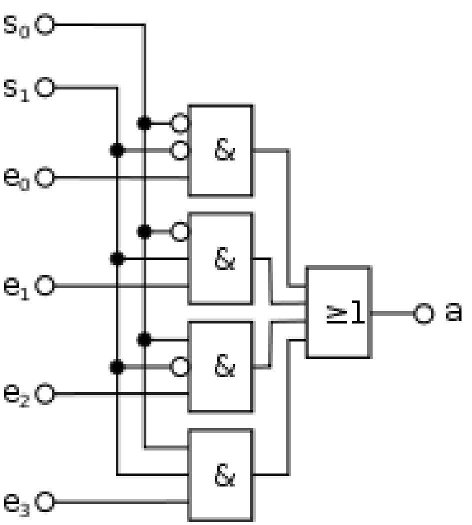
Einfach-Multiplexer: (1-aus-2)

s_0	e_0	e_1	a	a
0	0	-	0	e_0
0	1	-	1	
1	-	0	0	e_1
1	-	1	1	



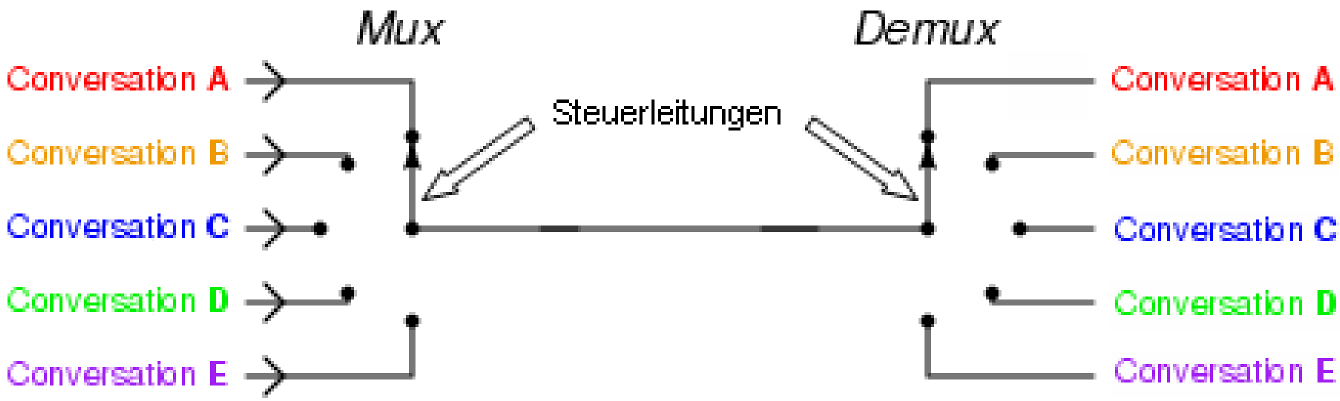
Zweifach-Multiplexer: (1-aus-4)

s_0	s_1	e_0	e_1	e_2	e_3	a	a
0	0	0	-	-	-	0	e_0
0	0	1	-	-	-	1	
0	1	-	0	-	-	0	e_1
0	1	-	1	-	-	1	
1	0	-	-	0	-	0	e_2
1	0	-	-	1	-	1	
1	1	-	-	-	0	0	e_3
1	1	-	-	-	1	1	



Mux-Demux

Zusammenhang zwischen Multi- und Demultiplexer:

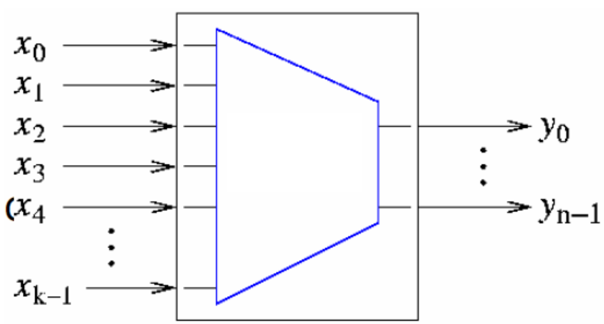


Beim Multiplexverfahren werden mehrere verschiedene Signale gebündelt oder zeitlich ineinander verschachtelt, um sie ohne gegenseitige Beeinflussung simultan und gemeinsam übertragen zu können.

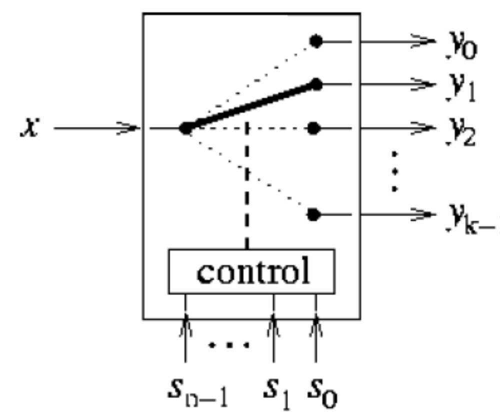
Übungsaufgaben

1. Welche Schaltnetze sind hier abgebildet? Benennen Sie 1-4 und begründen Sie ihre Wahl.

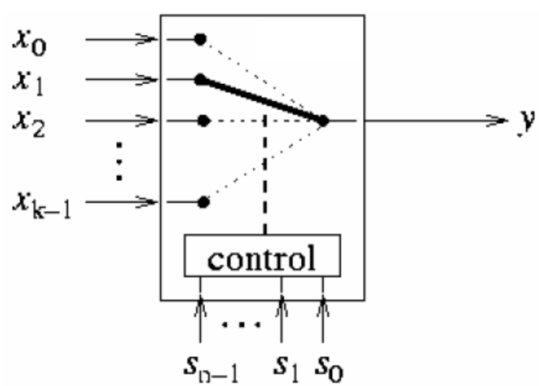
1.1.



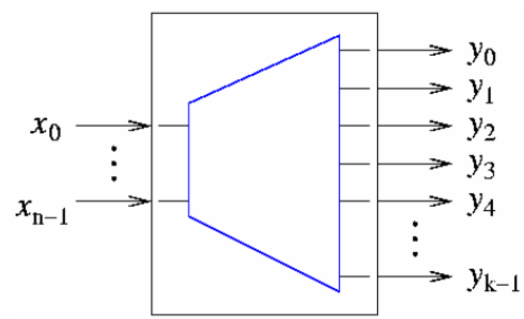
1.2.



1.3.



1.4.



Übungsaufgaben

2. Vereinfachen sie mithilfe des Quine-McCluskey-Verfahrens.

$$
\begin{aligned}
a = {} & \overline{X}_1 \overline{X}_2 \overline{X}_3 \overline{X}_4 + \overline{X}_1 \overline{X}_2 \overline{X}_3 X_4 + \\
& \overline{X}_1 X_2 \overline{X}_3 X_4 + \overline{X}_1 X_2 X_3 \overline{X}_4 + \\
& \overline{X}_1 X_2 X_3 X_4 + X_1 \overline{X}_2 \overline{X}_3 \overline{X}_4 + \\
& X_1 \overline{X}_2 \overline{X}_3 X_4 + X_1 \overline{X}_2 X_3 \overline{X}_4 + \\
& X_1 X_2 X_3 \overline{X}_4 + X_1 X_2 X_3 X_4
\end{aligned}
$$