

Rechnerarchitektur

Kombinatorische Logik II

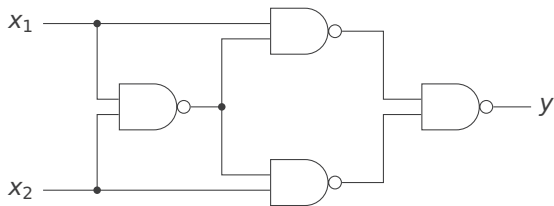
Univ.-Prof. Dr.-Ing. Rainer Böhme

Wintersemester 2021/22 · 20. Oktober 2021

Was tut diese Schaltung ?



24 82 94 16



x_1	x_2	y
0	0	
0	1	
1	0	
1	1	

Bitte wählen Sie die passende Spalte für y in ARSnova.

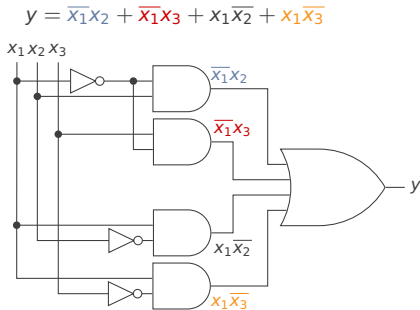
Zugang: <https://arsnova.uibk.ac.at> mit Zugangsschlüssel **24 82 94 16**. Oder scannen Sie den QR-Kode.

Beispiel einer logischen Schaltung (W)

Gesucht Schaltung, die 1 ausgibt, wenn einer oder zwei von drei Eingängen x_1, x_2, x_3 den Wert 1 annehmen.

Wahrheitstabelle, Boolesche Funktion, Realisierung:

x_1	x_2	x_3	y
0	0	0	0
1	0	0	1 ✓
0	1	0	1 ✓
1	1	0	1 ✓
0	0	1	1 ✓
1	0	1	1 ✓
0	1	1	1 ✓
1	1	1	0



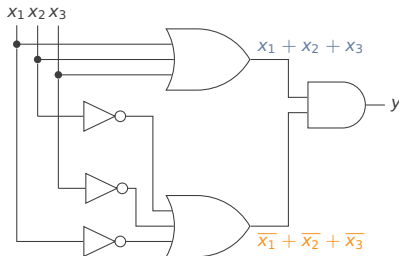
Beispiel einer logischen Schaltung (Forts.)

Gesucht Schaltung, die 1 ausgibt, wenn einer oder zwei von drei Eingängen x_1, x_2, x_3 den Wert 1 annehmen.

Alternative:

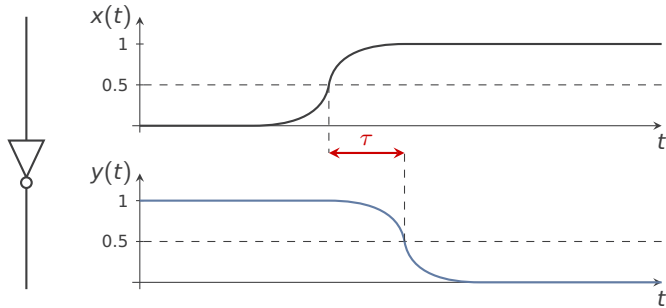
x_1	x_2	x_3	y
0	0	0	0
1	0	0	1
0	1	0	1
1	1	0	1
0	0	1	1
1	0	1	1
0	1	1	1
1	1	1	0

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



Zeitverhalten eines Gatters

Gatter sind physische Bausteine. Sie verhalten sich nicht ideal.



Das Ausgangssignal reagiert verzögert

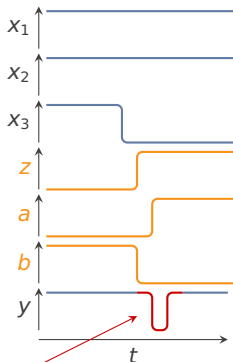
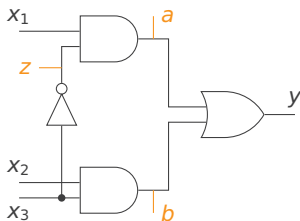
Die Verzögerung τ ist definiert als Dauer zwischen den Zeitpunkten der Überschreitung des 50 %-Pegels an Ein- bzw. Ausgang.

Störimpulse durch Laufzeiteffekte

Beispiel: Eingang x_3 steuert, ob x_1 oder x_2 am Ausgang y anliegt.

x_1	x_2	x_3	y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

$$y = \bar{x}_3 x_1 + x_3 x_2$$

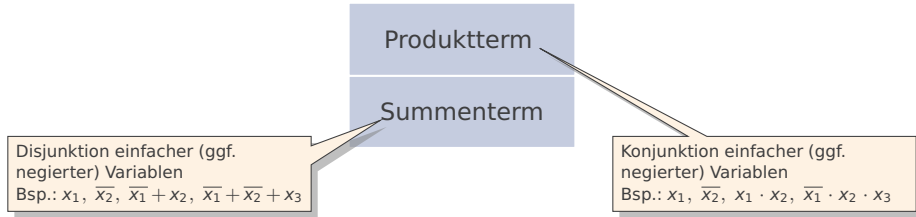


„statischer Eins-Hazard“

Gliederung heute

- 0. Konfrontation mit der Realität
- 1. Kanonische Darstellungen**
- 2. Minimierung
- 3. Typische Schaltnetze

Systematik der Darstellung Boolescher Funktionen



Systematik der Darstellung Boolescher Funktionen

Minterm

Produktterm, in dem jede Variable einer Booleschen Funktion genau einmal (ggf. negiert) vorkommt

Bsp.: $\overline{x_1} \cdot x_2 \cdot x_3$ ist ein Minterm von $f(x_1, x_2, x_3)$

Maxterm

Summenterm, in dem jede Variable einer Booleschen Funktion genau einmal (ggf. negiert) vorkommt

Bsp.: $\overline{x_1} + x_2 + x_3$ ist ein Maxterm von $f(x_1, x_2, x_3)$

Systematik der Darstellung Boolescher Funktionen

Disjunktion von Produkttermen
(Summe von Produkten, DNF)
Bsp.: $(x_1 \cdot x_2) + (\bar{x}_1 \cdot x_2 \cdot x_3)$

Disjunktive
Normalform

Konjunktion von Summentermen
(Produkt von Summen, KNF)
Bsp.: $(x_1 + x_2) \cdot (\bar{x}_1 + x_2 + x_3)$

Konjunktive
Normalform

Systematik der Darstellung Boolescher Funktionen

Eindeutige Darstellung einer Booleschen Funktion f als Disjunktion von Mintermen

Bsp.:

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

Eindeutige Darstellung einer Booleschen Funktion f als Konjunktion von Maxtermen

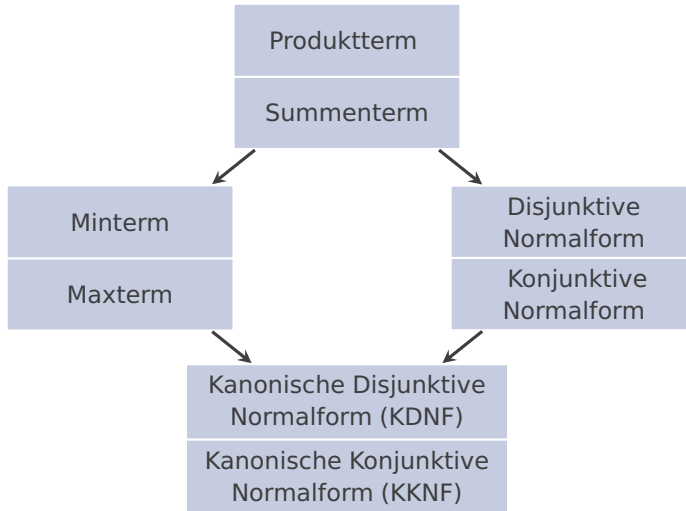
Bsp.:

$$(\overline{x_1} + \overline{x_2}) \cdot (\overline{x_1} + x_2) \cdot (x_1 + \overline{x_2})$$

Kanonische Disjunktive
Normalform (KDNF)

Kanonische Konjunktive
Normalform (KKNF)

Systematik der Darstellung Boolescher Funktionen



Sätze zur Darstellung Boolescher Funktionen

1. **Jede** Boolesche Funktion lässt sich als **genau eine KDNF** (Disjunktion von Mintermen) darstellen.
2. **Jede** Boolesche Funktion lässt sich als **genau eine KKNF** (Konjunktion von Maxtermen) darstellen.
3. **Jede KDNF** kann in eine KKNF umgewandelt werden.
4. **Jede KKNF** kann in eine KDNF umgewandelt werden.
5. Aufgrund der **Dualität** gilt:

$$\text{KKNF}(f(x_1, x_2, \dots, x_n)) = \overline{\text{KDNF}(\overline{f(x_1, x_2, \dots, x_n)})}$$

und

$$\text{KDNF}(f(x_1, x_2, \dots, x_n)) = \overline{\text{KKNF}(\overline{f(x_1, x_2, \dots, x_n)})}$$

Bildung der KDNF (Disjunktion von Mintermen)

aus der Wahrheitstabelle einer n -stelligen Booleschen Funktion

- **Idee:** Summe nimmt den Wert **1** an, wenn mindestens ein Summand **1** ist.
- Für jede Zeile der Wahrheitstabelle mit $f(x_1, \dots, x_n) = \mathbf{1}$ wird einer der Minterme ermittelt.
- Variable x_i wird negiert, wenn in der entsprechenden Zelle der Wert der Variable 0 ist.

Beispiel

x_1	x_2	x_3	$f(\mathbf{x})$	
\vdots	\vdots	\vdots	\vdots	
1	0	1	1	$\longrightarrow x_1 \cdot \overline{x_2} \cdot x_3$
\vdots	\vdots	\vdots	\vdots	

Bildung der KKNF (Konjunktion von Maxtermen)

aus der Wahrheitstabelle einer n -stelligen Booleschen Funktion

- **Idee:** Produkt nimmt den Wert **0** an, wenn mindestens ein Faktor **0** ist.
- Für jede Zeile der Wahrheitstabelle mit $f(x_1, \dots, x_n) = \mathbf{0}$ wird einer der Maxterme ermittelt.
- Variable x_i wird negiert, wenn in der entsprechenden Zelle der Wert der Variable 1 ist.

Beispiel

x_1	x_2	x_3	$f(\mathbf{x})$
\vdots	\vdots	\vdots	\vdots
0	0	1	0 $\rightarrow (x_1 + x_2 + \overline{x_3})$
\vdots	\vdots	\vdots	\vdots

Äquivalenz von und über Normalformen

Eindeutigkeit

(Folgesätze)

- Die Darstellung einer Booleschen Funktion durch KDNF bzw. KKNF ist (abgesehen von der Reihenfolge) **eindeutig**.
- Zwei allgemeine Darstellungen Boolescher Funktionen sind **äquivalent**, wenn sie (durch Umformungen nach den Regeln der Booleschen Algebra) auf die gleiche KDNF bzw. KKNF zurückgeführt werden können.



MY HOBBY: POINTING THIS OUT EVERY DAY.

Illustration: xkcd.com

Realisierung günstiger Schaltungen

Systematische Realisierung einer Booleschen Funktion f in drei Schritten:

1. Aufstellen der Wahrheitstabelle von f
2. Bilden der KDNF (oder KKNF) von f

$$\text{KDNF: } f(x_1, x_2) = \overline{x_1} \cdot \overline{x_2} + \overline{x_1} \cdot x_2 + x_1 \cdot x_2$$

$$\text{KKNF: } f(x_1, x_2) = \overline{x_1} + x_2$$

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

3. Schaltungstechnische Realisierung mit Gattern
(hier: KKNF)



Einfache Optimierungsregel

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

Bemerkungen zur technischen Realisierung

Alle Booleschen Funktionen lassen sich mit ...

- maximal **zwei Gatterebenen** realisieren, wenn alle Eingangssignale x_i sowohl einfach als auch **negiert** vorliegen,
- sonst mit maximal **drei Gatterebenen**.

Realisierung einer KDNF

- Max. 2^n AND-Gatter mit je n Eingängen (eines pro Minterm)
- Ein OR-Gatter zur Disjunktion aller Minterme (mit max. 2^n Eingängen)

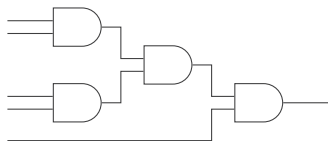
Realisierung einer KKNF

- Max. 2^n OR-Gatter mit je n Eingängen (eines pro Maxterm)
- Ein AND-Gatter zur Konjunktion aller Maxterme (mit max. 2^n Eingängen)

Bemerkungen zur technischen Realisierung (Forts.)

Viele Standardbauteile realisieren Gatter mit **zwei Eingängen**.

Beispiel für ein Gatter mit $k = 5$ Eingängen aus Standardbauteilen:



Anzahl Gatter = $k - 1$

Anzahl Ebenen = $\lceil \log_2 k \rceil$

Realisierung einer **kanonischen Normalform**:

- Max. $2^n(n - 1) + (2^n - 1) = n \cdot 2^n - 1$ Gatter (mit 2 Eingängen)
- Max. $\log_2 n + \log_2 (2^n) = \log_2 n + n$ Ebenen (aus Gattern mit 2 Eingängen)

„Universelle“ Gatter

NAND-Gatter zur Realisierung von [K]DNF



NOR-Gatter zur Realisierung von [K]KNF



Gliederung heute

- 0. Konfrontation mit der Realität
- 1. Kanonische Darstellungen
- 2. **Minimierung**
- 3. Typische Schaltnetze

Minimierung

„Einfache“ Optimierungskriterien

- Anzahl Gatter → Anzahl **Boolescher Operationen**
- Anzahl Verbindungen
- Anzahl Produkt- bzw. Summenterme

Ansätze

- Händisches Umformen nach Regeln der Booleschen Algebra
- Graphische Verfahren (z. B. **Karnaugh-Veitch-Diagramme**)
- Algorithmen (z. B. **Quine & McCluskey, auch bei vielen Variablen**)

Resolutionsregeln

Für [K]DNF

Wenn sich zwei Summanden nur in **genau einer** komplementären Variable unterscheiden, dann können beide Terme durch ihren gemeinsamen Teil ersetzt werden.

Beispiel

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

Beweis über

- Distributivität $x_1 \cdot \overline{x_2} \cdot x_3 \cdot (x_4 + \overline{x_4})$ sowie (3)
- komplementäre und neutrale Elemente. (7) (6)

Siehe Folie **Axiome** aus der vergangenen Woche.

Resolutionsregeln (Forts.)

Für [K]KNF

Wenn sich zwei Faktoren nur in **genau einer** komplementären Variable unterscheiden, dann können beide Terme durch ihren gemeinsamen Teil ersetzt werden.

Beispiel

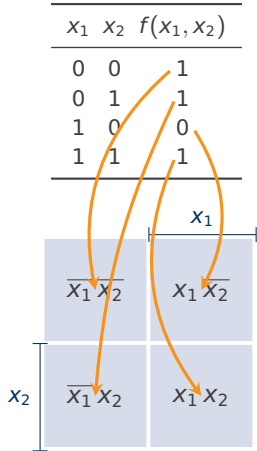
$$(x_1 + x_2 + \overline{x_3} + \overline{x_4}) \cdot (x_1 + x_2 + x_3 + \overline{x_4}) \Leftrightarrow (x_1 + x_2 + \overline{x_4})$$

Beweis über

- Kommutativität $(x_1 + x_2 + \overline{x_4} + \overline{x_3}) \cdot (x_1 + x_2 + \overline{x_4} + x_3)$ (1)
- Assoziativität $((x_1 + x_2 + \overline{x_4}) + \overline{x_3}) \cdot ((x_1 + x_2 + \overline{x_4}) + x_3)$ (11)
- Distributivität $(x_1 + x_2 + \overline{x_4}) + (x_3 \cdot \overline{x_3})$ sowie (4)
- komplementäre und neutrale Elemente. (8) (5)

Siehe Folie **Axiome** aus der vergangenen Woche.

Karnaugh–Veitch-Diagramme (KV)



- 2-dimensionale Darstellung der Funktionswerte aus der Wahrheitstabelle
- Jedes Element der Matrix repräsentiert einen Minterm.
- Anordnung der Elemente, sodass sich zwei (zyklisch) **benachbarte Elemente im Vorzeichen genau einer Variable unterscheiden**
- Ermöglicht Zusammenfassung benachbarter Minterme

Herkunft: Maurice Karnaugh's Weiterentwicklung (1953) der Diagramme von Edward Veitch ('52)

Minimierung einer KDNF mit KV-Diagrammen

1. Gegebene KDNF: z. B. $f(x_1, x_2) = \overline{x_1} \cdot \overline{x_2} + \overline{x_1} \cdot x_2 + x_1 \cdot x_2$

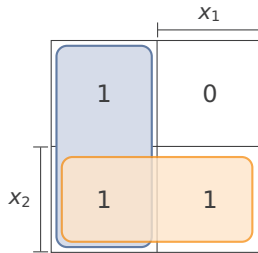
2. Erstellen des KV-Diagramms:

1 für jeden Minterm mit $f(\mathbf{x}) = 1$, sonst 0

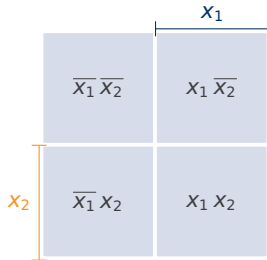
3. Markierung möglichst weniger, großer, rechteckiger und ggf. überlappender Bereiche aus 2^k Einsen, sodass alle Einsen überdeckt sind

4. Bildung einer minimalen DNF durch Summierung von genau einem

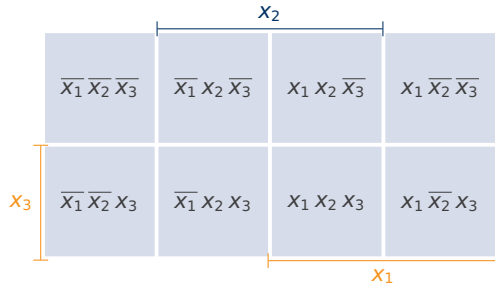
Produktterm pro markiertem Bereich: $f(x_1, x_2) = \overline{x_1} + x_2$



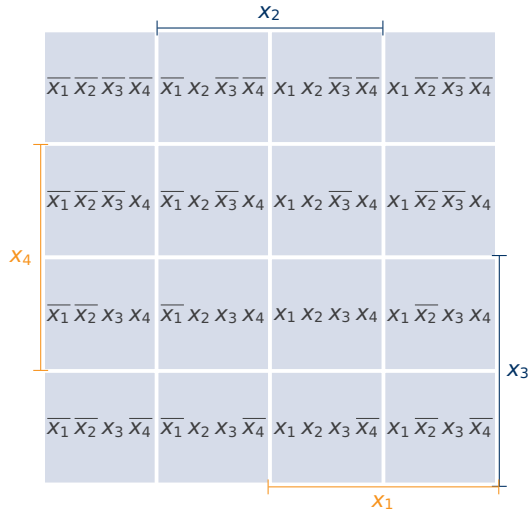
KV-Diagramme für mehrstellige Funktionen



KV-Diagramme für mehrstellige Funktionen

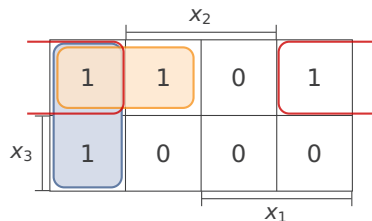
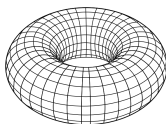


KV-Diagramme für mehrstellige Funktionen



Beispiel mit zyklischer Markierung

Minimiere $y = f(x_1, x_2, x_3)$



x_1	x_2	x_3	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

Minimale DNF: $y = \overline{x_1} \cdot \overline{x_2} + \overline{x_1} \cdot \overline{x_3} + \overline{x_2} \cdot \overline{x_3}$

Minimierung einer KKNF mit KV-Diagrammen

1. Markierung möglichst weniger, großer, rechteckiger und ggf. überlappender Bereiche aus 2^k **Nullen**, sodass alle Nullen überdeckt sind
2. Bildung einer **DNF** durch Summierung von genau einem Produktterm pro markiertem Bereich
3. Umwandlung in eine **minimale KNF** durch abschließende Negation

Grenzen der Karnaugh-Veitch-Diagramme

- Zyklische Markierungen können leicht übersehen werden.
- KV-Diagramme für Boolesche Funktionen mit fünf oder mehr Stellen sind ungebräuchlich.

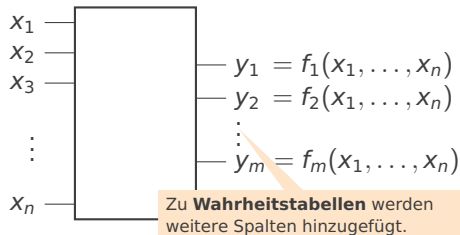
Gliederung heute

- 0. Konfrontation mit der Realität
- 1. Kanonische Darstellungen
- 2. Minimierung
- 3. **Typische Schaltnetze**

Synthese von Schaltnetzen

(Wiederholung)

Jede Schaltfunktion $f : \{0, 1\}^n \rightarrow \{0, 1\}^m$ mit $m, n \geq 1$ ist
in m Boolesche Funktionen mit den gleichen n Eingangsvariablen zerlegbar:

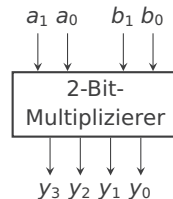


Definition

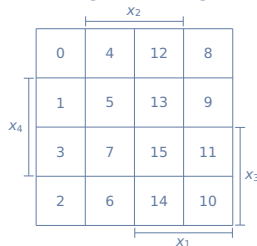
Ein **Schaltnetz** (auch synonym: *kombinatorische Logik*) ist eine schaltungstechnische Realisierung einer Schaltfunktion.

Beispiel mit Minimierung: 2-Bit-Multiplizierer

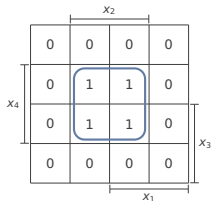
	1. Faktor		2. Faktor		Ergebnis			
$a \times b = y$	$a_1 = x_1$	$a_0 = x_2$	$b_1 = x_3$	$b_0 = x_4$	y_3	y_2	y_1	y_0
$0 \times 0 = 0$	0	0	0	0	0	0	0	0
$0 \times 1 = 0$	0	0	0	1	0	0	0	0
$0 \times 2 = 0$	0	0	1	0	0	0	0	0
$0 \times 3 = 0$	0	0	1	1	0	0	0	0
$1 \times 0 = 0$	0	1	0	0	0	0	0	0
$1 \times 1 = 1$	0	1	0	1	0	0	0	1
$1 \times 2 = 2$	0	1	1	0	0	0	1	0
$1 \times 3 = 3$	0	1	1	1	0	0	1	1
$2 \times 0 = 0$	1	0	0	0	0	0	0	0
$2 \times 1 = 2$	1	0	0	1	0	0	1	0
$2 \times 2 = 4$	1	0	1	0	0	1	0	0
$2 \times 3 = 6$	1	0	1	1	0	1	1	0
$3 \times 0 = 0$	1	1	0	0	0	0	0	0
$3 \times 1 = 3$	1	1	0	1	0	0	1	1
$3 \times 2 = 6$	1	1	1	0	0	1	1	0
$3 \times 3 = 9$	1	1	1	1	1	0	0	1



Reihenfolge für KV-Diagramm

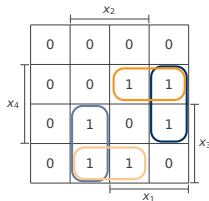


KV-Diagramme für den 2-Bit-Multiplizierer



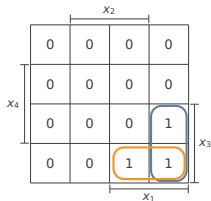
$$y_0 = x_2 \cdot x_4$$

$$= a_0 \cdot b_0$$



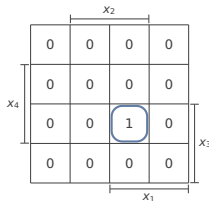
$$y_1 = \overline{x_1}x_2x_3 + x_2x_3\overline{x_4} + x_1\overline{x_3}x_4 + x_1\overline{x_2}x_4$$

$$= \overline{a_1}a_0b_1 + a_0b_1\overline{b_0} + a_1\overline{b_1}b_0 + a_1\overline{a_0}b_0$$



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

$$= a_1 \cdot \overline{a_0} \cdot b_1 + a_1 \cdot b_1 \cdot \overline{b_0}$$



$$y_3 = x_1 \cdot x_2 \cdot x_3 \cdot x_4$$

$$= a_1 \cdot a_0 \cdot b_1 \cdot b_0$$

Realisierung als minimiertes Schaltnetz

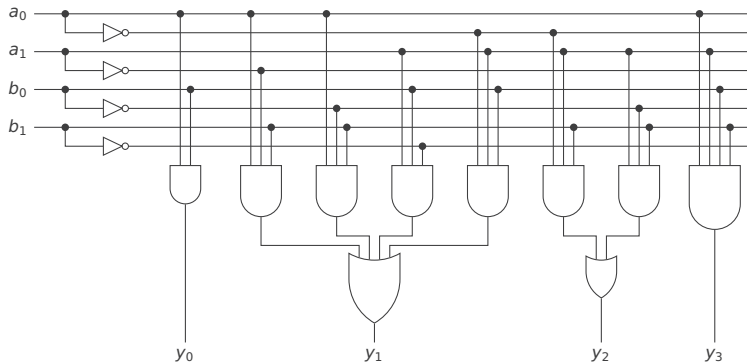
2-Bit-Multiplikierer

$$y_0 = a_0 \cdot b_0$$

$$y_1 = a_0 \cdot \overline{a_1} \cdot b_1 + a_0 \cdot \overline{b_0} \cdot b_1 + a_1 \cdot b_0 \cdot \overline{b_1} + \overline{a_0} \cdot a_1 \cdot b_0$$

$$y_2 = \overline{a_0} \cdot a_1 \cdot b_1 + a_1 \cdot \overline{b_0} \cdot b_1$$

$$y_3 = a_0 \cdot a_1 \cdot b_0 \cdot b_1$$

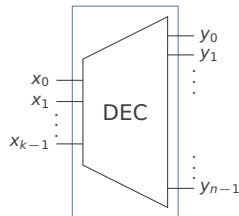


Dekodierer

k -zu- n -Dekodierer: **Auswahl eines** von n Ausgängen $y_i = 1$ durch Binärdarstellung an den Eingängen (x_0, \dots, x_{k-1}) .

Es gilt $0 \leq i < n$ und $1 \leq n \leq 2^k$.

Anwendung: Adressen oder Instruktionen



Beispiel: 2-zu-4-Dekodierer

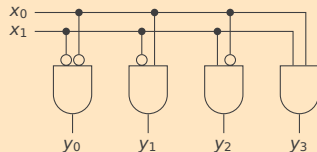
x_1	x_0	y_3	y_2	y_1	y_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

$$y_0 = \overline{x_0} \cdot \overline{x_1}$$

$$y_1 = x_0 \cdot \overline{x_1}$$

$$y_2 = \overline{x_0} \cdot x_1$$

$$y_3 = x_0 \cdot x_1$$

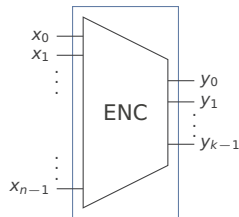


Kodierer

n -zu- k -Kodierer: Ausgabe (y_0, \dots, y_{k-1}) ist **Binärdarstellung** für den Index **eines** aktiven Eingangs $x_i = 1$.

Es gilt $0 \leq i < n$ und $k \geq \lceil \log_2 n \rceil$.

Anwendung: Kodierung gedrückter Taste

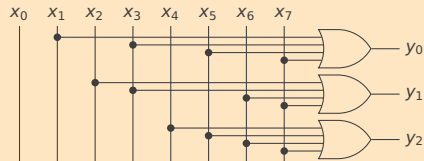


Beispiel: Naiver 8-zu-3-Kodierer

$$y_0 = x_1 + x_3 + x_5 + x_7$$

$$y_1 = x_2 + x_3 + x_6 + x_7$$

$$y_2 = x_4 + x_5 + x_6 + x_7$$



Problem: undefinierte Ausgabe, falls mehrere Eingänge aktiv sind.

Syllabus – Wintersemester 2021/22

06.10.21	1. Einführung
13.10.21	2. Kombinatorische Logik I
20.10.21	3. Kombinatorische Logik II
27.10.21	4. Sequenzielle Logik I
03.11.21	5. Sequenzielle Logik II
10.11.21	6. Arithmetik I
17.11.21	7. Arithmetik II
24.11.21	8. Befehlssatzarchitektur (ARM) I
01.12.21	9. Befehlssatzarchitektur (ARM) II
15.12.21	10. Ein-/Ausgabe
12.01.22	11. Prozessorarchitekturen
19.01.22	12. Speicher
26.01.22	13. Leistung
02.02.22	Klausur (1. Termin)