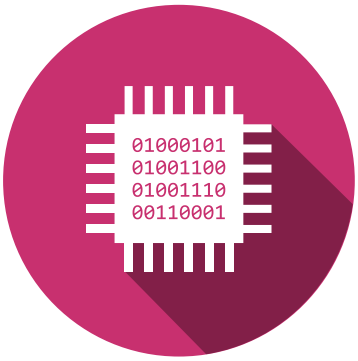
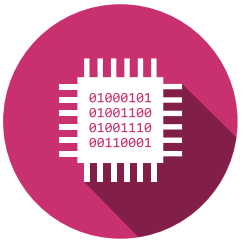


Kombinatorische Logikfunktionen (COM)

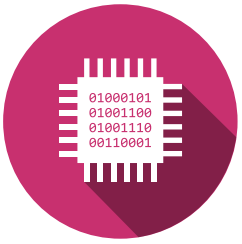
Vorlesung Digitales Design

Hes·so VALAIS WALLIS
Haute Ecole d'Ingénierie
Hochschule für Ingenieurwissenschaften

Orientierung: Energie und Umwelttechnik (ETE)
Kurs: Digitales Design (DiD)
Verfasser: Christophe Bianchi, François Corthay, Pierre Pompili, Silvan Zahno
Datum: 25. August 2022
Version: v2.1

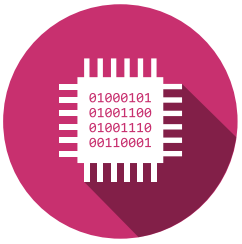


Inhaltsverzeichnis



1 Einführung

Die kombinatorischen Logiksysteme befassen sich mit Fällen, wo die Ausgänge nur von den Eingängen abhängen, im Gegensatz zu den sequentiellen Logiksystemen, wo die Ausgänge auch von der Speicherung gewisser Ereignisse abhän- gen. In diesem Kapitel werden die grundlegenden Funktionen der kombinatorischen Systeme beschrieben: **Inverter**, **logisches UND**, **logisches ODER**. Zudem werden die Techniken beschrieben, die für die Darstellung und Analyse der kombinatorischen Logikfunktionen benutzt werden.



2 Darstellung von kombinatorischen Funktionen

2.1 Darstellung mit einer Wahrheitstabelle

Für das Aufstellen einer Wahrheitstabelle benutzt man die Liste aller möglichen Kombinationen der Eingänge des Systems; für jede Kombination wird zudem der Wert des Ausgangs angegeben.

Ein System mit n unabhängigen Eingängen besitzt 2^n mögliche Kombinationen. Die Liste dieser Kombinationen entspricht den Binärzahlen zwischen 0 und $2^n - 1$.

Beispiel: Steuerung einer Ventilation

Für die Temperaturregelung eines Raums benützt man ein Ventilationssystem. Wenn die Aussentemperatur höher ist als die Temperatur im Raum, wird die Ventilation in der Regel eingeschaltet, um den Raum zu heizen. Im Sommer und während des Tages wird die Ventilation auch zur Kühlung des Raums benutzt: wenn die Aussentemperatur tiefer als die Temperatur im Raum ist, wird die Ventilation in Betrieb gesetzt.

Die Steuerung der Ventilation hängt hier von 3 logischen Variablen ab:

- *winter*, für die im Winter '1' und im Sommer '0' gilt,
- *night*, für die in der Nacht '1' und am Tag '0' gilt,
- *ext_temp*, gleich '1', wenn die Aussentemperatur höher als die Temperatur im Raum ist und gleich '0' in allen anderen Fällen.

Mit diesen 3 Eingangsvariablen kann der Wert des Signals Ventilation bestimmt werden, welches die Inbetriebnahme des Ventilators steuert. Die Tabelle 1 zeigt den Wert dieses Ausgangssignals in Abhängigkeit von den 2^3 möglichen Kombinationen der Eingänge.

winter	*night*	*ext_hot*	*ventilation*
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Tabelle 1: Wahrheitstabelle der Steuerung einer Ventilation

2.2 Darstellung mit einem Ablaufdiagramm

Ein Ablaufdiagramm stellt den zeitlichen Ablauf der Signale des Systems dar.

Damit die Funktionsweise des Systems vollständig spezifiziert werden kann, müssen alle Möglichkeiten aufgezeigt werden.

Beispiel: Steuerung einer Ventilation

Das Ablaufdiagramm der Abbildung 1 zeigt eine mögliche Funktionsweise der Steuerung für die Ventilation im vorangehenden Beispiel.

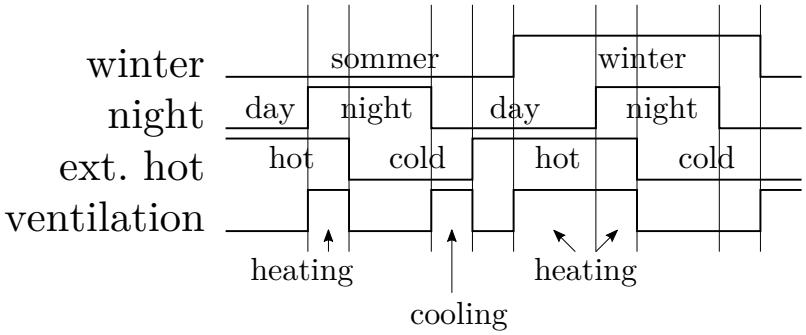


Abbildung 1: Ablaufdiagramm der Steuerung einer Ventilation

2.3 Darstellung mit dem Venn-Diagramm

Im Venn-Diagramm werden die Eingänge in Form von Mengen dargestellt. Jedem Bereich des Diagramms wird eine Farbe oder eine Schattierung zugeteilt, womit der Wert des Ausgangs spezifiziert wird.

In einem solchen Diagramm müssen die Mengen so angeordnet werden, dass alle möglichen Schnittpunkte aufgezeigt werden.

Beispiel: Steuerung einer Ventilation

Das Venn-Diagramm in der Abbildung 2 gibt, in Abhängigkeit von allen möglichen Eingängen, den Wert der Steuerung für die Ventilation im obengenannten Beispiel an.

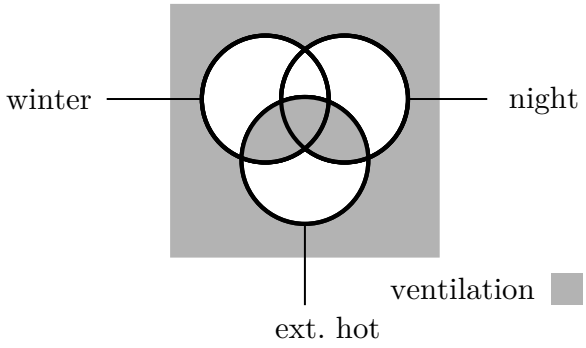
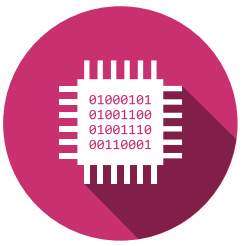


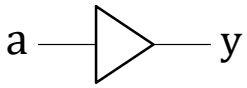
Abbildung 2: Venn-Diagramm der Steuerung einer Ventilation



3 Elementare Logikfunktionen

3.1 Puffer

Die Abbildung 3 zeigt das Symbol der Pufferfunktion, ihre Wahrheitstafel sowie ihre Darstellung in einem Venn-Diagramm.



a	y
0	0
1	1

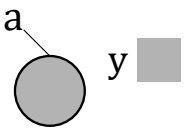
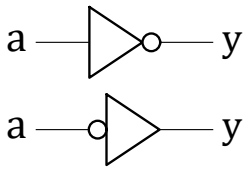


Abbildung 3: Puffer

3.2 Inverter

Die Abbildung 4 zeigt zwei Symbole des Inverters, seine Wahrheitstabelle sowie die Darstellung seiner Funktion in einem Venn-Diagramm.



a	y
0	1
1	0

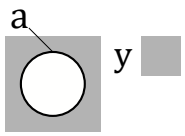
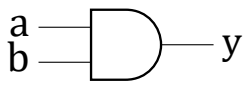


Abbildung 4: Inverter

3.3 UND-Funktion

Die Abbildung 5 zeigt das Symbol der UND-Funktion, ihre Wahrheitstabelle sowie ihre Darstellung in einem Venn-Diagramm.



a	b	y
0	0	0
0	1	0
1	0	0
1	1	1

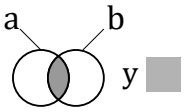
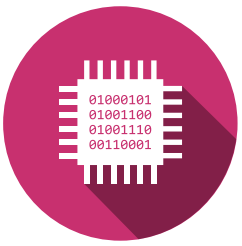
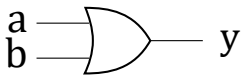


Abbildung 5: UND-Funktion

3.4 ODER-Funktion

Die Abbildung 6 zeigt das Symbol der ODER-Funktion, ihre Wahrheitstabelle sowie ihre Darstellung in einem Venn-Diagramm.





a	b	y
0	0	0
0	1	1
1	0	1
1	1	1

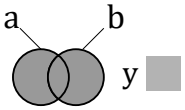
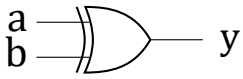


Abbildung 6: ODER-Funktion

3.5 Exklusiv-ODER-Funktion

Die Abbildung 7 zeigt das Symbol der Exklusiv-ODER-Funktion, ihre Wahrheits- tabelle sowie ihre Darstellung in einem Venn-Diagramm.



a	b	y
0	0	0
0	1	1
1	0	1
1	1	0

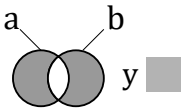


Abbildung 7: Exklusiv-ODER-Funktion

3.6 Nachschlagetabelle LUT

Look-**U**p **T**able (LUT) bestehen aus einer bestimmten Anzahl von ODER-Gattern, denen UND-Gatter vorangehen, deren Eingänge programmierbar sind. Dadurch können Funktionen aus Wahrheitstabellen realisiert werden. In der Abbildung 10 ist eine LUT mit 3 Eingängen a, b und c sowie einem Ausgang y dargestellt.

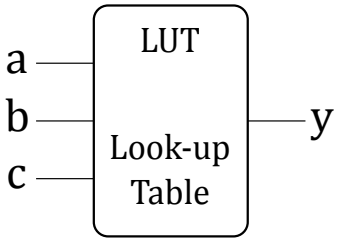
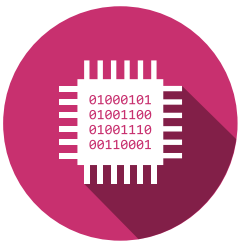
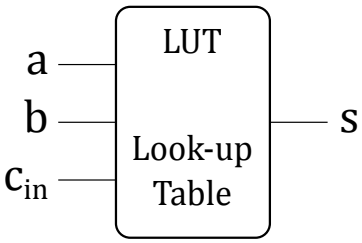


Abbildung 8: LUT mit 3 Eingängen

3.6.1 Beispiel

In der Abbildung 9 ist die Funktion eines Addierers dargestellt. Die Eingänge a, b und c_{in} werden addiert und ergeben das Resultat s.





a	b	c_{in}	s
0	0	0	0
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	1

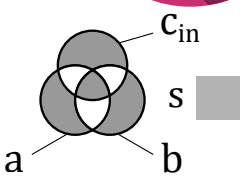


Abbildung 9: Addierer Funktion implementiert in einer LUT

In der Abbildung 10 ist die interne Struktur der LUT zu sehen, welche die Addierer Funktion implementiert.

In einem **P**rogrammable **A**rray **L**ogic (PAL) Chip sind LUT mit verschiedenen Eingängen vorhanden wie in der Abbildung 11 dargestellt. Dadurch können Funktionen aus der Polynomform ihrer Gleichungen realisiert werden, sofern die Anzahl der Terme des Polynoms kleiner oder gleich der Anzahl der Eingänge der zur Verfügung stehenden ODER-Gatter ist.

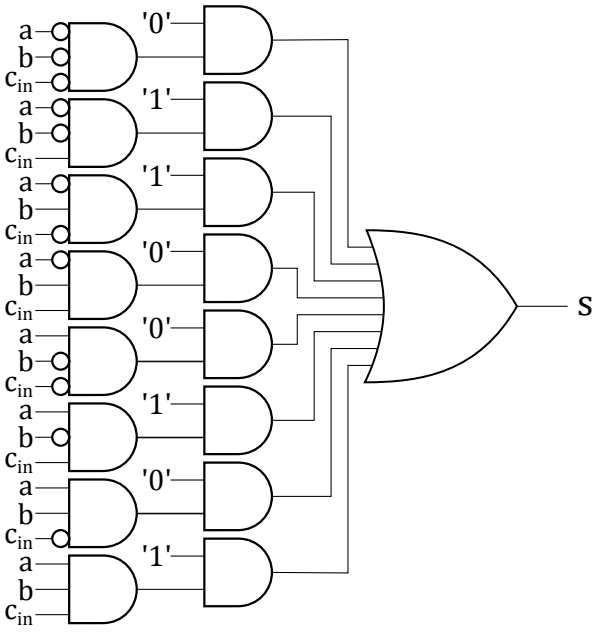


Abbildung 10: Interne Struktur der LUT

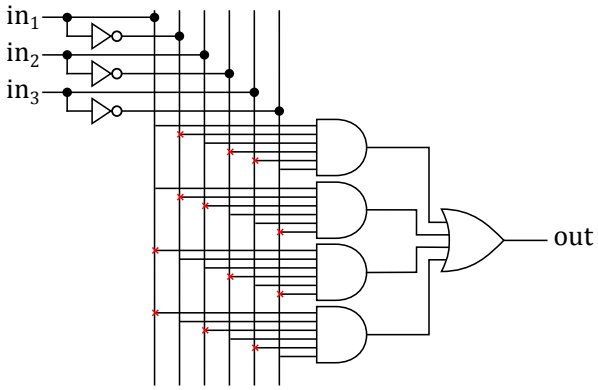
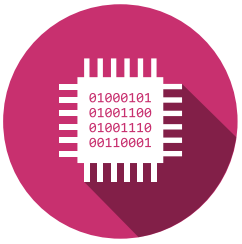


Abbildung 11: Interne Struktur einer PAL LUT



4 Boolesche Algebra

Die Boolesche Algebra, nach dem Mathematiker Boole benannt, befasst sich mit Operationen der logischen Variablen mit den Werten '0' oder '1'.

4.1 Bezeichnung: Boolescher Operator

In der Tabelle 2 ist der algebraische Ausdruck der kombinatorischen Logik-Grundfunktionen angegeben.

Operator	Ausdruck
Inverter	$y = \bar{a}$
UND	$y = a * b$
ODER	$y = a + b$
exklusiv-ODER	$y = a \oplus b$

Tabelle 2: Boolesche Operatoren

Die Exklusiv-ODER-Funktion kann mit Hilfe der Inverter-, der UND- und der ODER-Funktion definiert werden, wie in Gleichung 1 gezeigt.

$$a \oplus b = a\bar{b} + \bar{a}b \tag{1}$$

Kommutativität und Assoziativität Die UND-, ODER- und Exklusiv-ODER-Operatoren sind austauschbar.

$$\begin{aligned} a * b &= b * a \\ a + b &= b + a \\ a \oplus b &= b \oplus a \end{aligned} \tag{2}$$

Die UND-, ODER- und Exklusiv-ODER-Operatoren sind assoziativ.

$$\begin{aligned} (a * b) * c &= a * (b * c) = a * b * c \\ (a + b) + c &= a + (b + c) = a + b + c \\ (a \oplus b) \oplus c &= a \oplus (b \oplus c) = a \oplus b \oplus c \end{aligned} \tag{3}$$

4.2 Neutrale und absorbierende Elemente

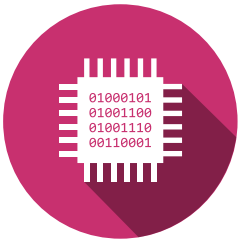
'1' ist das neutrale Element der UND-Operation.

$$a * 1 = a \tag{4}$$

'0' ist das absorbierende Element der UND-Operation.

$$a * 0 = 0 \tag{5}$$

'0' ist das neutrale Element der ODER-Operation.



$$a + 0 = a \tag{6}$$

'1' ist das absorbierende Element der ODER-Operation.

$$a + 1 = 1 \tag{7}$$

'0' ist das neutrale Element der Exklusiv-ODER-Operation.

$$a \oplus 0 = a \tag{8}$$

Mit einem UND-Gatter kann ein Signal übertragen oder sein Wert auf '0' gesetzt werden. Mit einem ODER-Gatter kann ein Signal übertragen oder sein Wert auf '1' gesetzt werden. Mit einem Exklusiv- ODER-Gatter kann ein Signal übertragen oder invertiert werden.

4.3 Distributivität

Die UND- und ODER-Operatoren sind zueinander distributiv.

$$\begin{aligned} a * (b + c) &= a * b + a * c \\ a + (b * c) &= (a + b) * (a + c) \end{aligned} \tag{9}$$

4.4 Theoreme von De Morgan

Dank den Theoremen von De Morgan kann ein UND-Operator durch einen ODER-Operator ersetzt werden.

$$\begin{aligned} \overline{a * b * c} &= \bar{a} + \bar{b} + \bar{c} \\ \overline{a + b + c} &= \bar{a} * \bar{b} * \bar{c} \end{aligned} \tag{10}$$

4.5 Exklusiv-ODER und Inversionen

Der Exklusiv-ODER-Operator macht Aussagen über die Parität: wenn die Anzahl Eingänge bei '1' ungerade ist, gibt die Funktion eine '1' zurück, wenn die Anzahl Eingänge bei '1' gerade ist, gibt die Funktion eine '0' zurück.

Kehrt man 2 oder eine gerade Anzahl Eingänge eines Exklusiv-ODER-Operators um, wird dadurch der Ausgang nicht verändert. Kehrt man hingegen 1 oder eine ungerade Anzahl Eingänge eines Exklusiv-ODER-Operators um, wird auch der Ausgang umgekehrt. Man kann somit eine gerade Anzahl Eingangs- oder Ausgangsleitungen eines Exklusiv-ODER-Operators umkehren, ohne dadurch die Funktion zu verändern. Dies ist in der Abbildung 12 dargestellt.

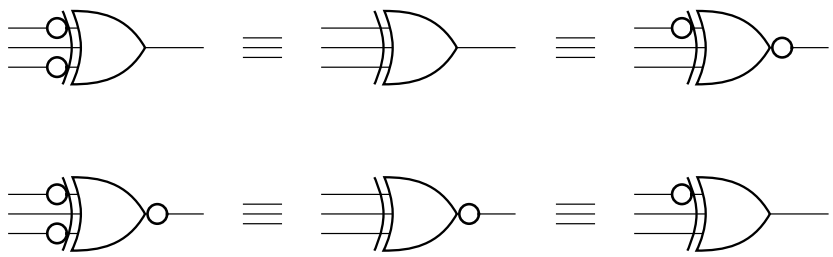


Abbildung 12: Exklusiv-ODER und Inversionen

4.6 Definition: Polynomische Form

Mit einer polynomischen Form kann eine kombinatorische Logikfunktion mit einer Summe von Produkten ausgedrückt werden.

Das Polynom besteht aus seinen Gliedern, die untereinander durch eine ODER-Funktion verbunden sind. Jedes Glied besteht aus Variablen, die untereinander durch eine UND-Funktion verbunden sind. Die Variablen können in invertierter oder nicht invertierter Form auftreten.

Beispiel

Die polynomische Form der Funktion $y = a \oplus b \oplus c$ ist $y = \bar{a}\bar{b}c + \bar{a}b\bar{c} + a\bar{b}\bar{c} + abc$.

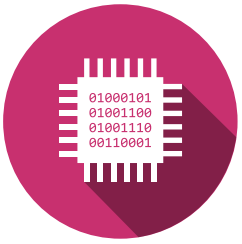
4.7 Definition: Monom

Ein Monom ist ein Glied eines Polynoms.

Beispiel

Das Polynom $ab + \bar{a}c + b\bar{c}$ besteht aus den Monomen ab, $\bar{a}c$ und $b\bar{c}$.

4.8 Zusammenfassung Boolean Algebra



Topic	Rules
Misc	$\overline{0} = 1$ $\overline{1} = 0$ $\overline{(\overline{a})} = a$
AND	$a * 0 = 0$ $a * 1 = a$ $a * a = a$ $a * \overline{a} = 0$ $(a * b) * c = a * (b * c) = a * b * c$
OR	$a + 0 = a$ $a + 1 = 1$ $a + a = a$ $a + \overline{a} = 1$ $(a + b) + c = a + (b + c) = a + b + c$
XOR	$a \oplus b = a\overline{b} + \overline{a}b$
Commutativity	$a * b = b * a$ $a + b = b + a$
Associativity	$(a * b) * c = a * (b * c) = a * b * c$ $(a + b) + c = a + (b + c) = a + b + c$
Distributivity	$a * (b + c) = (a * b) + (a * c)$ $a + (b * c) = (a + b) * (a + c)$
De Morgan	$\overline{(a * b)} = \overline{a} + \overline{b}$ $\overline{(a + b)} = \overline{a} * \overline{b}$
Absorption	$a + (a * b) = a$ $a * (a + b) = a$ $a + (\overline{a} * b) = a + b$ $a * (\overline{a} + b) = a * b$

Tabelle 3: Zusammenfassung Boolean Algebra

Literatur

[1] Suhail Almani. *Electronic Logic Systems*. second edition. New-Jersey: Prentice-Hall, 1989.

[2] Jean Michel Bernard und Jean Hugon. *Pratique Des Circuits Logiques*. quatrième édition. Paris: Eyrolles, 1987.

[3] Klaus Elektronik Bd Beuth. *Digitaltechnik*. Vogel Buchverlag, 1990. ISBN: 3-8023-1755-6.

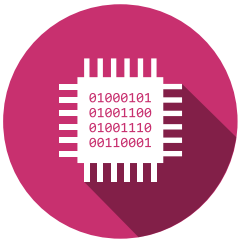
[4] Michael D. Ciletti und M. Morris Mano. *Digital Design*. second edition. New-Jersey: Prentice-Hall, 2007.

[5] David J. Comer. *Digital Logic and State Machine Design*. Saunders College Publishing, 1995.

[6] Donald L. Dietmeyer und R. David. "Logic Design of Digital Systems". In: *Journal of Dynamic Systems, Measurement, and Control* 94.2 (1. Juni 1972), S. 174–174. ISSN: 0022-0434. DOI: 10.1115/1.3426575. URL: https://doi.org/10.1115/1.3426575 (besucht am 01. 06. 2021).

[7] William I. Fletcher. *Engineering Approach to Digital Design*. New-Jersey: Prentice-Hall, 1980.

[8] Randy H. Katz und Gaetano Borriello. *Contemporary Logic Design*. California: The Benjamin/Cummings Publishing Company Inc, 2005.



[9] Martin V. Künzli und Marcel Meli. *Vom Gatter Zu VHDL: Eine Einführung in Die Digitaltechnik*. vdf Hochschulverlag AG, 2007. ISBN: 3 7281 2472 9.

[10] David Lewin und Douglas Protheroe. *Design of Logic Systems*. second edition. Hong Kong: Springer, 2013.

[11] Daniel Mange. *Analyse et synthèse des systèmes logiques*. Editions Géorgi. Bd. Traité d'électricité, volume V. St Saphorin: PPUR presses polytechniques, 1995. 362 S. ISBN: 978-2-88074-045-0. Google Books: 5NSdD4GRl3cC.

[12] Clive Maxfield. *Bebop to the Boolean Boogie*. Elsevier, 2009. ISBN: 978-1-85617-507-4. DOI: 10.1016/B978-1-85617-507-4.X0001-0. URL: https://linkinghub.elsevier.com/retrieve/pii/B9781856175074X00010 (besucht am 27. 05. 2021).

[13] Ronald J. Tocci und André Lebel. *Circuits Numériques: Théorie et Applications*. deuxième édition. Ottawa: Editions Reynald Goulet inc. / Dunod, 1996.

[14] John F. Wakerly. *Digital Design: Principles And Practices*. 3rd edition. Prentice-Hall, 2008. ISBN: 0-13-082599-9.

Akronyme

LUT **L**ook-**U**p **T**able. 6, 7

PAL **P**rogrammable **A**rray **L**ogic. 7