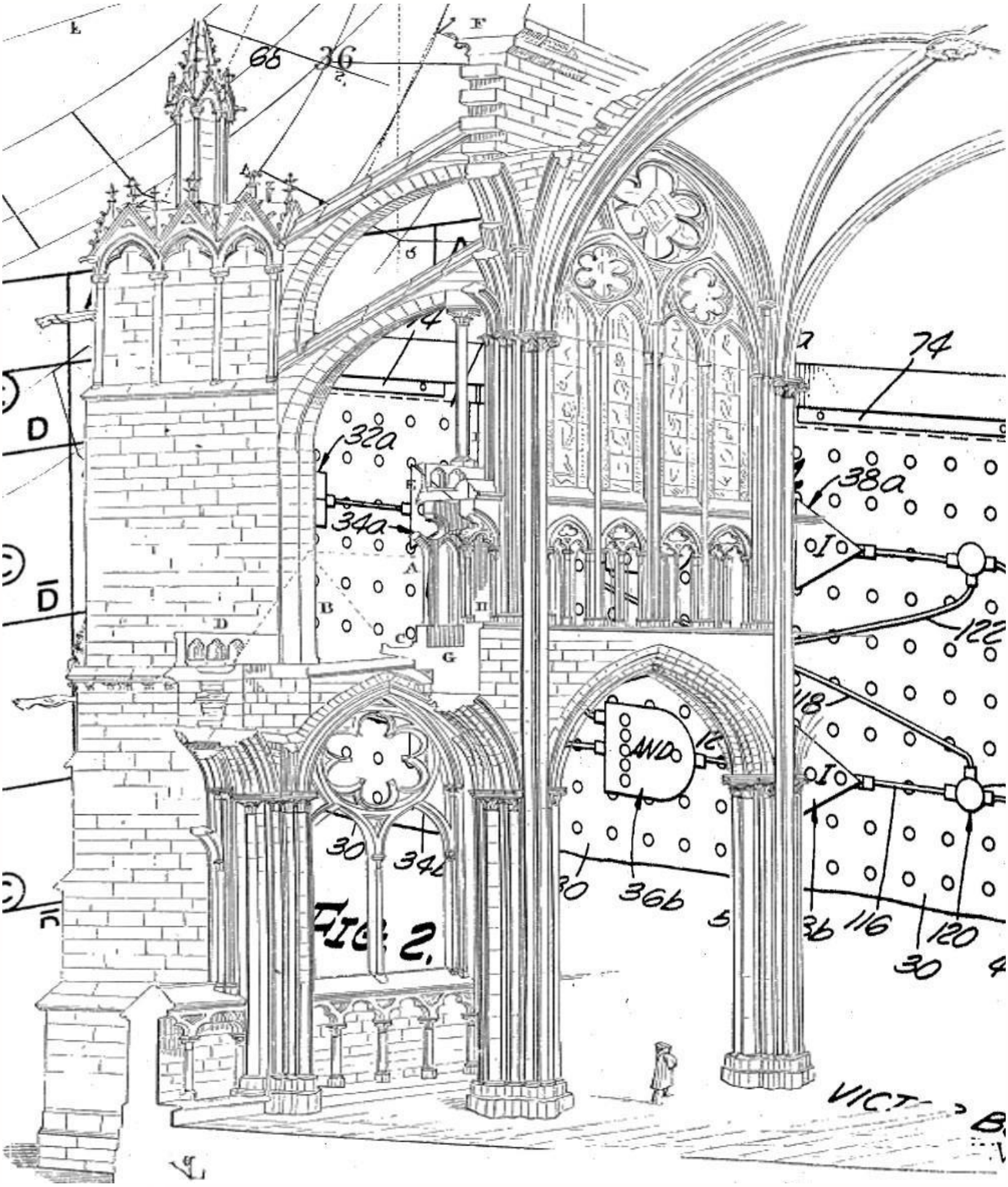
Cheatsheet

Grundlagen der Rechnerarchitektur

Von der Schaltung zum Prozessor



Fassung 21. November 2019

Satz: PDF-LaTeX 2_ε

Axiome von Peano

Die auf algebraischen Strukturen aufbauende Boolesche Algebra ermöglicht die Überführung der Logik in einen streng formalen mathematischen Formalismus und ein Schlussfolgern mittels Rechnen. Die Realisierung logischer Formeln mit Schaltungen erfordert allerdings weitere Vereinfachungen, um den Schaltungsaufwand zu reduzieren. Um die Boolesche Algebra einfach zugänglich zu machen, ist es notwendig, diese zunächst zu axiomatisieren. Die erste Axiomatisierung einer Booleschen Algebra gelang dem italienischen Mathematiker Guiseppe Peano (1858 - 1932) Ende des 19. Jahrhunderts.

Erfüllt die algebraische Struktur $(B,\cdot,+,0,1)$ die folgenden Axiome, ist sie eine Boolesche Algebra:

▶ Kommutativität:	(P1)	$a\cdot b$	=	$b\cdot a$
	(P1')	$a+b$	=	$b+a$
▶ Assoziativität:	(P2)	$(a\cdot b)\cdot c$	=	$a\cdot(b\cdot c)$
	(P2')	$(a+b)+c$	=	$a+(b+c)$
▶ Idempotenz:	(P3)	$(a\cdot a)$	=	a
	(P3')	$(a+a)$	=	a
▶ Distributivität:	(P4)	$a\cdot(b+c)$	=	$(a\cdot b)+(a\cdot c)$
	(P4')	$a+(b\cdot c)$	=	$(a+b)\cdot(a+c)$
▶ Neutralität:	(P5)	$a\cdot 1$	=	a
	(P5')	$a+0$	=	a
▶ Extremität:	(P6)	$a\cdot 0$	=	0
	(P6')	$a+1$	=	1
▶ Doppelnegation:	(P7)	$\overline{\overline{a}}$	=	a
▶ DeMorgan:	(P8)	$\overline{a\cdot b}$	=	$\overline{a}+\overline{b}$
	(P8')	$\overline{a+b}$	=	$\overline{a}\cdot\overline{b}$
▶ Komplementarität:	(P9)	$a\cdot\overline{a}$	=	0
	(P9')	$a+\overline{a}$	=	1
▶ Dualität:	(P10)	$\overline{0}$	=	1
	(P10')	$\overline{1}$	=	0
▶ Absorption:	(P11)	$a+(a\cdot b)$	=	a
	(P11')	$a\cdot(a+b)$	=	a

Axiome von Huntington

Die heute gebräuchliche Form der Axiomatisierung geht auf Edward Huntington (1874-1952) zurück. Dabei beschrieb Peano die Boolsche Algebra als Erster mithilfe zweier zweistelliger Verknüpfungen Verknüpfungen (*UND*: $\wedge$, *ODER*: $\vee$) sowie einer einstelligen Verknüpfung (*NICHT*: $\neg$) und Huntington konnte zeigen, dass jede Boolesche Formel auf eine Verknüpfung zweistelliger Operationen zurückgeführt werden kann. Beide Tatsachen bilden die formale Grundlage, um Operationen der Boolschen Funktionen in technische Schaltungen zu überführen.

Eine Boolesche Algebra ist eine algebraische Struktur $(B, +, \cdot, 0, 1)$, die die folgenden Axiome erfüllt:

▶ Kommutativität:	(H1)	$a + b$	$=$	$b + a$
	(H1')	$a \cdot b$	$=$	$b \cdot a$
▶ Distributivität:	(H2)	$a + (b \cdot c)$	$=$	$(a + b) \cdot (a + c)$
	(H2')	$a \cdot (b + c)$	$=$	$(a \cdot b) + (a \cdot c)$
▶ Identität:	(H3)	$a + 0$	$=$	a
	(H3')	$a \cdot 1$	$=$	a
▶ Komplementarität:	(H4)	$a + \bar{a}$	$=$	1
	(H4')	$a \cdot \bar{a}$	$=$	0

Zu jedem Element $a \in B$ existiert also ein Element $\bar{a} \in B$. An dieser Stelle sei daran erinnert, dass 0 die größte untere Schranke und 1 die kleinste obere Schranke der Trägermenge B ist.

Shannon-Zerlegung

Eine fundamentale Eigenschaft boolescher Funktionen wird durch die Shannon-Zerlegung oder den Satz von Shannon beschrieben. Die Shannon- Zerlegung wird zwar Claude Shannon (1916-2001) zugeschrieben, war jedoch schon George Boole (1815 - 1864) bekannt. Der Satz von Shannon wird in zahlreichen Algorithmen zur Optimierung, Synthese und Verifikation logischer Schaltungen eingesetzt, da er erlaubt, komplexe boolesche Funktionen durch Zerlegung zu vereinfachen. Mithilfe rekursiver Algorithmen kann das Problem nach dem Prinzip „Teile und herrsche “ immer weiter reduziert werden.

Ausgehend von der Schreibweise

$$f_{\bar{x}_1} = f_{|x_1=0} = f(0, x_2, \ldots, x_n)$$

$$f_{x_1} = f_{|x_1=1} = f(1, x_2, \ldots, x_n)$$

werden die Funktionen

$$f_{\bar{x}_1}(x_2, \ldots, x_n)$$

$$f_{x_1}(x_2, \ldots, x_n)$$

der positive und der negative Kofaktor von f bezogen auf x_1 genannt.

Positive und negative Kofaktoren einer booleschen Funktion können für jede Variable $x_1, \ldots, x_n$ der booleschen Funktion angegeben werden.

Wenn $f : B^n \mapsto B$ eine boolesche Funktion ist, dann gilt für die disjunktive Normalform

$$f(x_1, \ldots, x_i, \ldots, x_n) = \bar{x}_i \cdot f(x_1, \ldots, 0, \ldots, x_n) + x_i \cdot f(x_1, \ldots, 1, \ldots, x_n)$$

sowie analog für die konjunktive Normalform

$$f(x_1, \ldots, x_i, \ldots, x_n) = [\bar{x}_i + f(x_1, \ldots, 0, \ldots, x_n)] \cdot [x_i + f(x_1, \ldots, 1, \ldots, x_n)]$$

für alle $(x_1, \ldots, x_n) \in B$.

Abkürzend lässt sich sich für die disjunktive Normalform

$$f(x) = \bar{x}_i \cdot f_{i,0}(x) + x_i \cdot f_{i,1}(x)$$

sowie für die konjunktive Normalform

$$f(x) = [\bar{x}_i + f_{i,0}(x)] \cdot [x_i + f_{i,1}(x)]$$

schreiben.

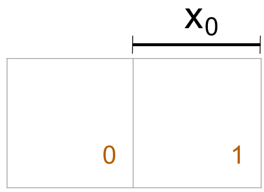
Karnaugh-Veitch

Konstruierte boolesche oder logische Karten heißen Karnaugh-Veitch-Diagramm (KV-Diagramm). In der englischen Literatur heißen diese Tafeln oft nur Karnaugh-Diagramm (Karnaugh map). Das Karnaugh-Veitch-Diagramm wurde von dem amerikanischen Informatiker Edward W. Veitch 1952 vorgeschlagen und vom Physiker Maurice Karnaugh 1953 verfeinert. Karnaugh-Veitch-Diagramme werden in der deutschen Literatur häufig Karnaugh-Tafel genannt.

Anmerkung: das jeweilige dem Funktionswert aus der Wahrheitstabelle zugehörige Feld wird im KV-Diagramm durch Angabe der entsprechenden Dezimalzahl gekennzeichnet.

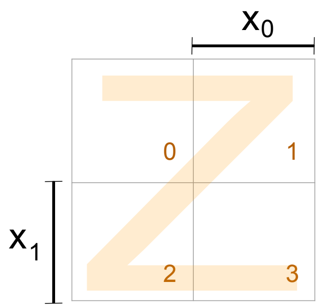
▶ KV-Diagramm für 1 Variable:

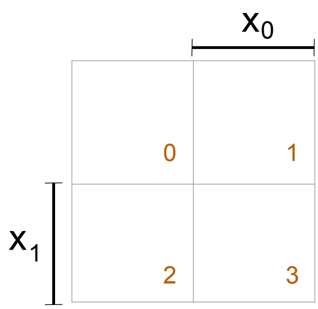
Dezimal	x_0
0	0
1	1



▶ KV-Diagramm für 2 Variablen:

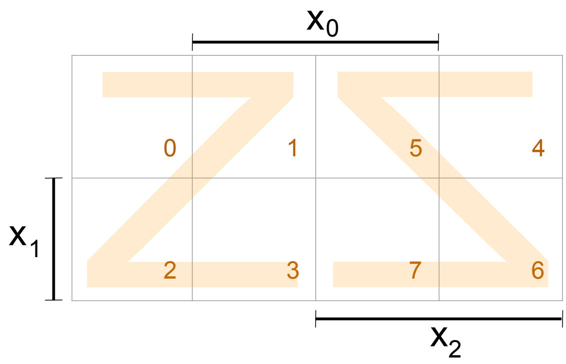
Dezimal	x_1	x_0
0	0	0
1	0	1
2	1	0
3	1	1

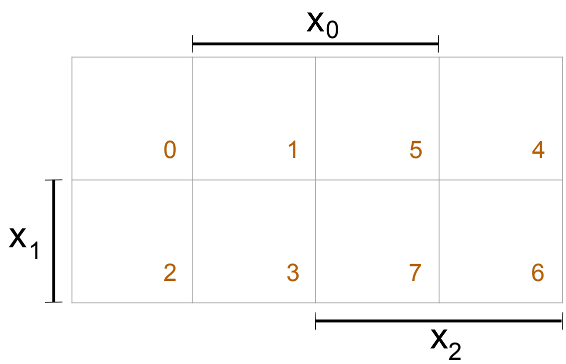




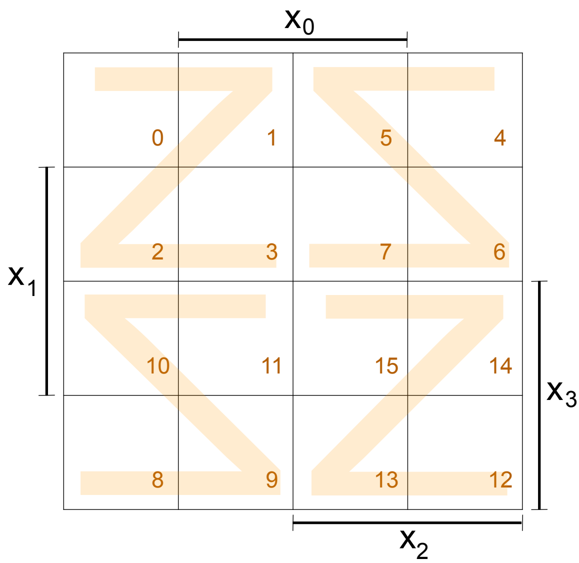
▶ KV-Diagramm für 3 Variablen:

Dezimal	x_2	x_1	x_0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1





▶ **KV-Diagramm für 4 Variablen:**



Dezimal	x_3	x_2	x_1	x_0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1