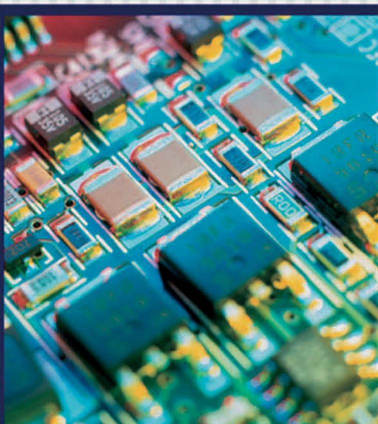
informatik



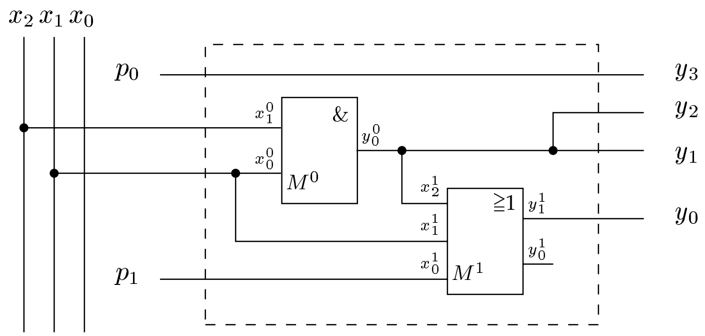
Heinz-Dietrich Wuttke
Karsten Henke

Schaltsysteme

Eine automatenorientierte Einführung

Pearson Studium

Bild 3.27 Strukturbeschreibung



Die prinzipiell möglichen Koppelbeziehungen für diese Strukturelemente können allgemein durch

$$\kappa: \quad \{x_1^0, x_0^0\} \cup \{x_2^1, x_1^1, x_0^1\} \cup \{y_3, y_2, y_1, y_0\} \Rightarrow \{y_0^0\} \cup \{y_1^1, y_0^1\} \cup \{x_2, x_1, x_0\} \cup \{p_1, p_0\}$$

beschrieben und entsprechend der konkreten Verbindungen als

$\kappa(x_1^0) = x_2$ sprich: »*Die Eingangsvariable x_1 des Moduls M^0 ist gekoppelt mit der Eingangsvariablen x_2 des Eingangsvektors x.*«

$\kappa(x_0^0) = x_1$	$\kappa(y_2) = y_0^0$
$\kappa(x_2^1) = y_0^0$	$\kappa(y_1) = y_0^0$
$\kappa(x_1^1) = x_1$	$\kappa(y_0) = y_1^1$
$\kappa(x_0^1) = p_1$	$\kappa(y_3) = p_0$

notiert werden.

3.5.2 Modulverkettung

Die oben definierte Koppelfunktion beschreibt die Kopplung jeweils zweier Module. Für Aussagen zur Gesamtstruktur ist auch von Interesse, auf welchem Weg die Belegung einer Eingangsvariablen (das »**Eingangssignal**«) den Ausgang erreicht, d.h. welche Module durchlaufen werden. Zur Beschreibung bedienen wir uns einiger Grundbegriffe der Graphentheorie, die im Anhang erläutert sind. Aus dieser Sicht sind die Module die *Knoten* des Graphen, die Verbindungen zwischen je zwei Knoten die *Kanten*. Die in diesem Kapitel betrachteten **kombinatorischen Strukturen** bilden je Ausgangsvariable eine **Baumstruktur**, in der die Ausgangsvariable die Wurzel und die Eingangsvariablen die Blätter darstellen. Eine Kantenfolge von einem Eingang (Blatt des Baumes) zum Ausgang der Struktur (Wurzel) heißt **Pfad** oder **Signalweg**. Die Module eines Signalweges (ohne Negatoren) heißen **Schaltstufen**. Die Anzahl der Schaltstufen definiert die **Pfadlänge**.

Die Definition der Koppelfunktion erlaubt neben der Beschreibung von Baumstrukturen nach Bild 3.27, die keine **Rückführungen** enthalten, auch die Beschreibung von Strukturen mit Rückführungen. Rückführungen sind aus graphentheoretischer Sicht Zyklen (oder Schleifen) in der Struktur. Schaltungsstrukturen mit Rückführungen zeichnen sich entsprechend dadurch aus, dass der Ausgang mindestens eines Moduls direkt oder indirekt auf dem Signalweg über mehrere Module einer Struktur mit einem Eingang desselben Moduls gekoppelt ist. Derartige Rückführungen haben qualitative Auswirkungen auf das Verhalten der Schaltung und sind Gegenstand des Kapitels 5 und folgender.

Uns interessiert an dieser Stelle zunächst eine systematische Beschreibung dieser Struktureigenschaft. Die bisher definierte Koppelfunktion beschreibt nur die Kopplung jeweils zweier Module. Wir müssen hier jedoch auch die Kopplung mehrerer auf einem Signalweg (Pfad im Graphen) liegenden Module beschreiben und definieren hierfür die **Modulverkettung** wie folgt:

Modulverkettung *Def. (3.7)*

$$M^m \vdash M^l$$

Zwei Module M^m und M^l heißen *verkettet*, wenn mindestens eine Koppelrelation der folgenden Form existiert:

$$\kappa(x_i^l) = y_j^m \qquad \text{mit } m \neq l \text{ und } x_i^l \in x^l \text{ und } y_j^m \in y^m$$

Die Verkettungsrelation ist irreflexiv und asymmetrisch. Die *transitive Hülle* der Kopplung zweier Module M^m, M^l soll mit einem Stern markiert werden und als

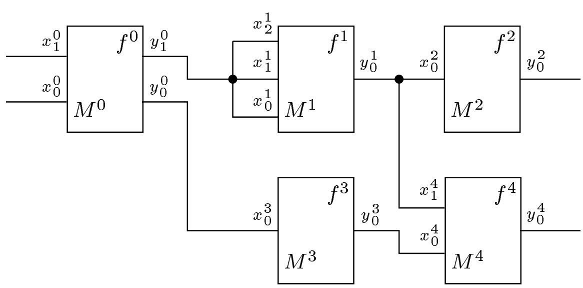
$$M^m \vdash^{*} M^l$$

beschrieben werden; eine dabei auftretende Folge von Kopplungen durch

$$\kappa^*(x_i^l) = y_j^m.$$

Anhand der Struktur in Bild 3.28 sollen diese Definitionen verdeutlicht werden.

Bild 3.28 Modulverkettung



Es gilt: $M^0 \vdash M^1,\ M^0 \vdash M^3,\ M^1 \vdash M^2,\ M^1 \vdash M^4,\ M^3 \vdash M^4,$
$M^0 \vdash M^1 \vdash M^4,\ M^0 \vdash M^3 \vdash M^4,\ M^0 \vdash^* M^2,\ M^0 \vdash^* M^4$

aber nicht: $M^1 \vdash^* M^3$

weiter gilt: $\kappa(x_2^1) = y_1^0,\ \kappa(x_1^4) = y_0^1,\ \kappa^*(x_1^4) = y_1^0$

Mithilfe dieser Beschreibungsmöglichkeit können wir die Unterschiede zwischen kombinatorischen und sequentiellen Strukturen nun formal beschreiben.

Verkettete Module bilden eine **kombinatorische Struktur** (Bild 3.29a), wenn für alle m, l gilt:

$$M^m \vdash^* M^l \rightarrow \nexists m, l(M^l \vdash^* M^m) \tag{3.113}$$

bzw.

$$M^m \vdash^* M^l \rightarrow \forall i, j \left(\nexists \kappa^*(x_i^m) = y_j^l\right)$$

Eine **sequentielle Struktur** (Bild 3.29b) enthält mindestens eine Kopplung κ, für die gilt:

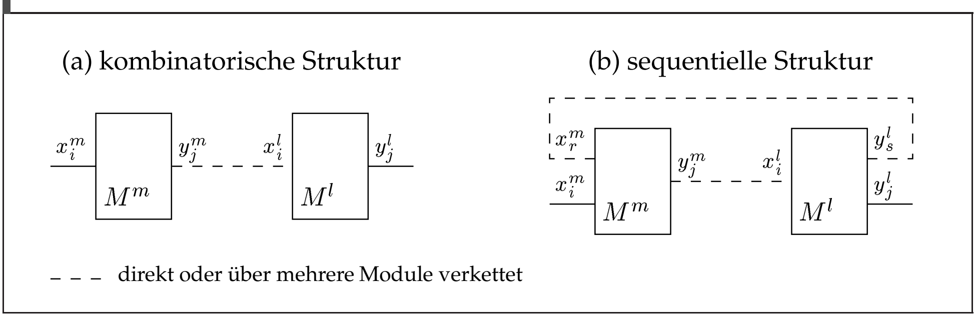
$$\exists m, l\left(\left(M^m \vdash^* M^l\right) \wedge \left(M^l \vdash^* M^m\right)\right) \tag{3.114}$$

bzw.

$$\exists m, l\Big(\left(\kappa^*(x_i^l) = y_j^m\right) \ \wedge\ \left(\kappa^*(x_r^m) = y_s^l\right)\Big)$$

Eine Kopplung mit der Eigenschaft (3.114) heißt **Rückkopplung**.

Bild 3.29 Kombinatorische und sequentielle Strukturen

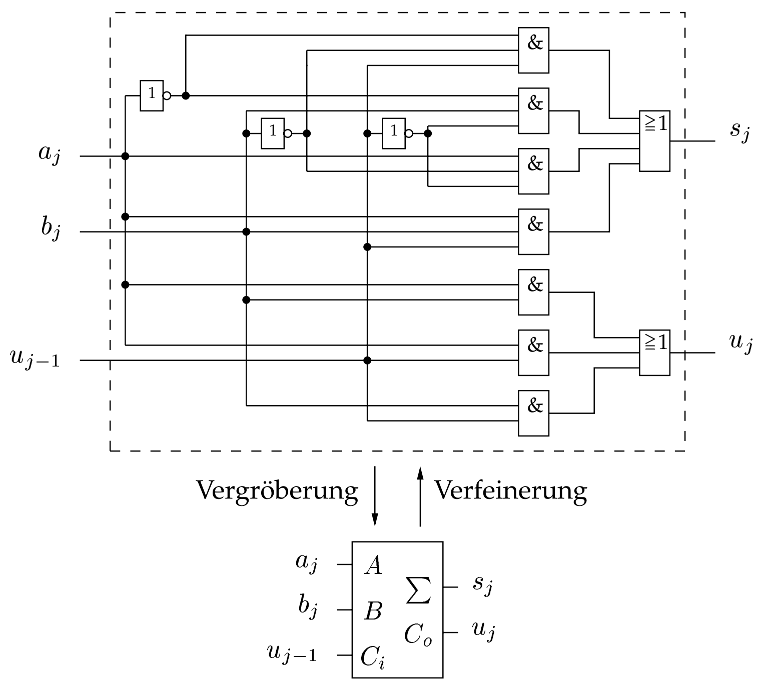


3.5.3 Hierarchie, Abstraktion

Die Darstellung komplexer Strukturen kann vereinfacht werden, indem man Module mit elementaren Funktionen zu komplexen Modulen zusammenfasst und deren innere Struktur verbirgt. Auf diese Weise kann man **Beschreibungshierarchien** erzeugen, die über mehrere Ebenen von der Grobstruktur (auch Blockschaltbild genannt) bis zum elementaren Gatterniveau reichen. Man abstrahiert dabei von Details, die für die Darstellung der jeweiligen Abstraktionsebene unwesentlich sind.

So kann beispielsweise die Funktion eines 8-Bit-Rechenwerkes für die Addition als einzelner Block, als Zusammenschaltung von acht einzelnen Addierschaltungen oder als detaillierte Gatterstruktur dargestellt werden. Bild 3.30 zeigt einen Volladder, dessen Funktionalität wir in Abschnitt 3.1 kennen gelernt haben, in zwei verschiedenen Abstraktionsebenen. Den Übergang von der Detailstruktur zu einer abstrahierten Darstellung nennt man auch **Vergröberung**, das umgekehrte Verfahren **Verfeinerung**.

Bild 3.30 Verschiedene Abstraktionsebenen eines Volladders



Bei der Vergröberung werden ausgewählte Ein- und Ausgänge der feineren Struktur als Schnittstelle x', y' des vergröberten Moduls M' definiert. Die Funktion f' des Moduls M' beschreibt das Verhalten des Moduls mithilfe der Schnittstellenvariablen ggf. unter Einbeziehung interner Zustandsvariablen (siehe Abschnitt 5.2). Man kann diese Hierarchie-Ebenen auch für einen systematischen Entwurf nutzen. Eine

Methode, die auf schrittweiser Verfeinerung beruht, heißt *Top-Down-Methode*, die umgekehrte Vorgehensweise einer schrittweisen Vergröberung *Bottom-Up-Methode*.

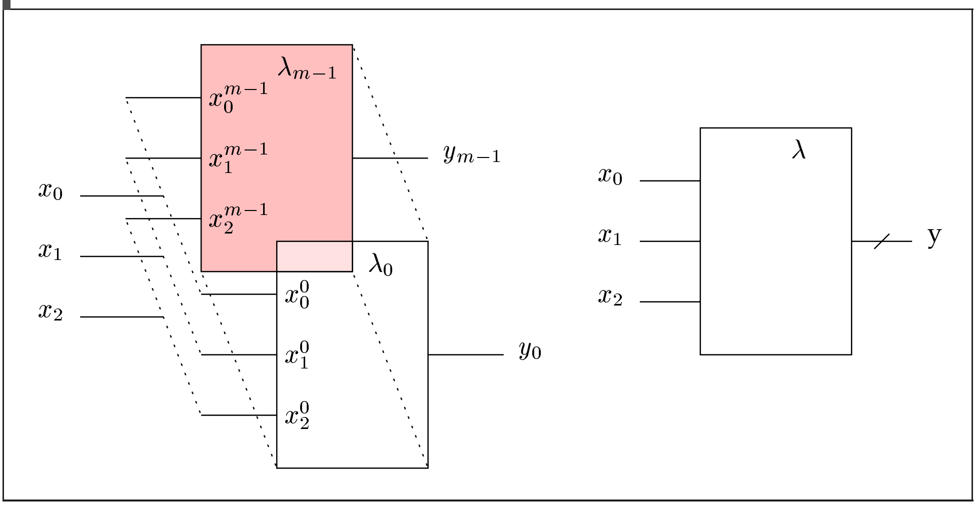
3.5.4 Blockbildung, Kaskadierung

Zur Realisierung digitaler Schaltungen werden Schaltkreise verwendet, die eine bestimmte Anzahl von Modulen mit einer festgelegten Anzahl von Ein- und Ausgängen beinhalten. Reicht die Anzahl der so vorgegebenen Eingänge bzw. Ausgänge für die angestrebte Realisierung nicht aus, so können unter Verwendung der gleichen Bausteine und zusätzlicher schaltungstechnischer Ergänzungen Strukturen mit der gesuchten Anzahl von Ein- bzw. Ausgängen realisiert werden. Die Erweiterung der Ausgänge realisiert man durch so genannte Blockbildungen, die Erweiterung der Eingänge durch eine Kaskadierung.

3.5.4.1 Blockbildung

Entsprechend der Definition der Koppelfunktion κ dürfen Eingänge der Struktur bzw. Ausgänge von Modulen beliebig vielen Eingängen von Modulen zugeordnet werden. Für eine Erweiterung der Ausgänge einer Struktur ist es dementsprechend ausreichend, die Struktur mehrfach zu verwenden und mit denselben Eingangssignalen zu versorgen. Wir werden in Abschnitt 3.6 Strukturen diskutieren, die man mithilfe der Blockbildung zu komplexeren Strukturen zusammensetzt. Bild 3.31 zeigt das Prinzip der Blockbildung sowie die symbolische Darstellung eines Blockes im rechten Teil des Bildes. Während die Module des Blockes jeweils nur eine Ausgangsvariable y_k besitzen und eine Teilkomponente λ_k realisieren, gehört zum Block der Ausgangsvektor y mit m Ausgangsvariablen. Der Ausgangsvektor wird in der Zeichnung als Leitungsbündel (sog. »Bus«) dargestellt, symbolisiert durch einen Querstrich auf der y-Leitung.

Bild 3.31 Blockbildung

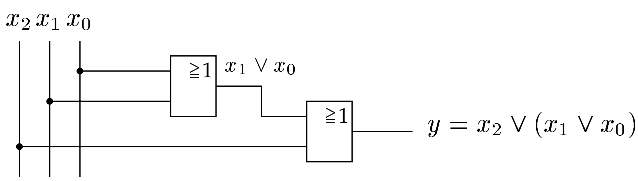


3.5.4.2 Kaskadierung

Ist die Anzahl der Eingänge eines Moduls für die Realisierung einer bestimmten Funktionalität nicht ausreichend, so kann man diese mithilfe der **Kaskadierung** erweitern. Eine einfache Methode der Kaskadierung besteht darin, mehrstufige Schaltungen zu realisieren, die jeweils aus gleichen Gattern aufgebaut sind.

Um beispielsweise eine ODER-Funktion für drei Eingangsvariablen mit Strukturelementen aus Tabelle 3.17 zu realisieren, verknüpft man – wie in Bild 3.32 dargestellt – in einer ersten Stufe zwei Variablen disjunktiv und führt den Ausgang dieses ODER-Gatters mit der dritten Eingangsvariablen in einer zweiten Stufe in einem weiteren ODER-Gatter mit zwei Eingängen zusammen.

Bild 3.32 Einfache Kaskadierung



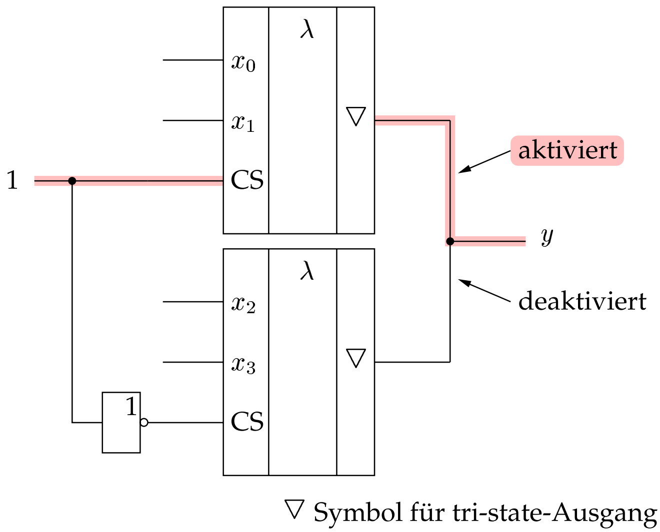
Für eine andere Variante der Kaskadierung werden Schaltkreise mit so genanntem *tri-state-Verhalten* genutzt. Das bei der Kopplung von Modulen streng verbotene Verbinden von Ausgängen wird dadurch möglich. Tri-state-Ausgänge werden deshalb besonders gekennzeichnet (siehe Bild 3.33). Es ist ein häufig verwendetes Prinzip der Kopplung und wird auch zur Mehrfachnutzung von Leitungen für den zeitlich versetzten Transport von unterschiedlichen Informationen beispielsweise beim Adress-, Daten- und Steuerbus eines Digitalrechners genutzt. Wir wollen dieses Prinzip kurz erklären, ohne dabei auf die schaltungstechnischen Details einzugehen.

Das Prinzip des **tri-state-Verhaltens** besteht darin, Module über einen Steuereingang mit einem Auswahlsignal »chip select« (CS[27]) in einen so genannten »dritten Zustand« am Ausgang (neben den sonst üblichen Zuständen 0 und 1) zu schalten. In diesem Zustand hat der Ausgang des Moduls einen sehr hohen Widerstand (man sagt »*Er ist hochohmig*«), was sich wie eine Leitungsunterbrechung am Ausgang auswirkt. Der Ausgang ist inaktiv und unempfindlich gegenüber Signalen eines anderen Ausgangs, der zu diesem Zeitpunkt die gleiche Leitung aktiv nutzt. Selbstverständlich muss man sicherstellen, dass jeweils nur *ein* an der gleichen Leitung angeschlossener Ausgang aktiv sein kann (»exklusiver Zugriff«). Man gewährleistet dieses Verhalten mithilfe von Dekoder- oder Multiplexer-Modulen, die wir in Abschnitt 3.6.4 beschreiben. Hier wollen wir nur die einfache Variante der

[27] Der Einfachheit halber nehmen wir an, das Signal würde mit 1 aktiviert. Der CS-Eingang realer Schaltkreise ist meist »*low-aktiv*«, d.h. wird mit 0 aktiviert und deshalb häufig mit /CS bezeichnet.

Kaskadierung zweier Module darstellen, bei der wir die Exklusivität des Leitungszugriffs über die Negation des Steuersignals CS sicherstellen.

Bild 3.33 Kaskadierung



Mithilfe der in Abschnitt 3.5 erläuterten strukturellen Kopplungs- und Umformungsmöglichkeiten können aus elementaren Strukturen beliebige Strukturen aufgebaut werden. Im nächsten Abschnitt gehen wir auf eine Reihe von häufig verwendeten Strukturen ein, deren Kenntnis das Verständnis komplexer Schaltungen erleichtert.

3.6 Synthese und Beispiele kombinatorischer Strukturen

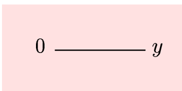
3.6.1 Elementare Funktionen und Strukturen mit zwei Variablen

An dieser Stelle erinnern wir an die Einführung der BAA in Abschnitt 3.2.3. Am Beispiel aller Funktionen mit zwei Eingangsvariablen haben wir erläutert, dass diese Algebra mithilfe von Repräsentanten je einer unendlichen Menge wertverlaufsgleicher Ausdrücke definiert wird. Diese Repräsentanten haben auch für die Bestimmung einer Menge elementarer Strukturen digitaler Schaltungen Bedeutung, da aus ihnen alle weiteren Strukturen aufgebaut werden können. Sie werden auch als Gatter bezeichnet und sind die Grundbausteine des »Baukastensystems« für digitale Schaltungen.

Tabelle 3.17 gibt einen Überblick über diese Funktionen und zugehörige Strukturelemente. Jeweils einer der Ausdrücke in DNF, KNF oder einer weiteren Form kann dabei als Repräsentant für die jeweilige Funktion dienen.

Tabelle 3.17 Elementare Funktionen und Strukturen

Null



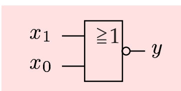
x_1	x_0	y_0
0	0	0
0	1	0
1	0	0
1	1	0

DNF: 0

KNF: 0

weitere NF: 0

NOR
(not or)

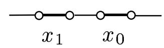


x_1	x_0	y_1
0	0	1
0	1	0
1	0	0
1	1	0

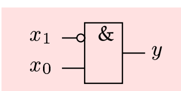
DNF: $\overline{x_1}\,\overline{x_0}$

KNF: $\overline{x_1}\,\overline{x_0}$

weitere NF: $\overline{x_1 \vee x_0}$



Inhibition
(von x_0 auf x_1)

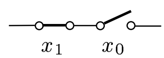


x_1	x_0	y_2
0	0	0
0	1	1
1	0	0
1	1	0

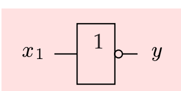
DNF: $\overline{x_1}\,x_0$

KNF: $\overline{x_1}\,x_0$

weitere NF: $\overline{x_0 \to x_1}$



NOT
(Negation von x_1)

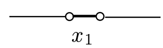


x_1	x_0	y_3
0	0	1
0	1	1
1	0	0
1	1	0

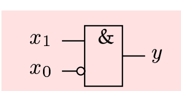
DNF: $\overline{x_1}$

KNF: $\overline{x_1}$

weitere NF: $\overline{x_1}$



Inhibition
(von x_1 auf x_0)

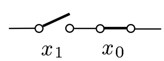


x_1	x_0	y_4
0	0	0
0	1	0
1	0	1
1	1	0

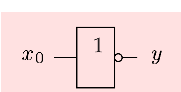
DNF: $x_1\,\overline{x_0}$

KNF: $x_1\,\overline{x_0}$

weitere NF: $\overline{x_1 \to x_0}$



NOT
(Negation von x_0)

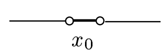


x_1	x_0	y_5
0	0	1
0	1	0
1	0	1
1	1	0

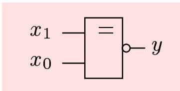
DNF: $\overline{x_0}$

KNF: $\overline{x_0}$

weitere NF: $\overline{x_0}$

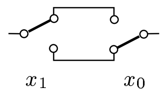


Antivalenz
(XOR, Exclusiv-Oder)

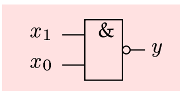


x_1	x_0	y_6
0	0	0
0	1	1
1	0	1
1	1	0

DNF: $x_1\,\overline{x_0} \vee \overline{x_1}\,x_0$
KNF: $(\overline{x_1} \vee \overline{x_0})(x_1 \vee x_0)$
weitere NF: $x_1 \not\sim x_0$

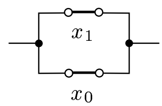


NAND
(not and)

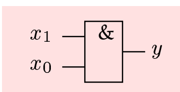


x_1	x_0	y_7
0	0	1
0	1	1
1	0	1
1	1	0

DNF: $\overline{x_1} \vee \overline{x_0}$
KNF: $\overline{x_1} \vee \overline{x_0}$
weitere NF: $\overline{x_1\,x_0}$

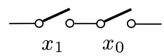


AND
(Konjunktion)

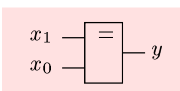


x_1	x_0	y_8
0	0	0
0	1	0
1	0	0
1	1	1

DNF: $x_1\,x_0$
KNF: $x_1\,x_0$
weitere NF: $x_1\,x_0$

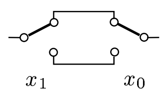


Äquivalenz



x_1	x_0	y_9
0	0	1
0	1	0
1	0	0
1	1	1

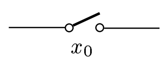
DNF: $x_1\,x_0 \vee \overline{x_1}\,\overline{x_0}$
KNF: $(\overline{x_1} \vee x_0)(x_1 \vee \overline{x_0})$
weitere NF: $x_1 \sim x_0$



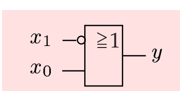
Identität
(von x_0)

x_1	x_0	y_{10}
0	0	0
0	1	1
1	0	0
1	1	1

DNF: x_0
KNF: x_0
weitere NF: x_0

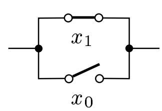


Implikation
(von x_1 auf x_0)

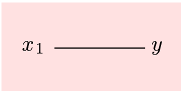


x_1	x_0	y_{11}
0	0	1
0	1	1
1	0	0
1	1	1

DNF: $\overline{x_1} \vee x_0$
KNF: $\overline{x_1} \vee x_0$
weitere NF: $x_1 \rightarrow x_0$

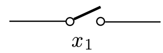


Identität
(von x_1)

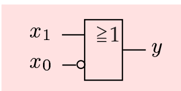


x_1	x_0	y_{12}
0	0	0
0	1	0
1	0	1
1	1	1

DNF: x_1
KNF: x_1
weitere NF: x_1

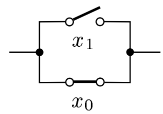


Implikation
(von x_0 auf x_1)

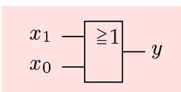


x_1	x_0	y_{13}
0	0	1
0	1	0
1	0	1
1	1	1

DNF: $x_1 \vee \overline{x_0}$
KNF: $x_1 \vee \overline{x_0}$
weitere NF: $x_0 \rightarrow x_1$

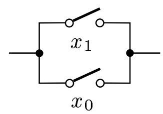


OR
(Disjunktion)

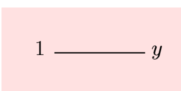


x_1	x_0	y_{14}
0	0	0
0	1	1
1	0	1
1	1	1

DNF: $x_1 \vee x_0$
KNF: $x_1 \vee x_0$
weitere NF: $x_1 \vee x_0$



Eins



x_1	x_0	y_{15}
0	0	1
0	1	1
1	0	1
1	1	1

DNF: 1
KNF: 1
weitere NF: 1

3.6.2 Struktursynthese

Aus den elementaren Strukturen nach Abschnitt 3.6.1 können beliebige kombinatorische Strukturen unter Beachtung der in Abschnitt 3.5 diskutierten Regeln zusammengesetzt (synthetisiert) werden. Ausgangspunkt der **Struktursynthese** ist ein schaltalgebraischer Ausdruck. Seine syntaktische Struktur (d.h. die Anzahl der Terme, die Art der Verknüpfung der Terme, Klammerung von Teiltermen usw.) bestimmt die Struktur der synthetisierten Schaltung. Stimmt die syntaktische Struktur eines Ausdrucks mit der Struktur der Schaltung überein, so heißt der Ausdruck **strukturgleicher Ausdruck**. Umgekehrt heißt eine Schaltung, deren Struktur der syntaktischen Struktur eines Ausdrucks entspricht, **strukturgleiche Schaltung**.

Zu jeder kombinatorischen Schaltung existiert ein eindeutig zuzuordnender strukturgleicher Ausdruck.

Bildet ein Ausdruck in Normalform (z.B. DNF oder KNF) den Ausgangspunkt der Synthese, so erhält man aufgrund der Zweistufigkeit des Ausdrucks (Disjunktion von Konjunktionstermen bzw. Konjunktion von Disjunktionstermen) eine so genannte **zweistufige Schaltungsstruktur**. Die erste Stufe verknüpft die Eingangsvariablen und die zweite Stufe verknüpft die Ausgänge der ersten Stufe zum Ausgangssignal der gesuchten Struktur.

Wir wollen die Struktursynthese am Beispiel eines in DNF vorliegenden Ausdrucks demonstrieren. Dabei greifen wir auf das Beispiel des Volladders zurück. Ausgehend von einem schaltalgebraischen Ausdruck werden den Termen des Ausdrucks Module (in unserem Beispiel UND-Gatter) zugeordnet, die die entsprechende Funktion (hier die Konjunktion) realisieren. Die Anzahl der Variablen im Term bestimmt dabei die Anzahl der notwendigen Eingänge des Moduls. Entsprechend den Verknüpfungszeichen zwischen den Termen (im Beispiel Disjunktionen) wird ein weiteres Modul benötigt, das den Ausgang der gesuchten Struktur liefert. Seine Eingänge werden mit den Ausgängen der Module der ersten Stufe verbunden.

In Bild 3.34 ist die strukturgleiche Schaltung des Volladders als Ergebnis der Synthese dargestellt, die sich aus den Gleichungen für die Summe s_j und den Übertrag u_j ergibt:

$$s_j = \underbrace{\underbrace{\overline{a_j}\,\overline{b_j}\,u_{j-1}}_{M^0} \vee \underbrace{\overline{a_j}\,b_j\,\overline{u_{j-1}}}_{M^1} \vee \underbrace{a_j\,\overline{b_j}\,\overline{u_{j-1}}}_{M^2} \vee \underbrace{a_j\,b_j\,u_{j-1}}_{M^3}}_{M^4} \tag{3.115}$$

$$\begin{aligned} u_j &= \overline{a_j}\,b_j\,u_{j-1} \vee a_j\,\overline{b_j}\,u_{j-1} \vee a_j\,b_j\,\overline{u_{j-1}} \vee a_j\,b_j\,u_{j-1} \\ &= \underbrace{\underbrace{a_j\,b_j}_{M^5} \vee \underbrace{a_j\,u_{j-1}}_{M^6} \vee \underbrace{b_j\,u_{j-1}}_{M^7}}_{M^8} \end{aligned} \tag{3.116}$$

Die Klammern fassen die den Modulen zugeordneten syntaktischen Struktureinheiten zusammen.

3.6.3 Basissysteme

In Abschnitt 3.2.5 wurde gezeigt, dass sich Funktionen kombinatorischer Schaltungen unter ausschließlicher Verwendung der Operatoren UND, ODER, NICHT realisieren lassen. Die DNF und die KNF mögen hierfür ein Beispiel sein. Andere Normalformen wurden, wie beispielweise die NAND-Normalform, unter ausschließlicher Verwendung nur eines Funktionselementes (der NAND-Funktion) beschrieben.

Für welche der Elementarfunktionen aus Tabelle 3.17 ist dies noch möglich? Woran erkennt man, dass eine Elementarfunktion als Basis für beliebige Realisierungen Boolescher Funktionen genutzt werden kann?

Diesen Fragen wollen wir in diesem Abschnitt nachgehen. Offensichtlich kann eine Menge von Elementarfunktionen als Basis für die Realisierung beliebiger kombina-