

Schaltnetze

Es heißt oft, „der Rechner arbeitet mit Einsen und Nullen“. Genau genommen arbeitet der Rechner mit Strom. Je nachdem, ob Strom durch ein Bauteil des Rechners fließt, können zwei Zustände unterschieden werden. Diese werden durch Eins (Strom fließt) bzw. Null (Strom fließt nicht) symbolisiert.

Wenn wir Daten mithilfe von Nullen und Einsen codieren, handelt es sich dabei also um ein geeignetes Modell, um sich zu überlegen, wie der Rechner Daten verarbeiten kann.

Wir sind es gewohnt, am Rechner mit Texten, Bildern oder Tönen zu arbeiten. Damit der Rechner diese Daten verarbeiten kann, muss man sich also überlegen, wie man diese mithilfe von Einsen und Nullen codiert. Vielleicht hast du schon Beispiele für solche Codierungen kennengelernt. Die Einsen und Nullen lassen sich wie unsere Ziffern von 0 bis 9 zu größeren Zahlen zusammensetzen. Wir befinden uns dann in einem anderen Zahlensystem, dem Binärsystem. Unsere Zahlen aus dem Dezimalsystem werden also in das Binärsystem umgerechnet. Auch Buchstaben oder Farben lassen sich mit Zahlen codieren. Beim ASCII-Code steht z. B. die Zahl 65 für den Buchstaben A und die Farbe Rot wird durch den RGB-Wert 255 0 0 beschrieben. Diese Zahlen können dann wieder ins Binärsystem übersetzt werden. Somit reichen die zwei Werte Eins und Null tatsächlich aus, um alle Daten zu codieren und zu verarbeiten.

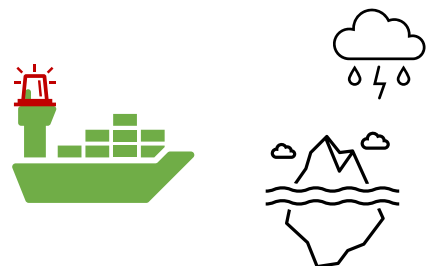
Natürlich sind Buchstaben oder ganze Bilder binär codiert für uns Menschen sehr unübersichtlich. Wir werden uns hier deshalb mit entsprechend einfachen Beispielen beschäftigen. Die Prinzipien der Verarbeitung sind dabei die gleichen, wie bei komplexeren, umfangreicheren Daten.

Wie werden Einsen und Nullen verarbeitet?

Logische Verknüpfungen

Die Tasten auf der Tastatur, die Tasten der Maus oder auch die Tasten A und B des Calliope mini können zwei Zustände annehmen: gedrückt (1) oder nicht gedrückt (0). In der technischen Informatik kann man die Eingaben mithilfe solcher Schalter symbolisieren, die zwischen 1 und 0 wechseln können.

Wir betrachten zunächst ein anschauliches Beispiel: Um ein Alarmsystem für eine Schiffsbesatzung zu konstruieren, weisen wir den Tasten A und B zwei verschiedene Aussagen zu. Taste A steht für „Es gibt eine Unwetterwarnung“ und Taste B steht für „Eisberg in Sicht“. Nun wäre die Frage, ob der Alarm ausgelöst wird, wenn bereits eine der Aussagen wahr



ist und die entsprechende Taste gedrückt wird oder erst, wenn beide Aussagen erfüllt sind. Das entspricht zwei verschiedenen logischen Verknüpfungen der Aussagen: *A oder B* bzw. *A und B*. Der Alarm wird also genau dann ausgelöst, wenn die Verknüpfung der beiden Aussagen wahr (1) ist.

In der technischen Informatik bezeichnet man Eingaben, die zwischen den Zuständen 0 und 1 wechseln können, als Schaltvariablen. Diese können mithilfe entsprechender Bauteile mit einem logischen **ODER** bzw. mit einem logischen **UND** verknüpft oder auch **negiert** (verneint) werden. Durch Kombination dieser logischen Operationen lassen sich dann auch komplexere Aussagen abbilden. Diese Verknüpfung der Eingaben mit logischen Operatoren entspricht der Verarbeitung. Das Ergebnis dieser Verknüpfung führt zu dem Ergebnis 1 oder 0, je nachdem ob die Gesamtaussage

wahr oder falsch ist. Tabelle 1 veranschaulicht die Zusammenhänge anhand des Alarmsystems für die Schiffsbesatzung. Eine solche Tabelle nennt man **Wahrheitstabelle**.



Die drei Operatoren UND, ODER bzw. NICHT reichen aus, um alle Daten im Rechner zu verarbeiten und alle denkbaren Schaltungen zu konstruieren.

	A „Eisberg in Sicht“	B „Unwetter- warnung“	A oder B	A und B	nicht A ¹	nicht A oder B
	A	B	$A \vee B$	$A \wedge B$	\bar{A}	$\bar{A} \vee B$
	0	0	0	0	1	1
	1	0	1	0	0	0
	0	1	1	0	1	1
	1	1	1	1	0	1

Tabelle 1: Beispiele für logische Verknüpfungen der Eingaben eines Alarmsystems

In der zweiten Zeile der Tabelle 1 siehst du die Notation der logischen Verknüpfungen in Form eines booleschen Ausdrucks². Im Zusammenhang mit technischen Schaltungen spricht man auch von einem **Schalterm** bzw. einer **Schaltfunktion**. In der Schaltfunktion Alarm = $A \wedge B$, wäre $A \wedge B$ der Schalterm.

Aufgabe 1:

a) Formuliere die logischen Verknüpfungen in Tabelle 1 in vollständigen Sätzen.

Beispiel: $A \wedge B$ steht für *Es ist ein Eisberg in Sicht und es gibt eine Unwetterwarnung*.

b) Ergänze in Tabelle 1 die Auswertung der folgenden Verknüpfungen:

I. Es ist ein Eisberg in Sicht und es gibt keine Unwetterwarnung.

II. Es ist entweder ein Eisberg in Sicht oder es gibt eine Unwetterwarnung.

Notiere die Aussagen I und II auch als booleschen Ausdruck.

Aufgabe 2: Wir betrachten die Aussagen A *Das Fahrzeug ist ein Auto* und B *Das Fahrzeug ist blau*.

Werte in Tabelle 2 die booleschen Ausdrücke aus und formuliere sie in vollständigen Sätzen.

	A	B	A oder B	nicht (A oder B)	A und nicht B	nicht (A oder B) und A
			$(A \vee B)$	$\overline{A \vee B}$	$A \wedge \bar{B}$	$(\overline{A \vee B}) \wedge A$

Tabelle 2: Wahrheitstabelle zu Aufgabe 2

¹ Für die Verneinung findet man anstelle eines Querbalkens über der Schaltvariablen auch die Schreibweise $\neg A$

² Die Bezeichnung geht auf George Boole zurück, der mit der Booleschen Algebra Mitte des neunzehnten Jahrhunderts die mathematische Grundlage für die Verwendung logischer Verknüpfungen gelegt hat.

Darstellungsformen für logische Verknüpfungen

Ähnlich wie es in der Mathematik für Funktionen unterschiedliche Darstellungsformen gibt (die Funktionsgleichung, den Funktionsgraphen oder eine Wertetabelle), können auch die logischen Verknüpfungen unterschiedlich dargestellt werden: als **Schaltterm** (boolescher Ausdruck), als **Wahrheitstabelle** oder als **Schaltung** aus Schaltsymbolen für die verwendeten Bauteile. Tabelle 3 zeigt die unterschiedlichen Darstellungsformen für die drei grundlegenden logischen Verknüpfungen **UND**, **ODER** bzw. **NICHT**.

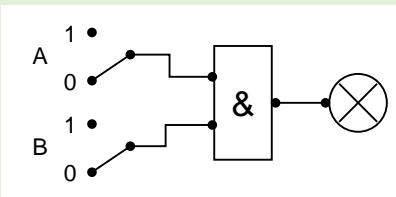
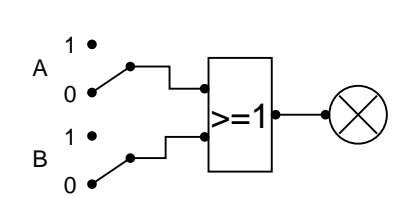
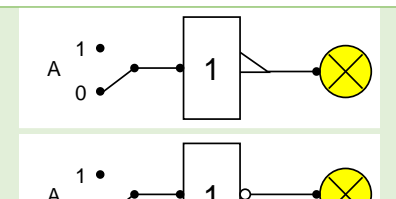
Logische Verknüpfung	Schaltterm	Wahrheitstabelle	Schaltung aus Schaltsymbolen															
UND	$A \wedge B$	<table><tr><th>A</th><th>B</th><th>$A \wedge B$</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	$A \wedge B$	0	0	0	0	1	0	1	0	0	1	1	1	
A	B	$A \wedge B$																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
ODER	$A \vee B$	<table><tr><th>A</th><th>B</th><th>$A \vee B$</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	$A \vee B$	0	0	0	0	1	1	1	0	1	1	1	1	
A	B	$A \vee B$																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
NICHT	\bar{A}	<table><tr><th>A</th><th>\bar{A}</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	\bar{A}	0	1	1	0										
A	\bar{A}																	
0	1																	
1	0																	

Tabelle 3: Darstellungsformen der Grundsaltungen

Die Darstellung als Schaltterm und Wahrheitstabelle haben wir schon kennengelernt. In der Schaltung aus Schaltsymbolen sehen wir, dass stellvertretend für die Eingaben Schalter, die zwischen 0 und 1 wechseln können, verwendet werden. Für die Verknüpfung **NICHT** sind zwei Schaltsymbole gebräuchlich, daher sind hier beide Varianten abgebildet.

Aufgabe 3:

- Konstruiere die UND-, die ODER- sowie die NICHT-Schaltung aus Tabelle 3 mit einer Simulationssoftware für digitale Schaltungen.
- Simuliere nacheinander alle vier Eingabekombinationen. Beobachte dabei die Lampe und vergleiche mit der Wahrheitstabelle. Beschreibe die Bedeutung der Lampe.

Aufgabe 4: Erstelle für die Terme a) bis d) eine Wahrheitstabelle. Konstruiere anschließend die entsprechende Schaltung in einer Simulationssoftware und vergleiche die Simulation jeweils mit deiner Wahrheitstabelle.

- $\bar{A} \wedge B$
- $(A \wedge B) \vee (\bar{A} \wedge \bar{B})$
- $(A \vee B) \wedge C$
- $(A \wedge \bar{B}) \vee (\bar{A} \wedge B)$

Aufgabe 5:

- Konstruiere eine Schaltung mit zwei Schaltern für die Eingabe, einer Lampe für die Ausgabe und dem Bauteil aus Abbildung 1.
- Beschreibe das Verhalten der Schaltung in deinen eigenen Worten. Findest du einen booleschen Ausdruck, der die Schaltung beschreibt?
Wir kommen später noch einmal auf diese Schaltung zurück.

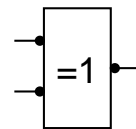


Abbildung 1: Bauteil zu Aufgabe 5

Es gibt Schaltterme, die bei einer identischen Belegung der Schaltvariablen jeweils die gleiche Ausgabe erzeugen, obwohl sie unterschiedliche logische Verknüpfungen verwenden. In einem solchen Fall spricht man von **äquivalenten** Termen.

Aufgabe 6: Welche der folgenden Terme sind äquivalent? Begründe mithilfe einer Wahrheitstabelle und / oder einer entsprechenden Schaltung.

- | | |
|-----------------------------|---------------------------|
| a) $\bar{A} \wedge \bar{B}$ | c) $\bar{A} \vee \bar{B}$ |
| b) $\overline{A \wedge B}$ | d) $\overline{A \vee B}$ |

Exkurs: Technische Realisierung logischer Verknüpfungen

Wir verwenden hier fertige Bauteile, um die logischen Verknüpfungen als Schaltung zu realisieren. Um eine Vorstellung davon zu bekommen, wie diese logischen Verknüpfungen technisch umgesetzt werden können, machen wir einen kurzen Ausflug in den Physikunterricht. Dort hast du sicher schon eine Reihen- und eine Parallelschaltung kennengelernt.

Aufgabe 7:

- Mache die Lampe in den Abbildungen 2 und 3 immer dann an, wenn Sie bei der eingezeichneten Schalterstellung leuchtet.
- Ordne der Parallelschaltung in Abbildung 2 und der in Reihenschaltung Abbildung 3 jeweils eine logische Verknüpfung zu, die durch diese Schaltung umgesetzt werden kann.

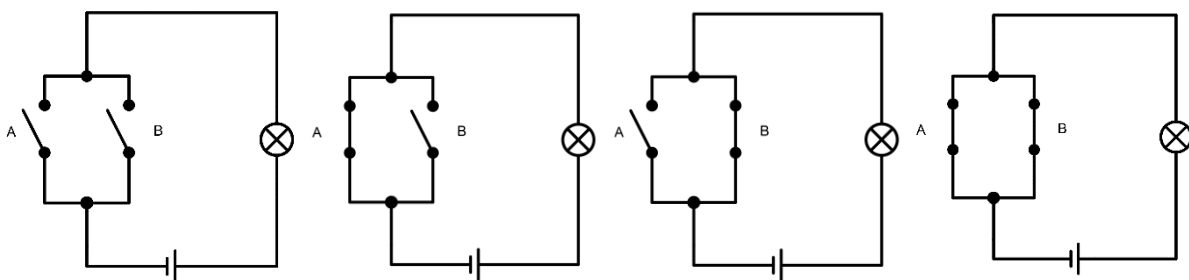


Abbildung 2: Parallelschaltung

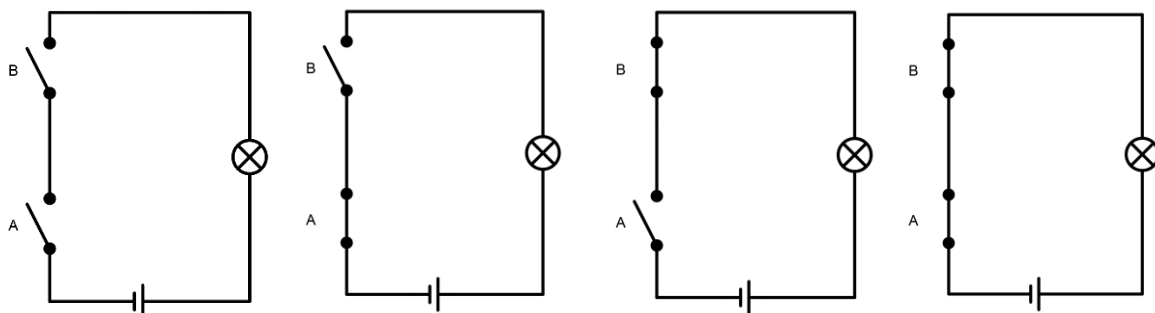


Abbildung 3: Reihenschaltung

Konstruktion von Schaltnetzen

Wenn wir Schaltungen betrachten, die in technischen Geräten zum Einsatz kommen, handelt es sich bei den Eingaben nicht nur um Schalter, sondern häufig um Sensorwerte. Gängige Ausgaben sind neben Lampen bzw. LEDs Aktionen, z. B. Geräusche oder das Starten eines Motors. Bei der Simulation der Schaltung können wir die Eingaben aber weiterhin stellvertretend als Schalter und die Ausgaben als Lampen symbolisieren.

Bei allen technischen Geräten, die wir hier betrachten, hängen die Ausgaben nur von den aktuellen Eingaben ab. Eine solche Schaltung bezeichnet man auch als **Schaltnetz**. Im Unterschied dazu gibt es auch Schaltwerke. Ein **Schaltwerk** ist eine Schaltung, die bei der Berechnung der Ausgaben nicht nur die aktuellen, sondern auch die vorherigen Eingaben berücksichtigt. Die technische Konstruktion von Schaltwerken würde an dieser Stelle aber zu weit führen.

Aufgabe 8: Eine Motorsäge hat neben dem Gashebel (G) noch einen Sicherheitsschalter, die Gashebelsperre (S). Damit die Kette (K) der Motorsäge läuft, müssen sowohl der Gashebel als auch die Gashebelsperre gedrückt werden. Dadurch soll ein ungewolltes Starten der Kette z. B. durch Äste verhindert werden.

Konstruiere eine entsprechende Schaltung und gib die Schaltfunktion dazu an.

Aufgabe 9: Für ein Haus mit einer Haus- und einer Terrassentür soll eine Alarmanlage konstruiert werden, die Alarm auslöst, wenn mindestens eine der beiden Türen geöffnet wird. Dazu benötigen wir für jede Tür einen Sensor, der erkennt, ob die Tür geöffnet wurde.

Hautür geöffnet?	Terrassentür geöffnet?	Alarm	H: Sensor für Haustür	T: Sensor für Terrassentür	A: Alarm
nein	nein		0	0	
nein	ja		0	1	
ja	nein		1	0	
ja	ja		1	1	

Tabelle 4: Links ist die Situation umgangssprachlich formuliert, rechts ist eine Codierung angegeben

- Ergänze in Tabelle 4 die Ausgaben.
- Entscheide, welche der beiden Schaltfunktionen für die Konstruktion der Alarmanlage geeignet ist. Konstruiere die Schaltung anschließend in einer Simulationssoftware.

(1) $A = H \wedge T$

(2) $A = H \vee T$
- Der Einbrecher könnte auch durch das Fenster kommen. Ergänze in der Schaltfunktion und in deiner Schaltung einen weiteren Sensor F für das Fenster, der eine 1 ausgibt, wenn das Fenster geöffnet wird.
- Die Bewohner des Hauses möchten nicht jedes Mal selbst den Alarm auslösen, wenn sie zu Hause sind und Türen oder Fenster öffnen. Ergänze eine Möglichkeit, die Alarmanlage zwischenzeitlich zu deaktivieren.

Aufgabe 10: Ein sensorgesteuertes Nachtlicht L verfügt über zwei Sensoren: einen Helligkeitssensor H und einen Entfernungssensor E . Der Helligkeitssensor gibt den Wert 1 zurück, wenn es hell ist. Der Entfernungssensor gibt den Wert 1 zurück, wenn in einer Entfernung weniger als 2m ein Objekt erkannt wird. Das Nachtlicht soll nur angehen, wenn es dunkel ist und sich jemand in der Nähe befindet.

Erstelle eine entsprechende Schaltfunktion und konstruiere die Schaltung in einer Simulationssoftware.

Aufgabe 11:

- Die elektrische Tür zu einem Robbengehege hat auf jeder Seite einen Schalter S1 und S2, um sie zu öffnen. Konstruiere eine entsprechende Schaltung für die Steuerung der Tür und gib den booleschen Term dazu an.
- Der Zugang zum Gehege der flinken Gibbons ist eine Schleuse, damit die kleinen Affen nicht aus Versehen entweichen können. Die Schleuse besteht aus zwei Türen. Dabei befindet sich im Rahmen der Türen jeweils ein Drucksensor (D1 bzw. D2), der registriert, ob die Tür geschlossen ist. Jede Tür hat außerdem auf jeder Seite einen Schalter (S1a und S1b bzw. S2a und S2b), um sie zu öffnen. Gesucht sind zwei Schaltnetze. Ein Schaltnetz für die Steuerung von Tür 1 und ein Schaltnetz für die Steuerung von Tür 2.

Ordne Tür 1 und Tür 2 die relevanten Eingaben zu. Eine Skizze kann dabei hilfreich sein.

Gib eine Schaltfunktion für Tür 1 an und konstruiere die Schaltung in einer Simulationssoftware. Gehe analog für Tür 2 vor.

Tipp: Du kannst die Aufgabe vereinfachen, indem du zunächst nur von einem Schalter pro Tür ausgehst.

Aufgabe 12: Überlege dir zu den folgenden Schaltermen passende Beispiele für technische Geräte oder Aussagen aus dem Alltag.

- $A \wedge \bar{B}$
- $A \wedge (B \vee C)$
- $A \vee B \vee C$

Aufgabe 13: Wenn wir einen Rechner konstruieren möchten, benötigen wir eine Möglichkeit, Zahlen zu addieren. In der einfachsten Form hieße das, zwei einstellige Binärzahlen zu addieren. Ein entsprechendes Schaltnetz wird als **Halbaddierer** bezeichnet. Tabelle 5 zeigt eine entsprechende Wahrheitstabelle. Die Schaltvariablen **A** und **B** stehen für die Eingaben, also die Zahlen, die addiert werden sollen. Für die Ausgabe benötigen wir zwei Ausgangssignale, da das Ergebnis auch zweistellig sein kann. Wir haben daher das Ausgangssignal **S** für die einstellige Summe und das Ausgangssignal **Ü** für den Übertrag.

- Ergänze in Tabelle 5 die Ausgaben Ü und S.
- Ordne die Schaltermen $(\bar{A} \wedge B) \vee (A \wedge \bar{B})$ sowie $A \wedge B$ begründet den Ausgangssignalen Ü und S zu.
- Vervollständige die Schaltung in Abbildung 4 zu einem Halbaddierer. Teste die Schaltung mit einer Simulationssoftware.

A	B	Ü	S
0	0		
0	1		
1	0		
1	1		

Tabelle 5: Wahrheitstabelle für einen Halbaddierer

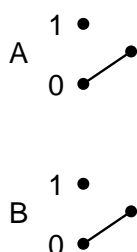


Abbildung 4: Grundgerüst für das Schaltnetz eines Halbaddierers

Weitere Logikbausteine

Wie oben bereits erwähnt, reichen die Logikbausteine UND, ODER sowie NICHT aus, um alle denkbaren Schaltungen zu konstruieren. Man bezeichnet sie daher auch als **Grundbausteine**. Wir können uns aber vorstellen, dass größere Schaltungen, die aus vielen Grundbausteinen bestehen, schnell unübersichtlich werden. Deshalb gibt es für Schaltungen, die sehr häufig auftreten und auch als Teil komplexerer Schaltungen verwendet werden, zusätzliche Schaltsymbole. Die entsprechenden Bausteine fassen die jeweilige Konstruktion aus Grundbausteinen zusammen.

Ein Beispiel wäre die Verknüpfung „entweder oder“, die wir beim Halbaddierer verwendet und in Aufgabe 5 bereits untersucht haben. Für den langen Term $(\bar{A} \wedge B) \vee (A \wedge \bar{B})$ gibt es die abkürzende Schreibweise $A \oplus B$. Das Schaltsymbol des Bausteins trägt die Beschriftung $=1$. Diese logische Verknüpfung wird als *XOR* bezeichnet.

Mit dem XOR-Baustein können wir die Schaltung für unseren Halbaddierer etwas übersichtlicher darstellen:

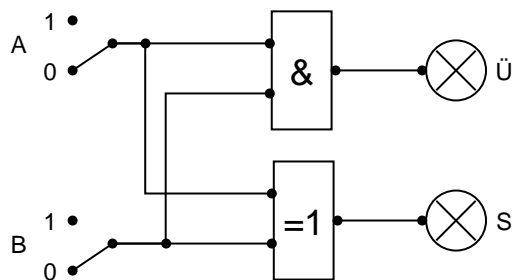


Abbildung 5: Halbaddierer mit UND- und XOR-Baustein

Auch für den Halbaddierer gibt es einen Baustein, der die gesamte Schaltung zusammenfasst. Das entsprechende Schaltsymbol ist mit HA beschriftet (s. Abbildung 6).

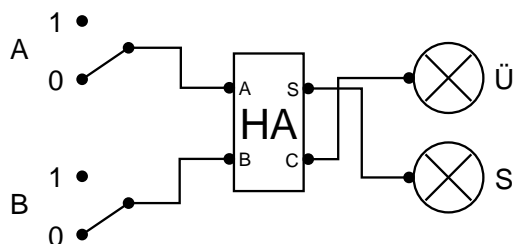


Abbildung 6: Halbaddierer-Baustein

Zu den eher grundlegenden Bausteinen gehören neben dem XOR noch die verneinten Versionen des UND-, des ODER- sowie des XOR-Bausteins. Sie werden als NAND, NOR bzw. XNOR bezeichnet.

Die Schaltung für den Term $\overline{A \wedge B}$ lässt sich z. B. aus einem UND-Baustein gefolgt von einem NICHT-Baustein realisieren (s. Abbildung 7).

Bei dem Schaltsymbol des zusammenfassenden NAND-Bausteins (s. Tabelle 6) wird das NICHT nur noch durch eine kleine Nase oder einen Kreis am Ausgang des UND-Bausteins symbolisiert. Analog werden auch die Schaltsymbole für den NOR- Baustein und den XNOR- Baustein gebildet. Die Reduzierung des NICHT auf eine Nase oder einen Kreis wird im

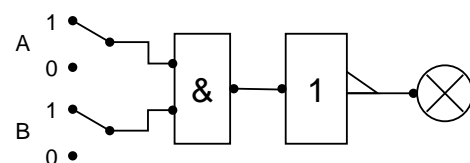


Abbildung 7: NAND-Schaltung

Schaltungsentwurf bei vielen Bausteinen verwendet. Diese vereinfachte Darstellung des NICHT kann nicht nur an den Ausgängen, sondern auch an den Eingängen verwendet werden, wenn das NICHT vorgeschaltet ist. Tabelle 6 zeigt eine Übersicht über die Bausteine NAND, NOR und XNOR.

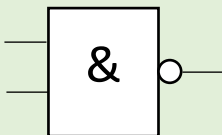
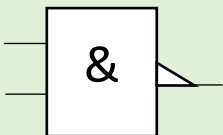
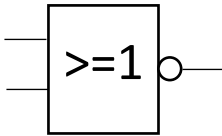
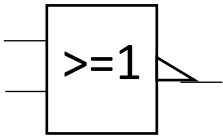
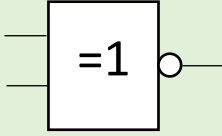
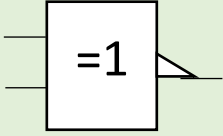
Name	Wahrheitstabelle	Schaltsymbol																
NAND	<table><tr><th>a</th><th>b</th><th>$\overline{A \wedge B}$</th></tr><tr><td>0</td><td>0</td><td></td></tr><tr><td>0</td><td>1</td><td></td></tr><tr><td>1</td><td>0</td><td></td></tr><tr><td>1</td><td>1</td><td></td></tr></table>	a	b	$\overline{A \wedge B}$	0	0		0	1		1	0		1	1			
	a	b	$\overline{A \wedge B}$															
	0	0																
	0	1																
	1	0																
1	1																	
NOR	<table><tr><th>a</th><th>b</th><th>$\overline{A \vee B}$</th></tr><tr><td>0</td><td>0</td><td></td></tr><tr><td>0</td><td>1</td><td></td></tr><tr><td>1</td><td>0</td><td></td></tr><tr><td>1</td><td>1</td><td></td></tr></table>	a	b	$\overline{A \vee B}$	0	0		0	1		1	0		1	1			
	a	b	$\overline{A \vee B}$															
	0	0																
	0	1																
	1	0																
1	1																	
XNOR	<table><tr><th>a</th><th>b</th><th>$\overline{A \oplus B}$</th></tr><tr><td>0</td><td>0</td><td></td></tr><tr><td>0</td><td>1</td><td></td></tr><tr><td>1</td><td>0</td><td></td></tr><tr><td>1</td><td>1</td><td></td></tr></table>	a	b	$\overline{A \oplus B}$	0	0		0	1		1	0		1	1			
	a	b	$\overline{A \oplus B}$															
	0	0																
	0	1																
	1	0																
1	1																	

Tabelle 6: Übersicht über die Bausteine NAND, NOR sowie XNOR

Aufgabe 14:

- Vervollständige die Wahrheitstabellen in Tabelle 6.
- Konstruiere eine NOR- sowie eine XNOR-Schaltung aus den Bausteinen ODER, XOR sowie NICHT.

Es sei noch angemerkt, dass ein UND gefolgt von einem NICHT logisch zwar gleichbedeutend mit einem NAND ist, dass sich dieser zusammengesetzte Baustein jedoch technisch leichter realisieren lässt als die zwei einzelnen Bausteine. Außerdem können alle logischen Ausdrücke so umgeformt werden, dass sie nur noch den NAND-Operator verwenden. Das macht man sich bei der praktischen Konstruktion von Schaltungen zunutze, da man so nur einen einzigen Baustein zur Verfügung stellen muss.

Zwischen verschiedenen Darstellungsformen eines Schaltnetzes wechseln

Bei den meisten Beispielen, die wir bislang betrachtet haben, hat sich der Term für die benötigte Schaltung direkt aus dem Zusammenhang ergeben. Bei den Schleusentüren musstest du vielleicht schon etwas länger überlegen, um den passenden Term für die Schaltung zu finden. Oft ist es einfacher zunächst eine Wahrheitstabelle zu erstellen, in der jede Eingabekombination einzeln betrachtet wird. Für komplexere Schaltnetze, bei denen der passende Term nicht durch bloßes *Hinschauen* erfasst werden kann, wäre es daher hilfreich ein Verfahren zu haben, mit dem sich der Schalterm aus einer Wahrheitstabelle systematisch ablesen lässt.

Außerdem wäre es denkbar, dass ein Schaltnetz bereits in Form einer Schaltung aus Schaltsymbolen existiert, jedoch weder die Wahrheitstabelle noch der Schaltterm dazu bekannt sind.

Im Folgenden werden wir uns daher anschauen, wie wir die Darstellungsformen Schaltterm, Wahrheitstabelle und Schaltung systematisch ineinander umwandeln können.

Von der Schaltung zum Term

Aufgabe 15: Konstruiere eine beliebige Schaltung aus den drei Grundbausteinen UND, ODER bzw. NICHT und versuche den passenden Schaltterm dafür zu finden.

Wenn wir den passenden Term zu einer Schaltung nicht sofort sehen, können wir systematisch vorgehen. Wir fangen beim letzten Schaltsymbol (ganz rechts) an und schreiben für jeden Eingang eine leere Klammer auf. Diese Klammer füllen wir dann mit den Schaltzeichen, die an den Eingängen anliegen, und schreiben für deren Eingänge wieder leere Klammern. So fahren wir fort, bis wir schließlich die letzten Klammern durch die Schaltvariablen ganz links ersetzen können. Tabelle 7 stellt die Übersetzungsregeln für die Grundsaltungen schematisch dar.

Verknüpfung	Schaltsymbol	wird übersetzt in
UND		$() \wedge ()$
ODER		$() \vee ()$
NICHT		$\overline{()}$

Tabelle 7: Übersetzung der Schaltsymbole in logische Operatoren

Wir führen das beschriebene Vorgehen einmal exemplarisch für das Schaltnetz in Abbildung 8 durch. Es ergeben sich die Schritte 1 bis 5.

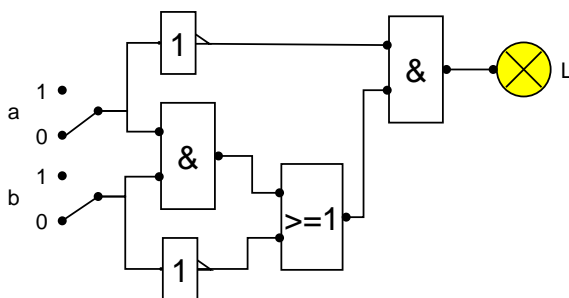


Abbildung 8: Beispiel für ein Schaltnetz, zu dem der Schaltterm gesucht ist.

Schritt 1: $L = () \wedge ()$

Schritt 2: $L = (\overline{()}) \wedge (() \vee ())$

Schritt 3: $L = (\overline{(a)}) \wedge ((() \wedge ()) \vee (\overline{()}))$

Schritt 4: $L = (\overline{(a)}) \wedge (((a) \wedge (b)) \vee (\overline{(b)}))$

Schritt 5 (überflüssige Klammern entfernen): $L = \bar{a} \wedge ((a \wedge b) \vee \bar{b})$

Aufgabe 16: Entwickle zu den Schaltungen in Abbildung 9 und Abbildung 10 schrittweise den passenden Schaltterm.

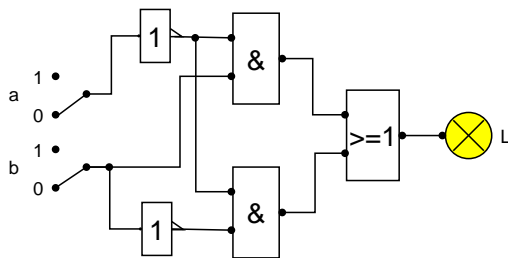


Abbildung 9: Schaltnetz 1 zu Aufgabe 16

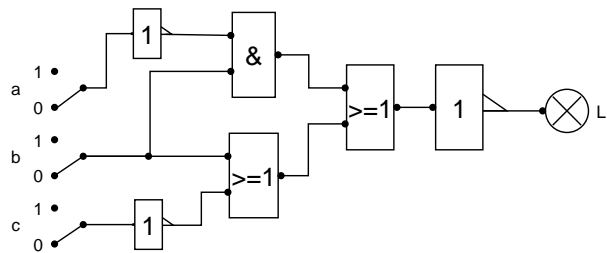
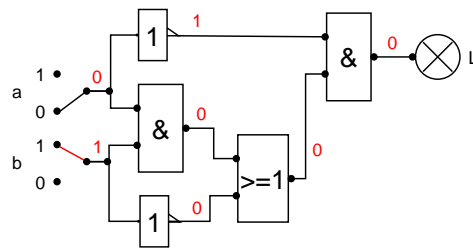
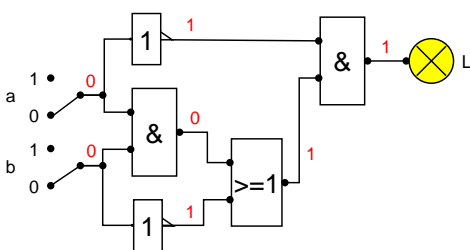


Abbildung 10: Schaltnetz 2 zu Aufgabe 16

Von der Schaltung zur Wahrheitstabelle

Um für eine gegebene Schaltung direkt die Wahrheitstabelle zu erstellen, müssen wir alle Eingabekombinationen durchspielen. Das ist ganz einfach, wenn uns die Schaltung in einer Simulationssoftware zur Verfügung steht. Dann müssen wir die Eingabekombinationen nur systematisch simulieren. Steht uns kein entsprechendes Programm zur Verfügung, müssen wir die einzelnen Kombinationen in Gedanken durchspielen. Dazu ist es hilfreich, für jede Leitung zu notieren, ob sie bei der aktuellen Eingabe eine 0 oder eine 1 weiterleitet. So arbeiten wir uns von den Eingaben (Schalter) bis zur Ausgabe (Lampe) vor.

Betrachten wir als Beispiel die Schaltung in Abbildung 11. Links ist die Belegung der Leitungen für die Schalterstellung $a = 0$ und $b = 0$ notiert. Die letzte Leitung vom UND-Baustein zur Lampe ist mit 1 belegt, so dass die Lampe leuchtet und eine 1 in die entsprechende Zeile der Wahrheitstabelle eingetragen wird. Rechts ist die Belegung der Leitungen für die Schalterstellung $a = 0$ und $b = 1$ notiert. Hier liegt an der Leitung zur Lampe eine 0 ein, so dass in der entsprechenden Zeile der Wahrheitstabelle eine Null eingetragen wird.



a	b	?
0	0	1
0	1	0
1	0	
1	1	

Abbildung 11: Systematisches Testen verschiedener Eingaben

Tabelle 8:
Wahrheitstabelle
zu Abbildung 11

Aufgabe 17: Vervollständige die Wahrheitstabelle (Tabelle 8) zu der Schaltung in Abbildung 11.

Aufgabe 18: Erstelle zu der Schaltung in Abbildung 12 zuerst die Wahrheitstabelle und dann den Schaltterm.

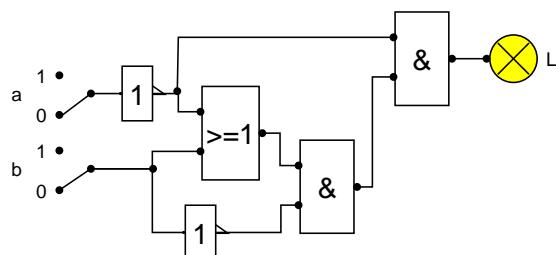


Abbildung 12: Schaltung zu Aufgabe 18

Aufgabe 19: Partnerarbeit

- Konstruiere eine Schaltung aus 5 bis 7 Schaltsymbolen. Verwende dabei nur die Grundbausteine UND, ODER bzw. NICHT. Erstelle systematisch den Schaltterm dazu. Stelle für den Term die Wahrheitstabelle auf.
- Tauscht eure Schaltung im Team gegenseitig aus. Erstelle zuerst die Wahrheitstabelle und dann den Schaltterm zu der Schaltung, die du erhalten hast.
- Vergleicht zu jeder Schaltung eure Ergebnisse. Verwendet ggf. ein Simulationsprogramm, um eure Wahrheitstabellen zu überprüfen.

Von der Wahrheitstabelle zum Schaltterm: Systematische Konstruktion von Schaltungen

Wenn Schaltungen konstruiert werden sollen, kennt man häufig zunächst nur die Wahrheitstabelle. Denn man weiß, welche Ausgaben man für welche Eingaben erwartet. Betrachten wir als Beispiel eine Pflanzenbewässerungsanlage, die mithilfe von drei Sensoren gesteuert wird: einem **Feuchtigkeitssensor F**, einem **Temperatursensor T** und einem **Helligkeitssensor H**. Der Feuchtigkeitssensor gibt den Wert 1 aus, wenn die Erde feucht genug ist, der Temperatursensor gibt den Wert 1 aus, wenn es wärmer als 25°C ist und der Helligkeitssensor gibt den Wert 1 aus, wenn es taghell ist. Die Pflanzen sollen nur bewässert werden, wenn die Erde zu trocken ist. Um die Pflanzen nicht zu schädigen, darf die Bewässerung aber erst beginnen, wenn es Nacht (also dunkel) ist oder wenn die Temperaturen unter 25°C gesunken sind.

Tabelle 9 zeigt die Wahrheitstabelle, die sich aus der Beschreibung der Bewässerungsanlage ergibt.

F	T	H	B	Erläuterung
0	0	0	1	zu trocken und sowohl kalt genug als auch dunkel → Bewässerung starten
0	0	1	1	zu trocken und kalt genug → Bewässerung starten
0	1	0	1	zu trocken und dunkel → Bewässerung starten
0	1	1	0	zu trocken, aber warm und hell → keine Bewässerung
1	0	0	0	feucht genug → keine Bewässerung
1	0	1	0	feucht genug → keine Bewässerung
1	1	0	0	feucht genug → keine Bewässerung
1	1	1	0	feucht genug → keine Bewässerung

Tabelle 9: Wahrheitstabelle der Bewässerungsanlage

Vielleicht hast du schon eine Idee für eine passende Schaltfunktion. Hier muss man allerdings schon etwas länger überlegen. Deshalb schauen wir uns jetzt einen Weg an, wie wir den passenden Term systematisch herleiten können. Dazu zerlegen wir die Spalte für die Ausgabe B zunächst in mehrere Hilfsspalten. Für jede Zeile mit der Ausgabe 1 legen wir eine **Hilfsspalte** mit einer 1 in dieser Zeile und ansonsten nur 0en an (s. Tabelle 10).

Für die einzelnen Hilfsspalten einen Schaltterm zu finden, ist nun deutlich einfacher, da wir eine logische Verknüpfung kennen, die für genau eine Eingabekombination den Wert 1

F	T	H	B1	B2	B3
0	0	0	1	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	0	0	0
1	1	1	0	0	0

Tabelle 10: Ausgabe unterteilt in Hilfsspalten

annimmt und sonst den Wert 0: Die **UND-Verknüpfung**. Damit die Ausgabe einer UND-Verknüpfung den Wert 1 annimmt, müssen alle Eingaben den Wert 1 haben. Wenn wir uns die erste Hilfsspalte anschauen, so steht die 1 jedoch in der Zeile, in der die Eingaben 0 sind. Um jeweils eine 1 als Eingabe zu erhalten, müssen die Eingaben vorher verneint werden. Für die erste Hilfsspalte ergibt sich dann die Schaltfunktion $B1 = \bar{F} \wedge \bar{T} \wedge \bar{H}$.

In der zweiten Hilfsspalte müssen nur die Eingaben F und T verneint werden, da die Eingabe H in der Zeile mit der Ausgabe 1 bereits mit 1 belegt ist: $B2 = \bar{F} \wedge \bar{T} \wedge H$

Für die dritte Hilfsspalte werden F und H verneint: $B3 = \bar{F} \wedge T \wedge \bar{H}$

F	T	H	B1	B2	B3	B
			$\bar{F} \wedge \bar{T} \wedge \bar{H}$	$\bar{F} \wedge \bar{T} \wedge H$	$\bar{F} \wedge T \wedge \bar{H}$	
0	0	0	1	0	0	1
0	0	1	0	1	0	1
0	1	0	0	0	1	1
0	1	1	0	0	0	0
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	0	0	0	0
1	1	1	0	0	0	0

Tabelle 11: Wahrheitstabelle der Bewässerungsanlage mit Hilfsspalten und zugehörigen Termen

Damit die Bewässerung startet, reicht es aus, wenn eine der Hilfsspalten den Wert 1 annimmt. Auch eine solche logische Verknüpfung, die eine 1 ausgibt, wenn mindestens eine Eingabe den Wert 1 hat, kennen wir: die **ODER-Verknüpfung**.

Den vollständigen Term für die Ausgabe B erhalten wir daher, wenn wir die Teilterme für die Hilfsspalten mit ODER verknüpfen: $B = B1 \vee B2 \vee B3 = (\bar{F} \wedge \bar{T} \wedge \bar{H}) \vee (\bar{F} \wedge \bar{T} \wedge H) \vee (\bar{F} \wedge T \wedge \bar{H})$

Wenn die Funktion für eine Schaltung auf diesem Weg systematisch erzeugt wird, erhält man den Schaltterm in der **disjunktiven Normalform (DNF)**. Die Terme für die Hilfsspalten bezeichnet man als **Minterme**.

Wir fassen das Verfahren zum Erzeugen der disjunktiven Normalform aus einer Wahrheitstabelle noch einmal zusammen:

1. Erstelle für jede Zeile, in der die Ausgabe 1 ist, eine Hilfsspalte mit einer 1 in dieser Zeile und sonst 0en.
2. Stelle für jede Hilfsspalte den Minterm auf. Dazu werden die Schaltvariablen mit UND verknüpft. Schaltvariablen, die in der Zeile, in der die Ausgabe 1 ist, mit 0 belegt sind, werden negiert.
3. Verknüpfe die einzelnen Minterme mit ODER.

Anmerkung zu Schritt 1 und 2: Anstatt die Hilfsspalten jedes Mal explizit anzulegen, reicht es aus, die Ausgabe in Gedanken zu zerlegen und jede Zeile mit der Ausgabe 1 einzeln zu betrachten.

Aufgabe 20: Leite für die Ausgaben K und L in der Wahrheitstabelle (Tabelle 12) den Schaltterm in der disjunktiven Normalform her.

Aufgabe 21: Die Wahrheitstabelle in Tabelle 13 wurde für eine Schaltung erstellt, die eine Prüfziffer erzeugen soll. Die Schaltung soll als Prüfziffer eine 1 ausgeben, wenn die Anzahl der Einsen an den Eingängen gerade ist.

- Vervollständige die Wahrheitstabelle entsprechend.
- Leite den Schaltterm in der disjunktiven Normalform her.
- Konstruiere die zum Term passende Schaltung in einem Simulationsprogramm. Vergleiche die Simulation der Schaltung mit der Wahrheitstabelle.

a	b	K	L
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

Tabelle 12:
Wahrheitstabelle zu
Aufgabe 20

Vereinfachung von Schalttermen

Die Terme, die wir als disjunktive Normalform erhalten, sind oft sehr lang. In Aufgabe 22 soll daher eine Regel erarbeitet werden, mit der sich Terme in der disjunktiven Normalform systematisch vereinfachen lassen.

Aufgabe 22:

- Begründe, dass die folgenden Terme äquivalent sind.
 - $(A \wedge B) \vee (A \wedge \bar{B}) = A$
 - $(A \wedge B \wedge C) \vee (A \wedge B \wedge \bar{C}) = A \wedge B$
 - $(A \wedge B \wedge \bar{C}) \vee (A \wedge \bar{B} \wedge \bar{C}) = A \wedge \bar{C}$
- Formuliere mithilfe deiner Überlegungen aus Aufgabenteil a) eine Vereinfachungsregel für Terme in der disjunktiven Normalform.
- Vereinfache die folgenden Terme soweit wie möglich:
 - $(\bar{A} \wedge B \wedge C) \vee (\bar{A} \wedge B \wedge \bar{C})$
 - $(A \wedge B \wedge C) \vee (\bar{A} \wedge \bar{B} \wedge C) \vee (\bar{A} \wedge \bar{B} \wedge \bar{C})$
 - $(A \wedge B) \vee (A \wedge \bar{B}) \vee (\bar{A} \wedge B)$
 - $(\bar{A} \wedge B \wedge C) \vee (A \wedge \bar{B} \wedge C) \vee (A \wedge B \wedge C) \vee (A \wedge \bar{B} \wedge \bar{C})$
 - $(A \wedge B \wedge C) \vee (A \wedge B \wedge \bar{C}) \vee (A \wedge \bar{B} \wedge C)$

a	b	c	?
0	0	0	1
0	0	1	0
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Tabelle 13: Wahrheits-
tabelle zu Aufgabe 21

Aufgabe 23: Verwende die Vereinfachungsregel, die du in Aufgabe 22 erarbeitet hast, um den Schaltterm für die Bewässerungsanlage $(\bar{F} \wedge \bar{T} \wedge \bar{H}) \vee (\bar{F} \wedge \bar{T} \wedge H) \vee (\bar{F} \wedge T \wedge \bar{H})$ zu vereinfachen.

Alternativer Zugang

Mit der disjunktiven Normalform und der Vereinfachungsregel aus Aufgabe 22 hast du einen zuverlässigen Weg kennengelernt, wie du systematisch von der Wahrheitstabelle zu einer überschaubaren Schaltfunktion kommst. Wenn du ein wenig Übung im Erstellen von Schalttermen hast, siehst du aber vielleicht schon anhand der Wahrheitstabelle Zeilen, in denen die Ausgabe 1 ist und die sich gut zusammenfassen lassen, weil eine Schaltvariable z. B. immer den gleichen Wert hat. Wenn wir noch einmal Tabelle 11, die Wahrheitstabelle für die Bewässerungsanlage, betrachten, führen die ersten drei Belegungen der Schaltvariablen zur Ausgabe 1. In allen drei Fällen ist $F = 0$ und es muss mindestens eine der Schaltvariablen T oder H mit Null belegt sein. Wenn wir diese Überlegungen in einem booleschen Term ausdrücken, erhalten wir direkt: $B = \bar{F} \wedge (\bar{T} \vee \bar{H})$.

Da für logische Verknüpfungen ein Distributivgesetz³ gilt, wie du es für die Addition und die Multiplikation aus der Mathematik kennst, lässt sich der Schaltterm umformen zu $(\bar{F} \wedge \bar{T}) \vee (\bar{F} \wedge \bar{H})$. Das ist der Schaltterm, den du nach Anwenden der Vereinfachungsregel in Aufgabe 23 erhalten hast. Bei der Suche nach einer passenden Schaltfunktion zu einer Wahrheitstabelle, solltest du dir die Tabelle also zunächst genau anschauen. Vielleicht kannst du hier bereits logische Zusammenhänge erkennen. Wenn nicht, steht dir mit der disjunktiven Normalform ein zuverlässiger Weg zum Erstellen der Schaltfunktion zur Verfügung.

Aufgabe 24: Betrachten wir die Fußgängerampel in Abbildung 13. Auf beiden Seiten der Ampel registrieren Sensoren den Verkehr. **Sensor AR** zeigt an, ob aktuell Autos von rechts kommen und **Sensor AL** zeigt an, ob aktuell Autos von links kommen. Außerdem verfügt die Ampel über einen **Timer T**, der das Signal 1 sendet, wenn seit der letzten Grünphase der Fußgängerampel mindestens drei Minuten vergangen sind. Wenn ein Fußgänger auf den **Knopf K** der Fußgängerampel drückt, springt sie auf grün, wenn nur aus einer der beiden Richtungen ein Auto kommt oder wenn seit der letzten Grünphase der Ampel mindestens drei Minuten vergangen sind.

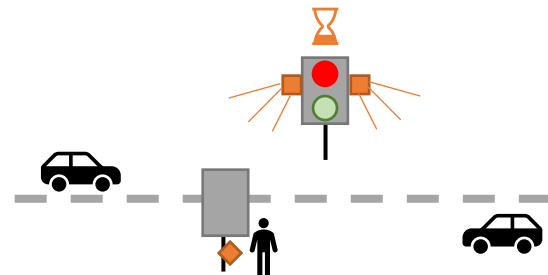


Abbildung 13: Fußgängerampel mit Sensoren

- Stelle die Funktionsweise der Ampel in einer Wahrheitstabelle dar. In Tabelle 14 wurde bereits ein Anfang gemacht.
- Leite den Term für die Schaltung her. Vereinfache den Term soweit wie möglich.
- Konstruiere zu dem vereinfachten Term die passende Schaltung in einem Simulationsprogramm. Vergleiche die Simulation mit deiner Wahrheitstabelle.

K (Knopf)	AR (Auto von rechts)	AL (Auto von links)	T (Timer)	G (Fußgängerampel schaltet auf grün)	Bemerkung
0	0	0	0	0	Knopf nicht gedrückt → Ampel bleibt rot
0	0	0	1	0	
0	0	1	0		

Tabelle 14: Wahrheitstabelle zur Fußgängerampel in Aufgabe 24

Aufgabe 25: Partnerarbeit

Ihr habt nun gelernt, die Darstellung eines Schaltnetzes in eine andere zu transformieren. Zeichnet in Abbildung 14 abwechselnd einen Pfeil ein, wenn ihr die entsprechende Transformationsrichtung beherrscht. Erläutere deinem Partner/deiner Partnerin zu jedem Pfeil, den du zeichnest, wie du bei der Transformation vorgehst.

Wahrheitstabelle

Schaltterm

Schaltung

Abbildung 14: Transformation der Darstellungen eines Schaltnetzes

³ Distributivgesetze: (1) $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$ (2) $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$

Konstruktion komplexerer Schaltungen

Bislang haben wir die Schalter an den Eingängen unserer Schaltung einzeln betrachtet. Jeder Schalter stand für eine Eingabe, die nur zwei Zustände annehmen kann: 0 oder 1. Wenn wir nun komplexere Eingaben verarbeiten wollen, z.B. die Zahlen von 0 bis 20 oder 26 verschiedene Buchstaben benötigen wir mehr als zwei Zustände. Dazu müssen wir mehrere Schalter zusammenfassen. Drei Schalter a, b und c ergeben eine Kette aus drei Nullen und Einsen. Daraus ergeben sich acht verschiedene Kombinationen, so dass wir mit drei Schaltern acht verschiedene Zustände unterscheiden können. Jeder Zustand kann zum Beispiel für eine Zahl von 0 bis 7 stehen. Die Zuordnung zwischen den Schalterkombinationen und den Zahlen erfolgt dabei mithilfe des Binärsystems. Wenn z. B. Schalter a und c auf 1 stehen und Schalter b auf 0, ergibt das die Zahl $(101)_2$, also eine 5 im Zehnersystem⁴.

Da alle Daten, wie Texte und Bilder mithilfe von Zahlencodes gespeichert werden, können wir mithilfe der Binärzahlen, alle Informationen darstellen, die wir verarbeiten wollen.

Aufgabe 26: Gesucht ist eine Schaltung, die anzeigt, ob es sich bei der Zahl an den Eingängen, um eine Primzahl handelt.

- Vervollständige zunächst die Wahrheitstabelle. Dabei stellt jede Zeile eine Binärzahl $a_2a_1a_0$ dar. 001 in der zweiten Zeile entspricht der Eins im Zehnersystem. Eins ist keine Primzahl, also muss die Schaltung eine Null ausgeben.
- Stelle den Schaltterm in der disjunktiven Normalform auf. Vereinfache den Term, wenn möglich.
- Konstruiere die Schaltung zu deinem vereinfachten Term mit einer Simulationssoftware.
- Erweitere deine Schaltung um eine zusätzliche Ausgabe, die genau dann 1 ist, wenn die Zahl durch zwei teilbar ist.

a_2	a_1	a_0	Primzahl
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Tabelle 15: Wahrheitstabelle zu Aufgabe 26

Schaltnetz eines Addierers

Die wichtigste Aufgabe eines Computers ist das Rechnen. Wir wollen deshalb eine Schaltung konstruieren, die zwei Binärzahlen addieren kann. Dazu müssen wir uns zunächst überlegen, wie Binärzahlen addiert werden. Erinnerst du dich an die schriftliche Addition, die du in der Grundschule gelernt hast? Auf diese Weise lassen sich auch Binärzahlen addieren. Hier ein Beispiel für zwei Dezimalzahlen und die entsprechende Rechnung im Binärsystem:

$$\begin{array}{r}
 1 \quad 4 \\
 + \quad 2_1 \quad 9 \\
 \hline
 4 \quad 3
 \end{array}
 \qquad
 \begin{array}{r}
 0 \quad 1 \quad 1 \quad 1 \quad 0 \\
 + \quad 1_1 \quad 0_1 \quad 0_1 \quad 1 \quad 1 \\
 \hline
 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1
 \end{array}$$

Aufgabe 27: Löse die folgenden Additionsaufgaben durch schriftliche Addition im Binärsystem.

a) $111 + 10$

b) $1001 + 1101$

c) $10111 + 100110$

⁴ Wenn du mit dem Binärsystem noch nicht vertraut bist, findest du in der Datei Exkurs_Binaersystem eine Erläuterung.

Eine Schaltung zur Addition zweier vierstelliger Binärzahlen hat acht Eingänge. Eine Wahrheitstabelle, die alle acht Eingaben berücksichtigt, wäre jedoch sehr unübersichtlich. Wenn wir uns die schriftliche Addition genauer anschauen, haben wir aber auch dort die Summanden nicht als Ganzes betrachtet, sondern Ziffer für Ziffer. Wir werden daher auch unsere Schaltung für die Addition schrittweise konstruieren, so dass immer nur die beiden gleichwertigen Stellen addiert werden müssen.

Aufgabe 28: In Aufgabe 13 haben wir eine Schaltung für einen Halbaddierer konstruiert, der zwei einstellige Binärzahlen addiert. Erläutere, weshalb dieser Baustein nicht ausreicht, um beispielsweise die Binärziffern an der dritten Stelle zu addieren.

Aufgabe 29: Einen Baustein, der drei einstellige Binärzahlen addieren kann, nennt man **Volladdierer**.

- Vervollständige die Wahrheitstabelle in Tabelle 16 für die Schaltung eines Volladdierers.
- Erstelle für die beiden Ausgaben, Summe S und Übertrag Ü die Schaltfunktion in der disjunktiven Normalform.
- Konstruiere die Schaltung in einer Simulationssoftware.

a	b	c	Ü	S
0	0	0	0	0
0	0	1	0	1
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Tabelle 16: Wahrheitstabelle zu Aufgabe 29

Aufgabe 30: Da man Halbaddierer und Volladdierer sehr häufig benötigt, hat man die Schaltungen jeweils in einem eigenen Baustein zusammengefasst.

- Suche in deinem Simulationsprogramm nach den entsprechenden Bausteinen. Sie sind mit HA und VA beschriftet. Die Ausgänge sind häufig mit S und C beschriftet. S steht für „Sum“, also Summe, und C für „Carry-Bit“, das ist der Übertrag.
- Teste ob die Bausteine sich genauso wie deine selbst konstruierten Schaltungen für den Halb- bzw. den Volladdierer verhalten!

Aufgabe 31: Ein Volladdierer lässt sich auch aus zwei Halbaddierern und einem ODER-Baustein konstruieren. Begründe, dass sich das Schaltnetz in Abbildung 15 genauso verhält, wie wir es von einem Volladdierer erwarten.

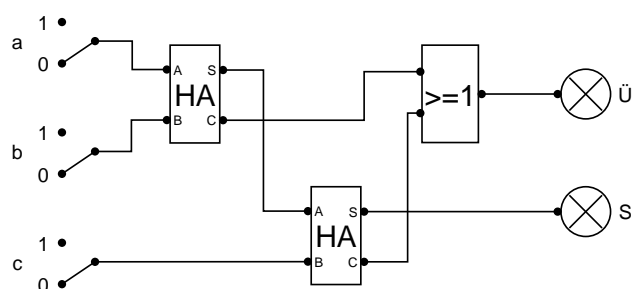


Abbildung 15: Konstruktion eines Volladdierers aus Halbaddierern.

Aufgabe 32: Aus einem Halbaddierer und drei Volladdierern soll eine Schaltung für die Addition zweier vierstelliger Binärzahlen $a_3a_2a_1a_0$ und $b_3b_2b_1b_0$ konstruiert werden.

- Verdrahte die Bausteine in Abbildung 16 entsprechend.
- Konstruiere die Schaltung in einem Simulationsprogramm und teste sie.
- Erweitere die Schaltung so, dass zwei fünfstellige Binärzahlen addiert werden können.

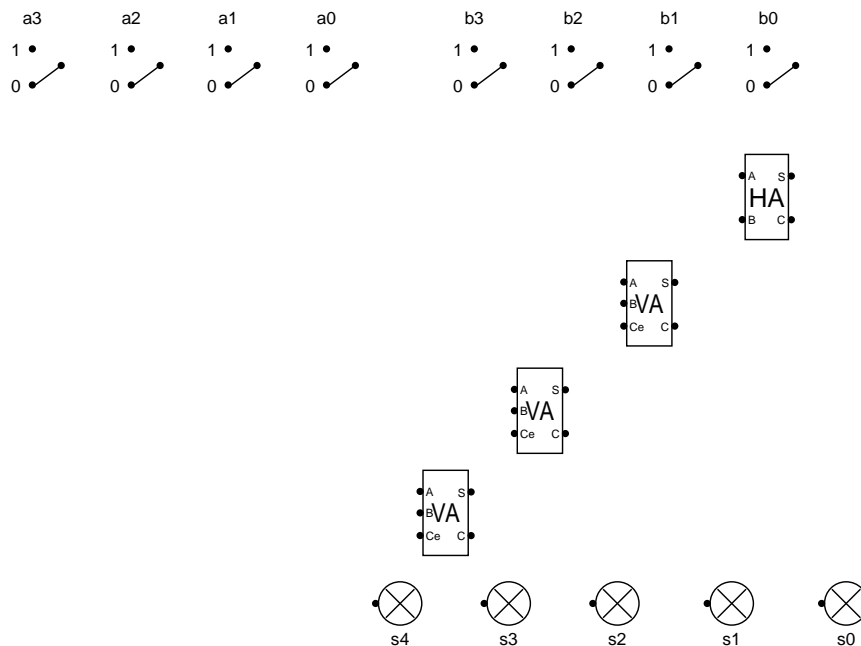


Abbildung 16: Vorlage für die Konstruktion einer Additionsschaltung für zwei vierstellige Binärzahlen.

Dieses Werk ist lizenziert unter einer [Creative Commons Namensnennung - Nicht-kommerziell - Weitergabe unter gleichen Bedingungen 4.0 International Lizenz](#). Sie erlaubt Bearbeitungen und Weiterverteilung des Werks unter Nennung meines Namens und unter gleichen Bedingungen, jedoch keinerlei kommerzielle Nutzung.

Bildnachweis: Die Abbildungen wurden unter Verwendung der Zeichenfunktion und Piktogrammen von MS Word 2016 sowie des Editors yed (<http://www.yWorks.com>) selbst erstellt.