

Digitale Steuerungstechnik

**Grundfunktionen
Boolsche Algebra
Zeitfunktionen
Zählfunktionen
Schaltalgebra
Karnaugh Veitch Diagramm
Speicherprogrammierbare Steuerungen**



Mag. Georg N. Strauss

November 2003

Inhaltsverzeichnis

1) Grundfunktionen	Seite
1.1. Binäre Verknüpfungen	Seite
1.2. Boolesche Algebra	Seite
1.3. SR/RS-Speicherfunktionen	Seite
1.4. Konnektor und Zuweisung	Seite
1.5. Flankenauswertung	Seite
2) Schaltalgebra – Karnaugh-Diagramm	Seite
2.1. Kombinatorische Steuerung	Seite
2.2. KV Diagramme	Seite
2.3. SR Speicherglieder	Seite
2.4. Konnektor und Zuweisung	Seite
2.5. Flankenauswertung	Seite
3) Zeitfunktionen	Seite
4) Zählfunktionen	Seite
5) SPS - Aufbau und Funktionsweise	Seite
5.1. Funktionsblöcke der SPS	Seite
5.2. Arbeitsweise einer SPS	Seite
5.3. Mechanischer Aufbau der SPS	Seite
5.4. Komponenten der SPS-Technik	Seite
5.5. Strukturierte Programmierung	Seite
5.6. Programmdarstellung	Seite

1. Grundfunktionen

1.1. Binäre Verknüpfungen

1.1.1. UND, AND, Konjunktion:

Die Verknüpfungen binärer Signalzustände können durch die UND (AND), ODER (OR), und NICHT (NOT) Funktionen verwirklicht werden. Häufig nicht erwähnt wird die IDENTITÄT (JA-Funktion). Die Identität liefert den Signalzustand 1, wenn der Signalzustand des Eingangs ebenfalls 1 ist.

Die UND-Funktion verknüpft zwei binäre Signalzustände miteinander und liefert ein Verknüpfungsergebnis 1, wenn beide Abfrageergebnisse gleichzeitig 1 sind. Wenn die UND-Funktion mehr als zwei Eingänge hat, müssen die Abfrageergebnisse aller Eingänge 1 sein, damit das gemeinsame Verknüpfungsergebnis 1 ist.

Funktionstabelle der UND-Funktion

Eingang 1	Eingang 2	Ausgang
0	0	0
0	1	0
1	0	0
1	1	1

1.1.2. ODER, OR, Disjunktion:

Die ODER-Funktion verknüpft binäre Zustände miteinander und liefert ein Verknüpfungsergebnis 1, wenn mindestens einer der Zustände (Abfrageergebnis) 1 ist.

Funktionstabelle der ODER-Funktion

Eingang 1	Eingang 2	Ausgang
0	0	0
0	1	1
1	0	1
1	1	1

Tipps

Wenn bei der Programmerstellung der Signalzustand eines Sensors abgefragt werden soll, so ist zu berücksichtigen, ob der Sensor ein Schließer oder ein Öffner ist. Die Steuerung hat keine Möglichkeit festzustellen, ob ein Eingang mit einem Schließer oder einem Öffner belegt ist. Ein Schließer liefert bei Betätigung den Signalzustand 1; er wird direkt abgefragt, wenn die Betätigung des Sensors das Abfrageergebnis 1 liefern soll. Ein Öffnerkontakt liefert bei Betätigung den Signalzustand 0; wenn bei dessen Betätigung das Abfrageergebnis 1 sein soll, muss er negiert (NICHT-Funktion) abgefragt werden. Auf diese Weise können auch Eingänge, die bei Signalzustand 0 Aktivitäten ausführen sollen („nullaktiv“ sind), abgefragt werden.

1.1.3. NICHT, NOT, Inverter:

Funktionstabelle der NICHT-Funktion

Eingang	Ausgang
0	1
1	0

Tipps:

Bei der Kombination verschiedener binärer Verknüpfungen ist auf die Rechenregeln zu achten. Es gilt UND vor ODER, ähnlich der Regel PUNKT vor STRICH in der Algebra. Bei einer ODER-vor-UND-Verknüpfung müssen daher Klammern gesetzt werden.

1.2. BOOLSCHES ALGEBRA:

- George Boole, britischer Mathematiker, 1815-1864 "The mathematical analysis of logic" (Algebra zur systematischen Behandlung von Logik) 1847, 1854
- 1904 E.V. Huntington führt eine axiomatische Ableitung dieser Algebra ein
- 1938 leitet Claude Elwood Shannon (1916-2001, USA, Bell Labs) eine 2-wertige Boolesche Algebra zur Beschreibung binärer Schaltung ab (Schaltalgebra, Switching Theory)

Sinn: Analyse, Vereinfachung und Synthese von digitalen Schaltungen nach definierten Gesetzen

Tabelle: Umsetzung des logischen Zustandes in ein technisches Signal

Technologie	0	1
Relais (Kontakt)	offen	geschlossen
Glasfaser	Licht aus	Licht an
CD	Keine Vertiefung	Vertiefung
CMOS	0 – 1,5V	3,5 – 5V
TTL	0 - 0,8V	2 – 5V

Low: Signal im niedrigeren Spannungsbereich

High: Signal im höheren Spannungsbereich

Tipps:

Zuordnung von Low und High zu einem logischen / booleschen / digitalen Wert:
 Positive Logik: Low = 0 High = 1 (intuitive, natürliche Festlegung)
 Negative Logik : Low = 1 High = 0 (weniger gebräuchlich)

Darstellung boolescher Funktionen:

- ?? Wertetabelle (Wahrheitstabelle)
- ?? Schaltungstechnisches Symbol
- ?? Algebraische Gleichung
- ?? Kontaktplan

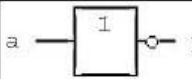
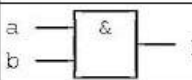
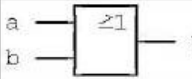
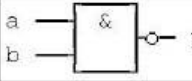
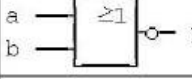
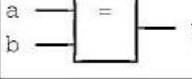
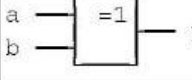
Zeichen	Benennung	Definitionen				Schreib- und Sprechweise	Symbol
		a=	0	1	0		
— ¬	Negation	a=1	y = 0				$y = \bar{a} = \text{nicht } a$ 
		a=0	y = 1				
∧	Konjunktion, UND/AND	y =	0	0	0	1	$y = a \wedge b;$ a und b 
∨	Disjunktion, ODER/OR	y =	0	1	1	1	$y = a \vee b;$ a oder b 
— ∧	NAND	y =	1	1	1	0	$y = \overline{a \wedge b};$ a nand b 
— ∨	NOR	y =	1	0	0	0	$y = \overline{a \vee b};$ a nor b 
≡	Äquivalenz, (E)XNOR	y =	1	0	0	1	$y = a \equiv b;$ a äquiv. b a EXNOR b 
≠ ⊗	Antivalenz (E)XOR	y =	0	1	1	0	$y = a \otimes b;$ a antiv. b a EXOR b 

Abb.1.1: Symbole der Schaltalgebra und logische Verknüpfungen nach DIN

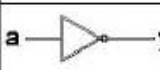




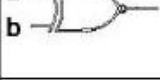
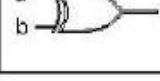
Zeichen	Benennung	Definitionen				Schreib- und Sprechweise	Symbol
		a=	0	1	0		
—	Negation NOT	a=1	y = 0				$y = \bar{a};$ not a 
		a=0	y = 1				
·	Konjunktion, UND/AND	y =	0	0	0	1	$y = a \cdot b = a b;$ a and b 
+	Disjunktion, ODER/OR	y =	0	1	1	1	$y = a + b;$ a or b 
— ·	NAND	y =	1	1	1	0	$y = \overline{a \cdot b};$ a nand b 
— +	NOR	y =	1	0	0	0	$y = \overline{a + b};$ a nor b 
— ⊗	Äquivalenz, (E)XNOR	y =	1	0	0	1	$y = a \otimes b;$ a EXNOR b 
⊗	Antivalenz (E)XOR	y =	0	1	1	0	$y = a \otimes b;$ a EXOR b 

Abb.1.2: Symbole der Schaltalgebra und logische Verknüpfungen nach amerikanischer Darstellung

Tipps:

In der amerikanischen Schreibweise - aber auch in der Mathematik – wird sehr oft der UND Operator mit \cdot und der ODER Operator mit $+$ bezeichnet.

Spezielle Verknüpfungen:

$0 \cdot 0 = 0$	$0 \cdot E1 = E1$
$1 \cdot 1 = 1$	$1 \cdot E1 = 1$
$0 \cdot 0 = 0$	$0 \cdot E1 = 0$
$1 \cdot 1 = 1$	$1 \cdot E1 = E1$
$1 \cdot 0 = 0$	$E1 \cdot E1 = 0$
$1 \cdot 0 = 1$	$E1 \cdot E1 = 1$

\cdot UND Funktion ist 0 dominant; ODER Funktion ist 1 dominant

1.2.1. Boolesche Theoreme (Regeln, Gesetze):

Ein Theorem in der Booleschen Algebra ist eine Regel, die eine fundamentale Beziehung zwischen den Booleschen Variablen zum Ausdruck bringt. Die Anwendung der Theoreme wird es erlauben, logische Schaltungen in vielfältiger Form zu manipulieren bzw. zu vereinfachen

Im folgenden werden die Theoreme vorgestellt:

1. Kommutativgesetz (Vertauschungsgesetz)
2. Assoziativgesetz (Zusammenfassungsgesetz)
3. Distributivgesetz (Verteilungsgesetz)
4. Idempotenzgesetz (Identitätsgesetz)
5. Absorptionsgesetz (Verschmelzungsgesetz, Redundanzgesetz)
6. Null- und Eins-Gesetz
7. Komplement-Gesetz
8. De Morgansches Theorem

Kommutativgesetz (Vertauschungsgesetz):

Disjunktion und Konjunktion sind kommutativ (vertauschbar).

Algebraische Form: $a + b = b + a$
 $a \cdot b = b \cdot a$

Interpretation: Es ist offenbar gleich in welcher Reihenfolge die Variablen verknüpft werden.

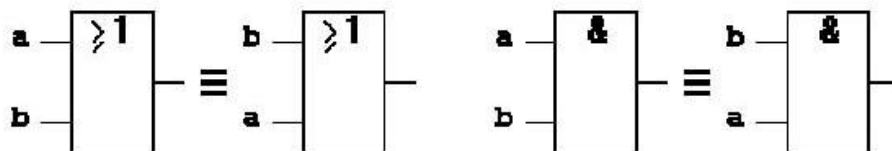


Abb.1.3: Es ist gleich, in welcher Reihenfolge die Eingangsvariablen an die Eingänge der UND und ODER Gatter gelegt werden

Assoziativgesetz (Zusammenfassungsgesetz):

Disjunktion und Konjunktion sind assoziativ (in Klammern zusammen fassbar).

Algebraische Form: $(a + b) + c = a + (b + c)$
 $(a \cdot b) \cdot c = a \cdot (b \cdot c)$

Interpretation: Es ist offenbar gleich in welcher Reihenfolge die Variablen verknüpft werden.

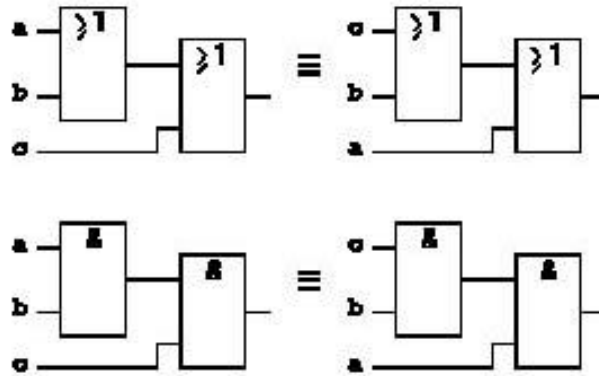


Abb.1.4: Klammerregel bei gleichen Operatoren - Jeder algebraischen Klammerung entspricht einer Gatterebene in der Symboldarstellung

Distributivgesetz (Verteilungsgesetz):

Disjunktion und Konjunktion sind distributiv (ODER verteilt sich über UND; UND verteilt sich über ODER).

Algebraische Form: $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$
 $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c) = a \wedge b \vee a \wedge c$

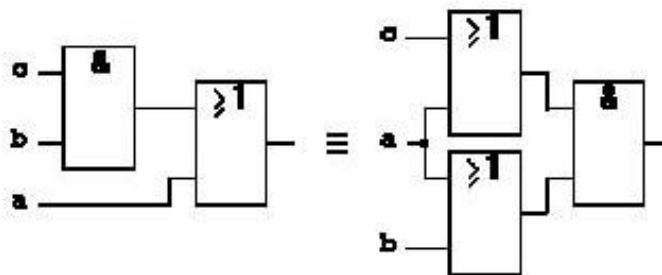


Abb.1.5: Klammerregel bei unterschiedlichen Operatoren

Idempotenzgesetz (Identitätsgesetz)

Algebraische Form: $a \vee a = a$
 $a \wedge a = a$

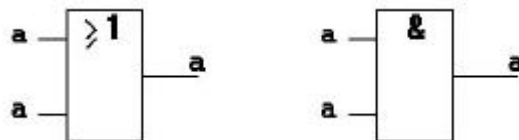


Abb.1.6: Die Gatterschaltung ist logisch redundant. Sie verzögert allerdings das Signal und hat damit schaltungstechnische Bedeutung

Absorptionsgesetz (Verschmelzungsgesetz)

Algebraische Form: $a \vee (a \wedge b) = a$
 $a \wedge (a \vee b) = a$

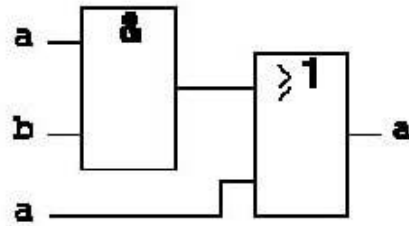


Abb.1.7: Redundanzgesetz

Null und Eins Gesetz

Algebraische Form: $0 \vee a = a$ $1 \wedge a = a$
 $0 \wedge a = 0$ $1 \vee a = 1$

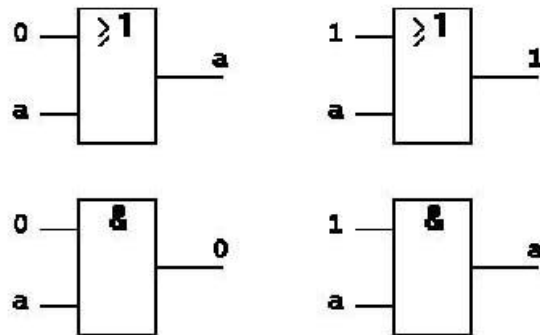


Abb.1.8: UND Funktion ist 0 dominant, ODER Funktion ist 1 dominant

Komplement Gesetz

Algebraische Form: $a \vee \bar{a} = 1$
 $a \wedge \bar{a} = 0$

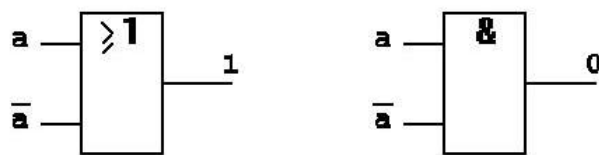


Abb.1.9: Zu jedem Element a existiert ein komplementäres Element \bar{a}

De Morgan'sches Theorem

Algebraische Form: $\overline{a \wedge b} = \bar{a} \vee \bar{b}$
 $\overline{a \vee b} = \bar{a} \wedge \bar{b}$

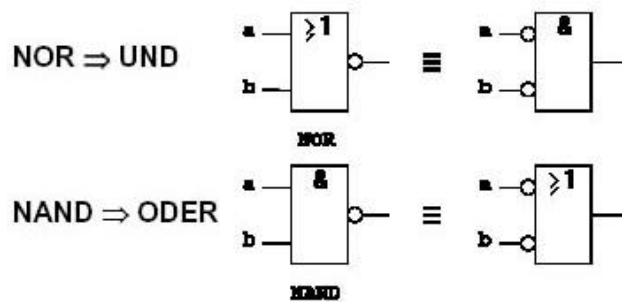


Abb.1.10: Jedes Gesetz der Schaltalgebra bleibt gültig, wenn 0 und 1 vertauscht wird, sowie UND und ODER; die Negation wirkt auf die Variable und den Operator

Tipps:

Elektronische Steuerungen sind meist binäre oder digitale Steuerungen. Bei binären Steuerungen werden zweiwertige Steuersignale, also Schaltsignale, miteinander verknüpft, gespeichert und als Schaltsignale wieder ausgegeben. Die Art der Signalverknüpfung wird durch die Schaltalgebra beschrieben.

Bei digitalen Steuerungen werden mehrere Binärsignale oder eine Serie von binären Impulsen als Zahlen codiert, gespeichert und verarbeitet.

Die Art der Zahlenverknüpfung wird durch die arithmetischen Rechenregeln, wie z.B. Addition und Multiplikation, beschrieben. Die Bausteine digitaler Steuerungen sind meist integrierte Schaltkreise (IC), Mikroprozessoren und Mikrorechner.

2. Kombinatorische Schaltungen, Karnaugh-Veitch

2.1. Kombinatorische Schaltungen

Eine Steuerung, deren Steuersignale nur von der Kombination ihrer Eingangssignale abhängen, nennt man eine kombinatorische Steuerung. Die zugehörige Schaltung kann mit UND-, ODER- und NICHT-Gliedern aufgebaut werden. Zur Ermittlung der Schaltung stellt man bei unübersichtlichen Schaltbedingungen zunächst die vollständige Funktionstabelle auf und bildet dann entweder die ODER-Normalform oder die UND-Normalform.

Die vollständige Funktionstabelle enthält alle Kombinationen der Eingangsvariablen und die dazugehörigen Ausgangsvariablen.

Da jede Eingangsvariable die Werte 1 und 0 annehmen kann, verfügt eine Schaltung mit n -Eingangsvariablen, entsprechend der möglichen Dualzahlen, über 2^n Kombinationen.

Die *ODER-Normalform* gibt alle Schaltfunktionen wieder, die die gestellte Aufgabe lösen können. Sie enthält die UND-Verknüpfungen aller Eingangsvariablen für alle Zeilen der Funktionstabelle, in denen das Ausgangssignal zu 1 wird.

Die *UND-Normalform* gewinnt man aus den Schaltfunktionen, die die gestellte Aufgabe nicht lösen. Negiert man diese Schaltfunktionen, so erhält man eine neue Schaltfunktion, die nun eine Lösung der Aufgabe darstellt. Durch diese Negation erhält man die UND-Verknüpfungen aller überhaupt auftretenden Eingangsvariablen, die untereinander Oder-verknüpft sind.

Die UND-Normalform ist der ODER-Normalform vorzuziehen, wenn die Ausgangsvariable in der vollständigen Funktionstabelle seltener 0 ist als 1. Sie enthält dann weniger Verknüpfungen.

Vereinfachen der Schaltfunktionen

Mit Hilfe der Rechenregeln der Schaltalgebra können die Schaltfunktionen umgewandelt und vereinfacht werden. Da die Anwendung der Rechenregeln jedoch sehr aufwendig sein kann,

verwendet man zur Vereinfachung der Schaltfunktionen eine graphische Darstellung, das Karnaugh-Veitch Diagramm (KV-Diagramm).

2.2. KV-Diagramm

Das Karnaugh-Veitch-Diagramm für 2, 3 und 4 Variablen hat $2^2 = 4$, $2^3 = 8$ bzw. $2^4 = 16$ Felder, z.B. bei den 3 Eingangsvariablen a, b, und c 8 Felder.

Den Eingangsvariablen wird jeweils 1 Streifen zugeordnet. Da sich diese Streifen überkreuzen, kommen in den n-Feldern alle Kombinationen der Eingangsvariablen vor. Man schreibt nun, ausgehend von der vollständigen Funktionstabelle, in diejenigen Felder, die die Schaltbedingungen erfüllen, eine 1 ein. Benachbarte Felder mit 1 können durch UND-Verknüpfungen zu 2er-, 4er- oder 8er-Blöcke zusammengefasst werden. Man fasst möglichst viele Felder zusammen, sucht also in 2er-, 4er- oder 8er-Blöcken zusammenhängende 1-Gebiete. Die Vereinfachung besteht nun darin, dass alle Eingangsvariablen verschwinden, die sich innerhalb eines Blockes ändern. Es bleiben nur die Variablen erhalten, die allen Felder eines Blockes gemeinsam sind.

Anstelle der 1-Gebiete kann man auch 0-Gebiete suchen und die Schaltfunktion negiert anschreiben.

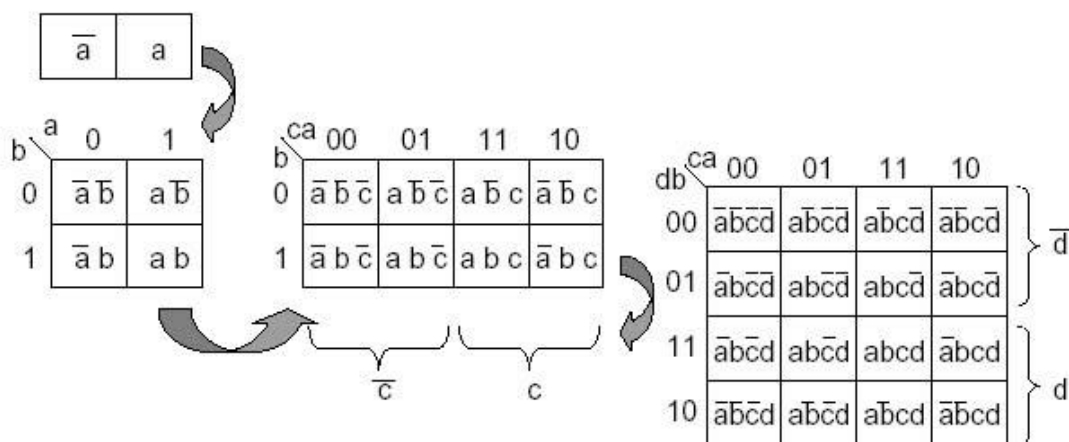


Abb.2.1: Konstruktion des KV-Diagrammes (Darstellung für 1, 2, 3 bzw. 4 Variablen)

BEISPIEL

Eine Schriedemaschine muss von 3, mindestens jedoch von 2 Personen bedient werden. Es sind 3 gleiche Bedienungsplätze mit den Schlüsselschaltern a, b und c eingerichtet. Damit die Maschine durch das Signal $x = 1$ eingeschaltet werden kann, müssen mindestens 2 oder 3 Schlüsselschalter geschaltet sein.

1) Aufstellen einer vollständigen Funktionstabelle

Vollständige Funktionstabelle mit 3 Variablen

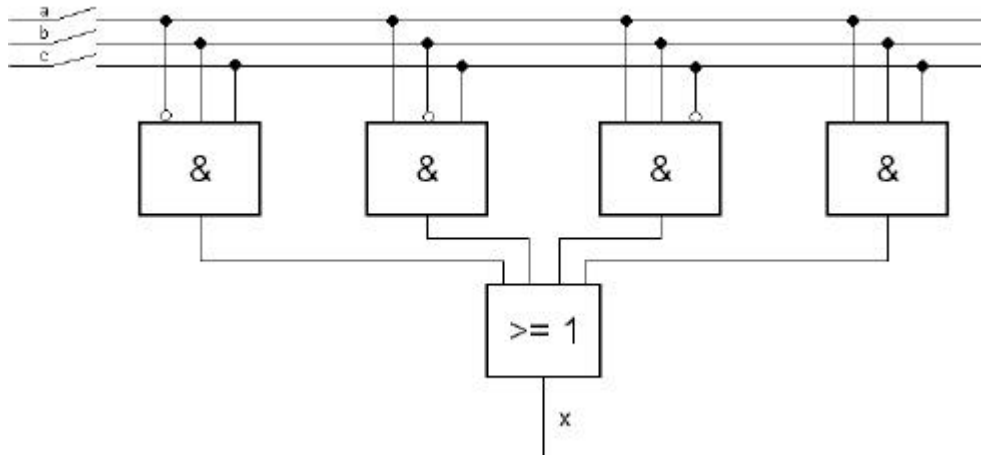
	a	b	c	x	
0	0	0	0	0	Zur Bildung der UND-Normalform
1	0	0	1	0	
2	0	1	0	0	Zur Bildung der ODER-Normalform
3	0	1	1	1	
4	1	0	0	0	
5	1	0	1	1	
6	1	1	0	1	
7	1	1	1	1	

2) Aufstellen der ODER-Normalform für die Funktionstabelle und zeichnen des Funktionsplans

ODER-Normalform

$$x = (\bar{a} \wedge b \wedge c) \vee (a \wedge \bar{b} \wedge c) \vee (a \wedge b \wedge c) \vee (a \wedge b \wedge \bar{c})$$

Funktionsplan für die ODER-Normalform mit 3 Variablen



3) Aufstellen der UND-Normalform für die Funktionstabelle und zeichnen des Funktionsplans

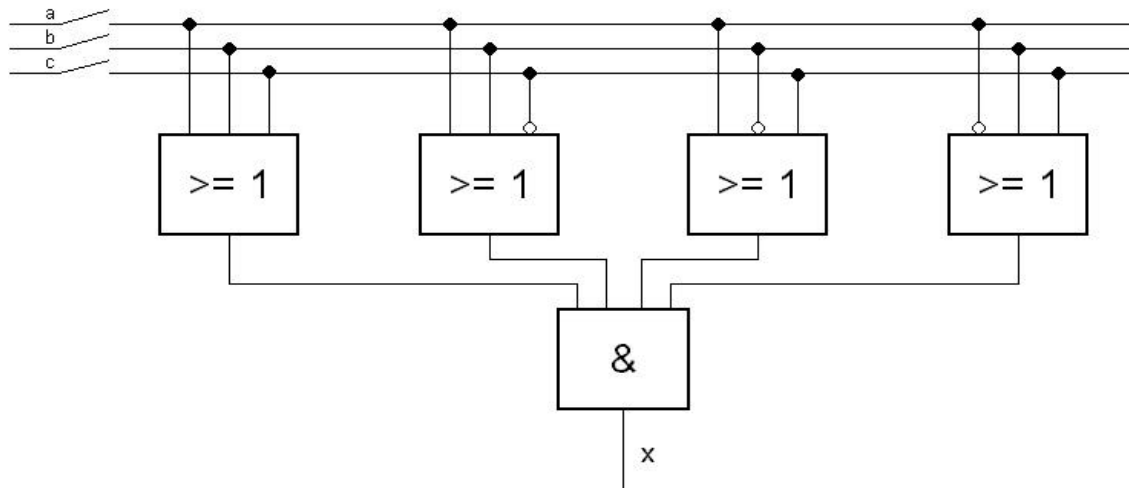
UND-Normalform

$$\bar{x} = (\bar{a} \wedge \bar{b} \wedge \bar{c}) \vee (\bar{a} \wedge \bar{b} \wedge c) \vee (a \wedge \bar{b} \wedge c) \vee (a \wedge b \wedge \bar{c})$$

Mit Hilfe des De Morgan Theorems folgt:

$$x = (a \vee b \vee c) \wedge (a \vee b \vee \bar{c}) \wedge (a \vee \bar{b} \vee c) \wedge (a \vee \bar{b} \vee \bar{c})$$

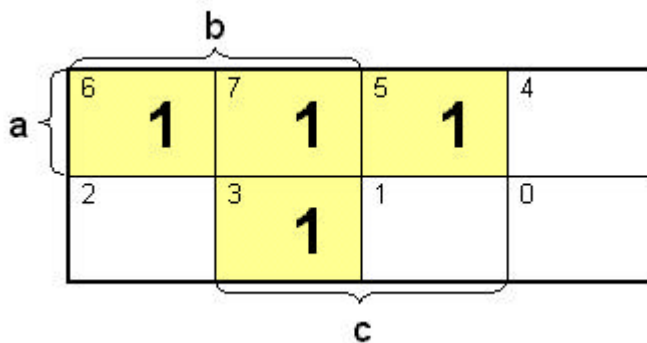
Funktionsplan für die UND-Normalform mit 3 Variablen



4) Aufstellen des Karnaugh-Diagramms für die Funktionstabelle

Es kommen 3 Eingangsvariablen vor. Das Karnaugh-Diagramm hat $2^3 = 8$ Felder. Die Felder, in denen sich 2 oder 3 Streifen überkreuzen, markiert man mit 1, da in der Aufgabenstellung mindestens 2 der 3 Schalter a, b, c eingeschaltet sein sollen. Entsprechend der Funktionstabelle sind dies die Felder 3, 5, 6, 7.

Karnaugh-Diagramm mit 3 Variablen



Aus dem Karnaugh-Diagramm kann man nun die 2er-Blöcke der 1-Felder mit der Gleichung $x = (a \wedge c) \vee (a \wedge b) \vee (b \wedge c)$ erfassen.

Damit erhält man einen vereinfachten, jedoch gleichwertigen Funktionsplan. Anstelle der vier Terme mit je drei Variablen sind nur noch 3 Terme mit je zwei Variablen erforderlich.

2.3. SR/RS-Speicherfunktionen

Oft ist erforderlich, ein kurzzeitiges Signal zu speichern. In einer SPS werden Merker hierfür eingesetzt. Einen Teil der Merker kann man remanent einstellen, d.h. dieser Teil behält dann seinen Signalzustand auch im spannungslosen Zustand der SPS bei. Eine Signalspeicherung kann mit binären Grundverknüpfungen verwirklicht werden.

Fast die gleiche Speicherfunktion zeigt der RS-Speicher. Ein Unterschied besteht dann, wenn am Setz- und am Rücksetzeingang gleichzeitig ein 1-Signal ansteht. Bedingt durch die fortlaufende Bearbeitung der Anweisungen setzt die CPU mit dem zuerst bearbeiteten Setzeingang beim SR-Speicher den Speicheroperanden (Merker, Ausgang, ...), setzt ihn jedoch anschließend beim Bearbeiten des Rücksetzeingangs wieder zurück. Für den Rest der Programmbearbeitung bleibt der Speicheroperand zurückgesetzt. Man sagt, der SR-Speicher sei rücksetzdominant. Ist der Speicheroperand ein Ausgang, findet dieses kurzzeitige Setzen nur im Prozessabbild der Ausgänge statt, der Ausgang auf der dazugehörigen Ausgabebaugruppe wird nicht beeinflusst.

Wenn dagegen beim RS-Speicher am Setz- und Rücksetzeingang gleichzeitig ein 1-Signal ansteht, setzt die CPU mit dem zuerst bearbeiteten Rücksetzeingang den Speicheroperanden zurück, setzt ihn jedoch anschließend beim Bearbeiten des Setzeingangs wieder. Für den Rest der Programmbearbeitung bleibt der Speicheroperand gesetzt. Man sagt, der RS-Speicher sei setzdominant. Ist der Speicheroperand ein Ausgang, gilt gleiches wie beim SR-Speicher. Soll sichergestellt sein, dass der Speicheroperand nach abgeschalteter Spannung ein 0-Signal hat, sollte ein Merker verwendet werden, der nicht remanent einstellbar ist. Dieses unterschiedliche Speicherverhalten kann bei Steuerungen sinnvoll eingesetzt werden, wenn z.B. bei Fehlbedienung an einer Anlage eine Funktion unterbleiben soll.

Speicherfunktionen können mit Hilfe einer UND-Verknüpfung am Setzeingang gegenseitig verriegelt werden. Der Speicheroperand kann nur gesetzt werden, wenn der jeweils andere Speicheroperand ein 0-Signal hat.

2.4. Konnektor und Zuweisung

Ein Konnektor ist ein Zwischenspeicher innerhalb einer binären Verknüpfung. Das bis zu dieser Stelle gültige Verknüpfungsergebnis wird in dem Binäroperanden des Konnektors gespeichert. Der zum Konnektor gehörende Binäroperand kann an anderer Stelle im Programm wieder abgefragt werden. Ein Konnektor darf keine Verknüpfung abschließen. Dies ist nur mit einer Zuweisung möglich.

In der Steuerungstechnik muss mitunter ein zeitlich anstehendes Signal in einen Impuls umgewandelt werden. Dies ist mit einem Wischkontakt möglich. Mit Hilfe eines Konnektors kann man einen Wischkontakt programmieren.

2.5. Flankenwertung

Mit einer Flankenwertung erfasst man die Änderung eines Signalzustands, eine Signalfanke. Eine positive (steigende) Flanke liegt vor, wenn das Signal von Zustand 0 nach Zustand 1 wechselt. Im

umgekehrten Fall spricht man von einer negativen (fallenden) Flanke. Für die Flankenauswertung gibt es vier verschiedene Elemente. Flankenauswertung ist bei einem Verknüpfungsergebnis oder einem Operanden möglich. Die Flankenauswertung eines Operanden steht am Anfang einer binären Verknüpfung.

Elemente zur Flankenauswertung:

Art der Flanke	Symbol	
	Auswertung eines Verknüpfungsergebnisses	Auswertung eines Operanden
positiv	<p>The symbol consists of a box with two input lines on the left. Inside the box, the symbol ≥ 1 is positioned above the letter 'P'. To the right of the box, the text 'M 0.1' is written above the box itself. An output line with a double arrow points to the right from the box.</p>	<p>The symbol consists of a box with one input line on the left. Inside the box, the text 'E 0.1' is above 'PDS'. To the left of the box, the text 'M 0.3' is written above the box. To the right of the box, the text 'M BIT Q' is written above the box. An output line with a double arrow points to the right from the box.</p>
negativ	<p>The symbol consists of a box with two input lines on the left. Inside the box, the symbol '&' is positioned above the letter 'N'. To the right of the box, the text 'M 0.2' is written above the box itself. An output line with a double arrow points to the right from the box.</p>	<p>The symbol consists of a box with one input line on the left. Inside the box, the text 'E 0.0' is above 'NEG'. To the left of the box, the text 'M 0.0' is written above the box. To the right of the box, the text 'M BIT Q' is written above the box. An output line with a double arrow points to the right from the box.</p>

Zur Programmierung eines Anwenderprogramms können die binären Verknüpfungen NICHT, UND, ODER eingesetzt werden. Außerdem stehen die SR-Funktion und RS-Funktion, die Zuweisung und die Konnektorfunktion zur Signalspeicherung zur Verfügung. Die Elemente zur Flankenauswertung können ebenfalls bei der Programmierung zur Signalspeicherung verwendet werden.

3. Zeitfunktionen

Mit den Zeitfunktionen kann man programmtechnisch zeitliche Abläufe verwirklichen, wie z.B. Wartezeiten und Überwachungszeiten, Messungen einer Zeitspanne oder die Bildung von Impulsen.

Die Verhaltensweisen einer Zeitfunktion sind:

- ?? Impulsbildung
- ?? Verlängerter Impuls
- ?? Einschaltverzögerung
- ?? Speichernde Einschaltverzögerung
- ?? Ausschaltverzögerung

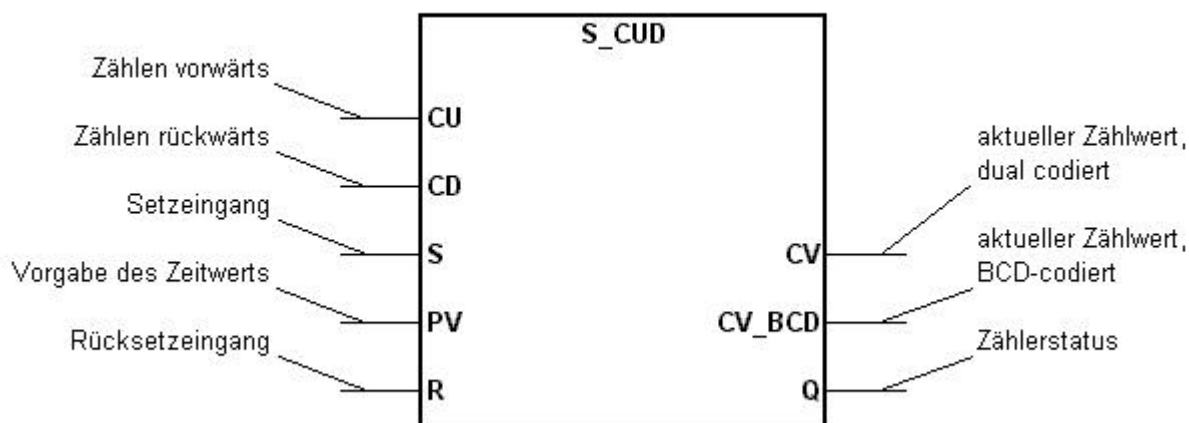
Eine Zeitfunktion startet, wenn das Verknüpfungsergebnis vor dem Starteingang wechselt. Bei einer Ausschaltverzögerung (SF) muss das Verknüpfungsergebnis von 1 nach 0 wechseln, alle anderen Zeitfunktionen werden mit einem Wechsel des Verknüpfungsergebnisses von 0 nach 1 gestartet. Die Vorgabe der Zeitdauer übernimmt der am TV-Eingang stehende Wert.

Starten der Zeit als	Start-signal	FBS/FUP	AWL
Impuls			L <Zeitwert> SP <Operand>
verlängerter Impuls			L <Zeitwert> SE <Operand>
Einschalt-verzögerung			L <Zeitwert> SD <Operand>
speichernde Einschalt-verzögerung			L <Zeitwert> SS <Operand>
Ausschalt-verzögerung			L <Zeitwert> SF <Operand>

4. Zählfunktionen

Mit den Zählfunktionen können Zählaufgaben ausgeführt werden. Die Zählfunktion enthält in einer Box die zusammenhängende Darstellung aller Zähloperationen in Form von Funktions-ein- und -ausgängen. Adressiert wird in absoluter oder symbolischer Form, z.B. C4 oder „Stückzahl 1“.

Zähloperand



Vorwärtszählen CU (Count UP)

Die Zählfunktion wird vorwärts gezählt, wenn das Verknüpfungsergebnis am Eingang CU von 0 nach 1 (positive Flanke) wechselt. Jeder Wechsel erhöht den Zählerwert um eine Einheit.

Rückwärtszählen CD (Count Down)

Die Zählfunktion wird rückwärts gezählt, wenn das Verknüpfungsergebnis am Eingang CD von 0 nach 1 (positive Flanke) wechselt. Jeder Wechsel verringert den Zählerwert um eine Einheit.

Vorgabe des Zählwerts PV (Programmed Value)

Die Zählfunktion übernimmt beim Setzen den am Eingang PV stehenden Wert als Zählwert. Die Vorgabe des Zählwerts kann als Konstante erfolgen, z.B. C#50 für Zählerwert 50.

Zähler setzen S

Ein Zähler wird gesetzt, wenn das Verknüpfungsergebnis vor dem Setzeingang S von 0 nach 1 (positive Flanke) wechselt. Zähler setzen heißt, die Zählfunktion wird auf einen Anfangswert gesetzt.

Zähler rücksetzen R

Eine Zählfunktion wird rückgesetzt, wenn am Rücksetzeingang R der Signalzustand 1 ansteht. Das Rücksetzen der Zählfunktion setzt den Zählwert auf „Null“.

Zählwert abfragen CV / CV BCD

Der Ausgang CV stellt den aktuellen Zählwert dual codiert und der Ausgang CV_BCD stellt den aktuellen Zählwert BCD-codiert zur Verfügung.

Zählerstatus abfragen Q

Der Ausgang Q führt Signalzustand 1, wenn der aktuelle Zählwert größer als Null ist.

5. Speicherprogrammierbare Steuerung SPS

5. 1. Aufbau und Funktionsweise

Eine speicherprogrammierbare Steuerung (SPS) ist ähnlich aufgebaut wie ein Computer. Sie besteht im Wesentlichen aus einem Netzteil (Spannungsversorgung), einer Signaleingabeeinheit, einer Zentralbaugruppe mit Mikroprozessor, Programmspeicher sowie weiteren Funktionseinheiten und einer Signalausgabeeinheit. Im Gegensatz zu verbindungsprogrammierten Steuerungen, bei denen der Steuerungsablauf durch die eingesetzten Bauelemente und deren Leitungsverbindungen festgelegt wird, sind die Steuerungseigenschaften bei einer SPS als Programm im Programmspeicher gespeichert.

Eine SPS wird meist als Verarbeitungsteil einer Steuerung eingesetzt. Für Steuerungen von geringem Umfang, d.h. maximal ca. 100 DI/DO (DI von engl. Digital-Input = digitaler Eingang, DO von engl. Digital-Output = digitaler Ausgang), werden Kompakt-SPS, auch Micro-SPS genannt, verwendet. Bei großen Steuerungsaufgaben mit mehr als 100 DI/DO werden aufgebaute SPS eingesetzt.

Oft wird auch an jeder Fertigungseinheit eine lokale SPS installiert. Diese lokalen SPS werden dann über einen BUS (Sammelleitung für Datenaustausch), z.B. PROFIBUS-DP, mit einer Masterbaugruppe verbunden. Die nachgeordneten SPS werden als Slave (engl.: Sklave) bezeichnet. Es können weitere Komponenten angeschlossen werden, wie z.B. Bedien- oder Beobachtungsgeräte oder eine ASI (Aktuator-Sensor-Interface)-Buskopplung für dezentrale Signalverarbeitung.

Eine SPS kann auch in andere Mikroprozessorsteuerungen integriert sein, z.B. in eine CNC-Steuerung. Die SPS wird dabei über das CNC-Programm angesprochen. Sind nur wenige SPS-Funktionen notwendig, dann werden diese nur softwaremäßig im Betriebsprogramm der CNC-Steuerung verwirklicht. In diesen Fällen wird die SPS auch PLC (engl.: programmable logic controller = programmierbare logische Steuerung) genannt. Immer häufiger werden auch Industrie-PC als SPS eingesetzt. In diesem Fall übernimmt eine Steckkarte im PC die Funktionen der SPS.

Eine SPS gibt es in den Bauformen kompakt, modular, vernetzt, integriert und als Industrie-PC.

Am häufigsten werden derzeit modulare SPS eingesetzt. Die Zentralbaugruppe einer modularen SPS besteht aus einem Netzteil, der CPU (engl.: central processing unit = zentrale Verarbeitungseinheit) und mindestens einer digitalen Eingabe- und einer digitalen Ausgabebaugruppe.

Zum Betrieb muss die CPU an eine Spannungsquelle, meist 24 V DC, angeschlossen sein. Um bei Netzausfall gespeicherte Daten nicht zu verlieren, dient bei unterbrochener Spannungsversorgung eine Pufferbatterie als Energiequelle.

Steht der Betriebsartenschalter in STOPP-Position, können Anwenderprogramme in die CPU übertragen werden. Zur Datenübertragung zwischen dem Programmiergerät und der CPU dient meist die mehrpunktfähige Schnittstelle MPI (engl.: Multi Point Interface = Mehrpunkt Schnittstelle). Die MRES (engl.: memory reset = Speicher zurücksetzen)-Position des Betriebsartenschalters dient zum Löschen der CPU. Ist die RUN-P-Einstellung gewählt, wird das Anwenderprogramm abgearbeitet und in der RUN-Einstellung können während dessen zusätzliche Variablen verändert werden.

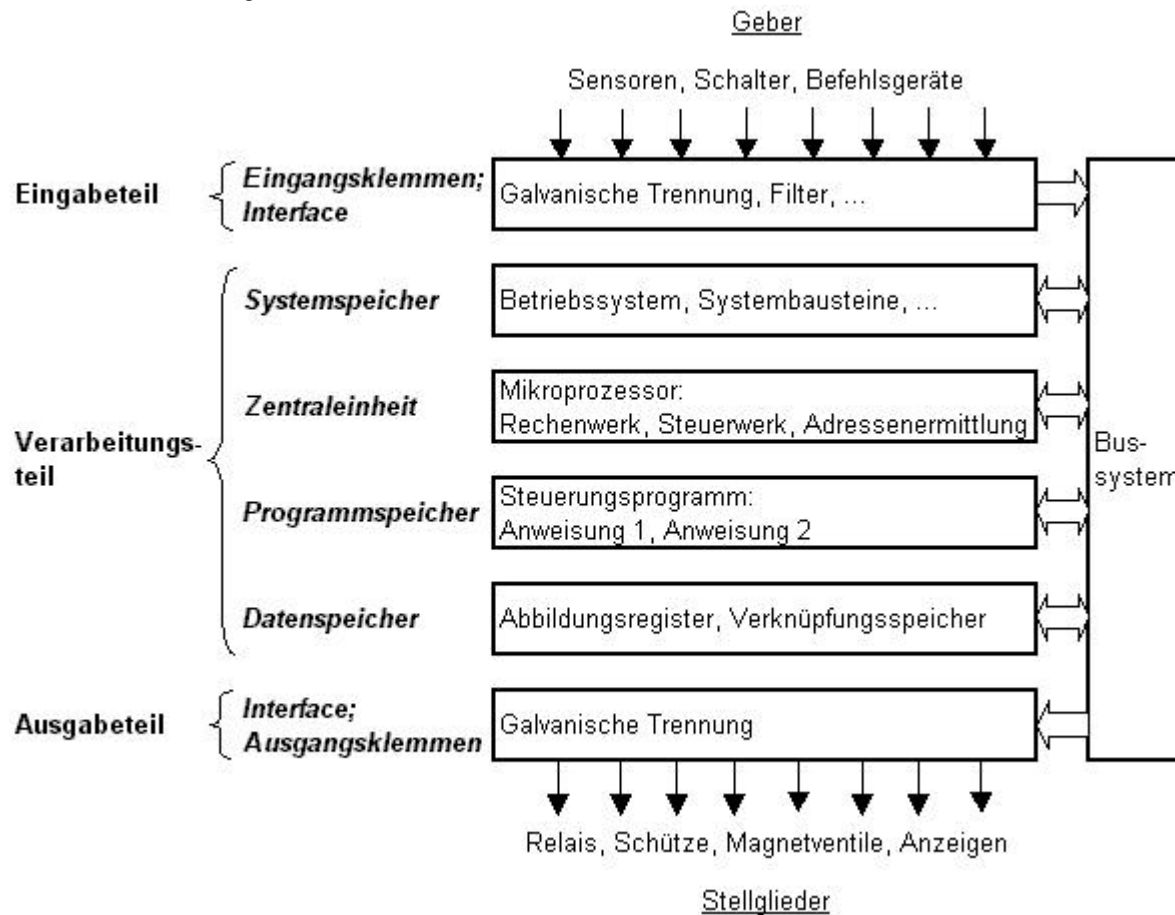
Über die PROFIBUS-DP-Schnittstelle kann die Verbindung zu weiteren Baugruppen hergestellt werden die auch als dezentrale Peripherie bezeichnet werden.

Die Status- und Fehleranzeigen geben Auskunft über den jeweiligen Betriebszustand der DPU. Bei einer Fehleranzeige, SF oder SF DP, können einem Diagnosepuffer genaue Informationen über die Art des Fehlers entnommen werden. Die CPU beinhaltet unter anderem den Mikroprozessor und

verschiedene Speicherbereiche, Programm-, Daten- und Systemspeicher, die über eine Busleitung miteinander verbunden sind.

Funktionsblöcke einer SPS

Eine SPS enthält folgende Funktionsblöcke:



Eingabeteil

Im Eingabeteil werden die ankommenden externen Signale der Geber für die interne Verarbeitung aufbereitet.

Die meisten SPS arbeiten mit digitalen Eingangsspannungen von + 24 V.

Dabei wird der positiven Spannung der Signalzustand „1“ zugeordnet.

Ein offener oder mit 0 V beschalteter Eingang hat den Signalzustand „0“.

Drahtbrüche ergeben also 0-Signal am Eingang der SPS.

Diese hohen Eingangsspannungen werden über Optokoppler von der internen Signalspannung von 5 V galvanisch getrennt, um Störspannungen im Verarbeitungsteil der SPS zu vermeiden.

Kurzzeitige Spannungsspitzen und Störungen auf den Geberleitungen werden über Zeitverzögerungen ausgefiltert. Diese Zeitverzögerungen überbrücken auch das Kontaktprellen mechanischer Geber.

Systemspeicher

Der Systemspeicher wird vom Hersteller der SPS programmiert. In ihm steht die Software, die das programmieren und Arbeiten der SPS ermöglicht.

Die Systemsoftware überwacht und führt den Dialog zwischen Programmierer und Steuerung. Sie macht auf Bedienfehler und Störungen aufmerksam.

Weiter enthält sie ein Übersetzerprogramm, welches die leicht erfassbare Programmiersprache, in der die Steueranweisungen eingegeben werden, in das Maschinenprogramm übersetzt.

Auch Programm-Funktionsbausteine wie Zeitwerke, Zähler oder Vergleicher gehören zur Systemsoftware und lassen sich vom Anwender direkt aufrufen.

Der Systemspeicher ist aus EPROM-Speicherbausteinen aufgebaut. Das Programm wurde vom Hersteller in den Baustein „gebrannt“ und lässt sich nur mit UV-Licht löschen. Der Baustein behält bei Netzausfall seine Informationen.

Der Systemspeicher enthält die Variablen, auch Operanden genannt, die vom Programm angesprochen werden. Die Variablen werden zu Bereichen, den Operandenbereichen, zusammengefasst. Die Größe der Bereiche ist abhängig von der eingesetzten CPU.

Operandenbereiche der CPU:

Eingänge (E) sind ein Prozessabbild der Digitaleingabebaugruppen

Ausgänge (A) sind ein Prozessabbild der Digitalausgabebaugruppen

Merker (M) sind Informationsspeicher

Zeitfunktionen (Timer, T) stellen Zeitglieder dar, mit denen Warte- und Überwachungszeiten verwirklicht werden

Zählfunktionen (Z) sind Softwarezähler, die vorwärts und rückwärts zählen können

Programmspeicher

Der Programmspeicher enthält das Steuerungsprogramm des Anwenders. Die Anweisungen zur Verknüpfung der Eingangssignale und zur Speicherung und Ausgabe der Ergebnisse sind nacheinander im Programmspeicher abgelegt. Diese Anweisungen sagen dem Mikroprozessor was er der Reihe nach machen soll.

Da das Anwenderprogramm verändert wird, muss es in einem Schreib-Lese-Speicher stehen. Solch ein Schreib-Lese-Speicher lässt sich aus RAM-Bausteinen aufbauen. Da sie bei Netzausfall ihren Inhalt verlieren, ist ihre Versorgungsspannung über Batterien zu puffern.

Ein erprobtes Anwenderprogramm kann in einem EPROM-Speicher geschrieben, und so vor Informationsverlust bei Netzausfall geschützt werden.

Datenspeicher

Der Datenspeicher dient zur Ablage der variablen Daten. Solche variablen Daten sind die Abbilder der Eingangssignale oder der Ausgangssignale. Auch Zwischenergebnisse der logischen Verknüpfungen werden darin festgehalten. Er ist aus RAM-Bausteinen aufgebaut.

Speichertyp		Löschen	Programmieren	Speicherinhalt ist bei Stromabschaltung
RAM	Random Access Memory; "Speicher mit wählfreiem Zugriff"; Schreib-Lese-Speicher	elektrisch	elektrisch	flüchtig
ROM	Read Only Memory; Nur-Lese-Speicher; Festwertspeicher	nicht möglich	durch Masken beim Herstellungsprozess	nicht flüchtig
PROM	Programmable ROM; Programmierbarer Festwertspeicher		elektrisch	
EPROM	Eraseable PROM; Löschbarer, programmierbarer Festwertspeicher	durch UV-Licht		
EEPROM	Electrically Erasable PROM; Elektrisch löschbarer, programmierbarer Festwertspeicher	elektrisch		

Ausgabeteil

Handelsübliche SPS haben Relais- oder Transistorausgänge.

Relaisausgänge können Leistungsschütze mit 220 V Spulenspannung schalten. Die Kontakte haben meist eine Schutzbeschaltung.

Die Ausgangstransistoren werden vom Verarbeitungsteil zur galvanischen Trennung über Optokoppler angesteuert.

Sie können z.B. 24 V Gleichstrom-Stellglieder schalten.

Bus-System

Das Bus-System besteht aus einem Bündel paralleler Leitungen. Über diese Leitungen wählt der Mikroprozessor die Speicher- oder Schnittstellen an (Adress-Bus).

Über den Datenbus liest der Prozessor die Daten des Eingabeteils oder des Programms. Die Ergebnisse der Verarbeitung schreibt er über den Datenbus in den Datenspeicher oder in das Ausgabeteil.

Zu den Adress- und Datenbus kommen noch einige Steuerleitungen (Steuerbus).

Arbeitsweise einer SPS

Die SPS arbeitet das Steuerungs- oder Anwenderprogramm von der ersten bis zur letzten Anweisung der Reihe nach ab und beginnt dann wieder von vorn.

Diese zyklische Abarbeitung ist charakteristisch für speicherprogrammierbare Steuerungen. Die kurze Zykluszeit ist ein wesentliches Qualitätsmerkmal. Sie kann im Millisekundenbereich liegen.

Bevor die SPS mit dem Bearbeitungszyklus beginnt, fragt sie einmal die Zustände aller Eingänge ab und speichert deren Signale im Eingangs-Abbildungsregister.

Dieses Zustandsbild aller Eingänge bleibt für einen Zyklus konstant. Im Bearbeitungszyklus werden die gespeicherten Werte im Eingangsregister entsprechend dem Anwenderprogramm miteinander verknüpft. Die SPS speichert die Verknüpfungsergebnisse im Ausgangs-Abbildungsregister und gibt sie am Ende des Bearbeitungszyklus parallel aus. Dadurch entsteht der Eindruck einer parallelen Verarbeitung. Dazu trägt auch die kurze Zykluszeit bei.

Der Anwender muss die zyklische Bearbeitung beim Entwurf seines Steuerungsprogramms berücksichtigen. In einigen Fällen lässt sich diese Arbeitsweise vorteilhaft nutzen, wenn z.B. Signale oder Voreinstellungen nur über eine Zykluszeit wirksam werden sollen.

Wird ein Ausgangssignal in einem Zyklus mehrfach verändert, dann bestimmt die gegen Programmende als letztes vor der Ausgabe stehende Anweisung den endgültigen Ausgabezustand.

Damit lässt sich z.B. bei einem RS-Flipflop durch die Reihenfolge der Programmierung festlegen, ob der Setz- oder Rücksetz-Eingang Vorrang haben soll. Die zuletzt programmierte Anweisung hat Vorrang.

Mechanischer Aufbau der SPS

Speicherprogrammierbare Steuerungen gibt es als Kompaktgeräte oder als modular aufgebaute Geräte.

Kompaktgeräte sind in der Regel wie Schütze oder Zeitwerke auf eine Normschiene aufzuschnappen. Sie haben in Höhe und Tiefe ähnliche Maße wie diese.

Kompakt-SPS besitzen eine bestimmte Anzahl von Ein- und Ausgängen. Reichen diese für die zu steuernde Anlage nicht aus, lassen sich Erweiterungsgeräte anschließen.

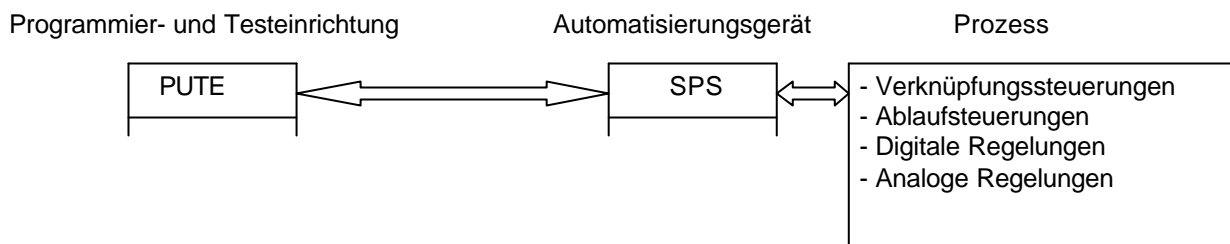
Modular aufgebaute SPS bestehen aus einzelnen Baugruppen, die auf ein Tragsystem mit Busleitungen aufgesteckt werden.

Die Funktion der SPS lässt sich durch die Zusammenstellung der Komponenten für eine Anlage maßschneidern.

Der modulare Aufbau hat außerdem den Vorteil, dass die Steuerung durch Nachbestücken mit Ein- oder Ausgabekarten oder Speichereinheiten einer geänderten Anlage angepasst werden kann.

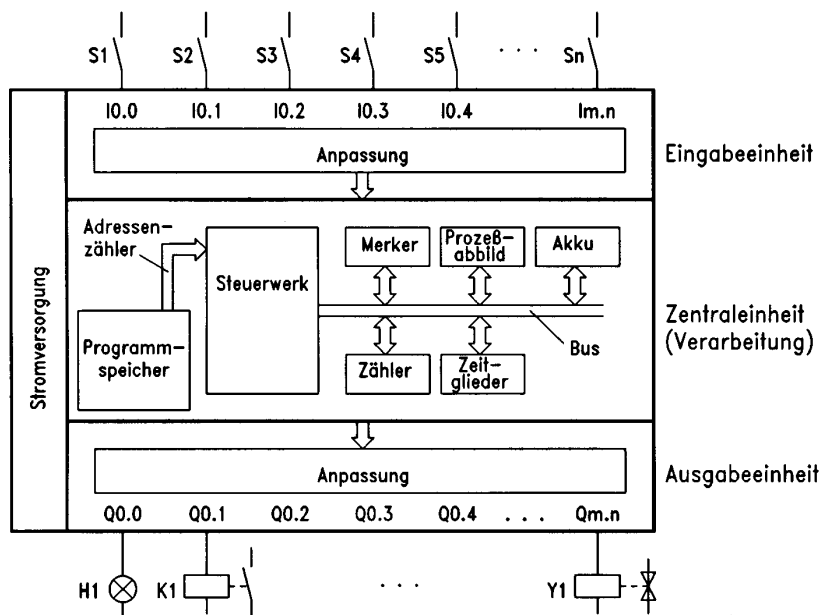
5.2. Komponenten der SPS-Technik

Es ist generell zwischen dem Automatisierungsgerät SPS und der Programmier- und Testeinrichtung für diese SPS zu unterscheiden.



Die SPS wird als Steuerung ausgeliefert und mit dem Prozess gekoppelt. Sie lässt sich nach folgenden Kriterien auswählen: Aufbau, Ein- und Ausgangssignale und Leistungsmerkmale

Die PUTE verbleibt beim Steuerungsbauer. Sie ist also für alle Steuerungen nur einmal anzuschaffen. Auf ihr wird das Steuerungsprogramm für die SPS entwickelt und in Verbindung mit der SPS getestet. Die Spannweite der angebotenen Varianten beginnt bei der Kompakt-SPS mit eingebauter



Programmierintelligenz und Handprogrammiergerät, mit dem die Anweisungsliste eingegeben und eine Zeile angezeigt wird.

Sie endet bei der modular aufgebauten SPS, für die sich das Programm auf einem Personalcomputer erstellen lässt.

Dazu gibt es eine SPS-Software, die ein komfortables

Programmieren in allen SPS-Sprachen ermöglicht. Die Dokumentation und Archivierung erfolgt ebenfalls mit Hilfe des PCs. Der Anwender bekommt ausgedruckte Pläne und Listen.

Das Anwenderprogramm lässt sich auf Diskette oder

Festplatte speichern. Die Programmerstellung und Dokumentation ist auch ohne angeschlossene Steuerung möglich.

Der Hersteller von Steuerungen muss sich also überlegen, ob er in ein einfaches Handprogrammiergerät mit alphanumerischer Zeilenanzeige oder in einen Personalcomputer mit komfortabler Software und graphischer Anzeige investieren will.

Meist ist der PC ohnehin vorhanden, es ist nur noch die Software zu beschaffen.

Leider sind sowohl Handprogrammiergerät als auch Programmiersoftware für PCs an bestimmte Hersteller, häufig sogar an bestimmte SPS-Typen gebunden.

5.3. Strukturierte Programmierung

Der wesentliche Unterschied zur linearen Programmierung besteht darin, dass ein strukturiertes Programm die Gesamtaufgabe einer SPS in einzelne und überschaubare Abschnitte aufgliedert. Zur Strukturierung wird das Programm in vier Bausteintypen aufgeteilt.

Organisationsbaustein (OB)

Er stellt die Schnittstelle zwischen Betriebssystem und Anwenderprogramm dar. In ihm wird festgelegt, in welcher Reihenfolge die einzelnen Programmabschnitte (Bausteine) bearbeitet werden sollen. Das Hauptprogramm steht im Organisationsbaustein OB1. Dieser wird zyklisch abgearbeitet.

Programmbaustein (PB)

Mit ihm programmiert man häufig wiederkehrende Automatisierungsfunktionen. Man strukturiert mit ihm das Anwenderprogramm in technologisch zusammengehörende Teile. Programmbausteine können auch andere Bausteine aufrufen.

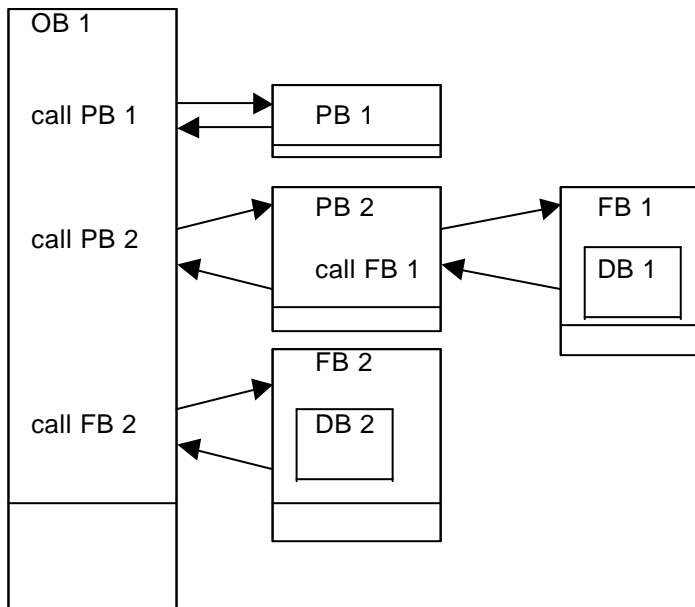
Funktionsbaustein (FB)

Funktionsbausteine enthalten Teilprogramme für häufig wiederkehrende Funktionen. Sie sind Teile des Programms, deren Aufruf über Bausteinparameter parametrierbar ist. Sie haben einen Variablenspeicher, der in einem Datenbaustein liegt.

Datenbaustein (DB)

Er beinhaltet die Zuordnung von absoluten Hardware-Adressen zu den für sie verwendeten symbolischen Namen sowie erläuternde Kommentare. Er beinhaltet auch die festen und variablen Daten des Programms, z.B. Messwert eines Drucksensors.

Strukturierung eines Programms



5.4. Programmdarstellung

Das Programm für die SPS soll sich möglichst einfach aus den bestehenden Steuerungsbeschreibungen erstellen lassen. Beschreibungsformen von Steuerungen wären z.B. der Stromlaufplan, der Logikplan, der Anschlussplan und das Programmablaufdiagramm, auch das Flussdiagramm genannt.

Kenntnisse allgemeiner Programmiersprachen sind nicht vorauszusetzen.

Deshalb haben sich Fachsprachen für die SPS-Programmierung durchgesetzt, die leicht erlernbar sind und sich an die allgemeinen Beschreibungsformen für Steuerungen anlehnen. SPS-Programmiersprachen sind im Grunde die folgenden drei: *der Kontaktplan KOP*, *der Funktionsplan FUP* und *die Anweisungsliste AWL*.

Benennungen, Zeichen und Symbole für die SPS-Programmierung sind in DIN 19239 festgelegt.

Trotz dieser Festlegungen unterscheiden sich die Programme der speicherprogrammierbaren Steuerungen verschiedener Hersteller. Deshalb muss man sich bei der Programmerstellung nach den Regeln des jeweiligen Produzenten richten.

Kontaktplan

Der Kontaktplan ähnelt dem Stromlaufplan. Er wurde in den USA entwickelt, wo Schaltungen mit Kontakten in dieser Form gezeichnet werden.

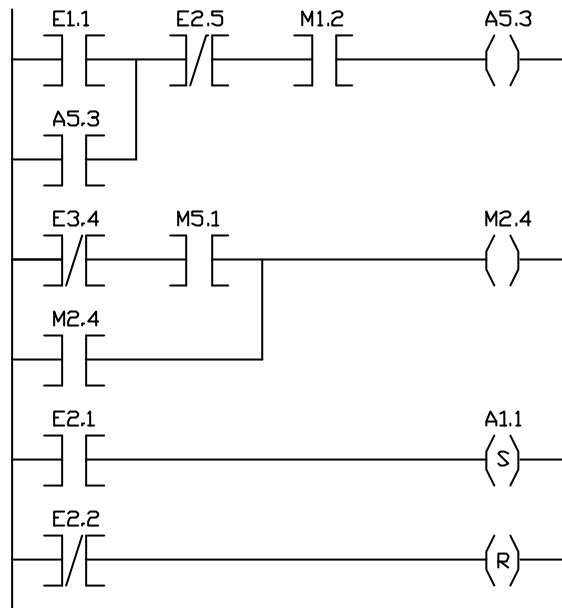
Ihn bevorzugen die Anwender mit Erfahrungen in der Schütz- und Relaisstechnik.

Er eignet sich zur Darstellung von Verknüpfungsschaltungen, die auch mit Schützen und Zeitwerken zu realisieren sind (Bitverarbeitung).

Komplexe Funktionseinheiten wie Zähler, Schieberegister oder Rechenwerke müssen im Kontaktplan wie im Funktionsplan als Kästen mit Anschlüssen gezeichnet werden. Die Innenschaltung und damit die genaue Funktion ist nicht darstellbar.

Die einzelnen Strompfade werden im Kontaktplan waagrecht dargestellt und untereinander angeordnet.

So lassen sie sich fortlaufend auf dem Bildschirm darstellen und ausdrucken.

Beispiel eines Kontaktplanes:Symbole und Kennzeichnung

Der Schaltplan und das Symbol geben an, wie das Signal logisch zu verknüpfen ist.

Symbole:

— — Schließer

— — Öffner
= Verknüpfung mit negiertem Signal

— — Spule
= Verknüpfungsergebnis

— — Verknüpfungsergebnis wird negiert

— — Verknüpfungsergebnis wird gespeichert
S = Setzen des Speichers

— — Schließer
R = Rücksetzen des Speichers

Die Symbole werden durch Buchstaben mit angefügten Nummern gekennzeichnet.

Diese Kennzeichnung gibt den Operanden an, mit dem die durch Schaltung und Symbol festgelegte logische Verknüpfung durchzuführen ist.

Kennzeichen von Operanden:

E	Eingang
A	Ausgang
M	Merker; Verknüpfungsergebnisse, die nicht auf einen Ausgang der SPS gehen
T	Zeitglied (Timer)
Z	Zähler

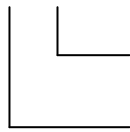
Die an das Kennzeichen angefügten Nummern werden Parameter genannt.

Die Parameter bestehen entweder aus einer fortlaufenden Zahl oder aus zwei durch einen Punkt getrennten Zahlen.

Parameter:

1, 2, 3, ... fortlaufende Zahlen

1 . 5



fortlaufende Zahl

Nummer der Baugruppe oder Funktionseinheit

z.B. Eingabe - Baugruppe 1
Ausgabe - Baugruppe 1
Merkerwort 1

Der Kontaktplan stellt die logische Verknüpfung zwischen den Eingangs- und Ausgangsklemmen der SPS dar. Er lässt keine Rückschlüsse auf die Außenbeschaltung der SPS zu.

In der Regel führt ein am Ausgang der SPS anstehendes Ausgangssignal zum Anziehen des angeschlossenen Relais oder Schützes.

Die im Kontaktplan als Öffner oder Schließer gezeichneten Eingänge stellen die logische Verknüpfung des am Eingang der SPS anliegenden Signals dar. Dabei lässt sich dieses anliegende Signal beliebig oft als Öffner oder Schließer verknüpfen. Die im Kontaktplan mit E bezeichneten Kontakte sind also nicht die im Stromlaufplan eingezeichneten echten Signalgeber, die an die SPS angeschlossen werden.

Ist ein Stromlaufplan mit eingezeichneten echten Gebern in einen Kontaktplan umzuformen, dann ist erst der Anschlussplan der Geber an die SPS zu erstellen. Dabei können die Geber, entsprechend den Sicherheitsüberlegungen, als Schließer oder Öffner an die SPS angeschlossen werden.

Aus Stromlaufplan und Anschlussplan lässt sich dann der Kontaktplan entwickeln.

Der Kontaktplan enthält die Anschlussbezeichnungen der SPS.

Regeln für die Eingangssignale der SPS

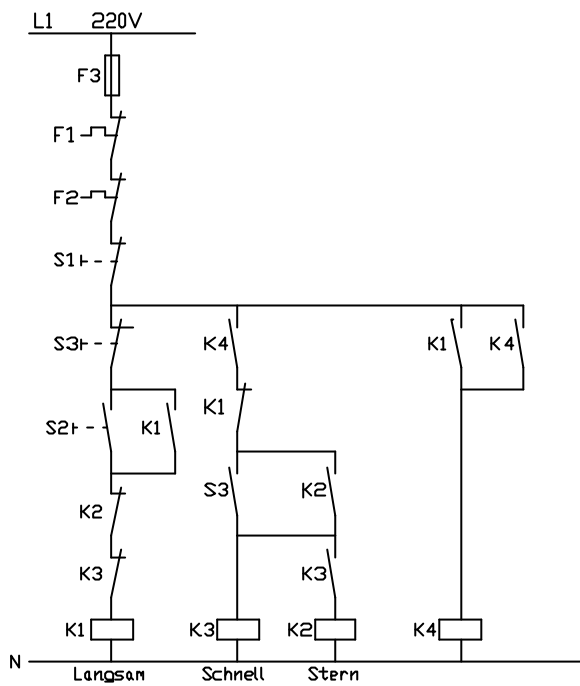
Symbole in Stromlaufplan und Anschlussplan stimmen überein (beides Schließer oder beides Öffner) = Schließer im Kontaktplan

Symbole in Stromlaufplan und Anschlussplan stimmen nicht überein (ein Schließer, ein Öffner) = Öffner im Kontaktplan

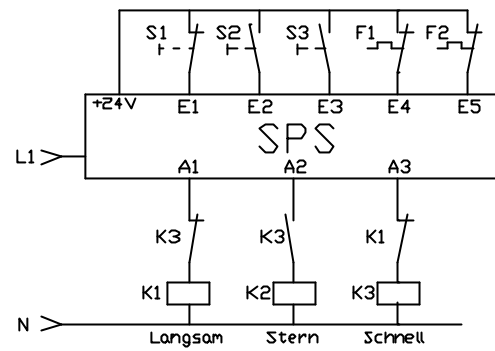
BEISPIEL

Erstellung eines Kontaktplanes als Programm für eine SPS für folgenden Stromlauf- und Anschlussplan.

Stromlaufplan

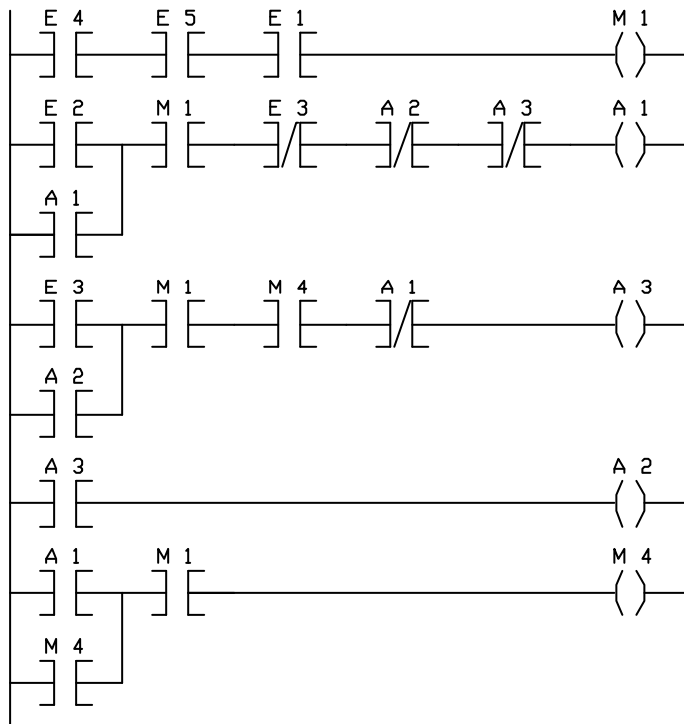


Anschlussplan



LÖSUNG:

Kontaktplan



Funktionsplan

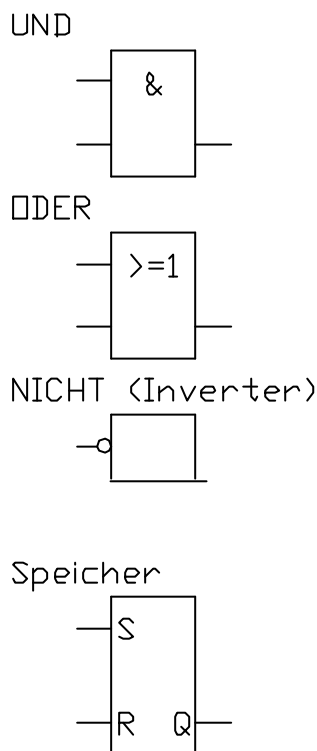
Der Funktionsplan nach DIN 40719 enthält

1. Graphische Symbole für binäre und digitale Funktionen
2. Graphische Symbole für Ablaufsteuerungen.

Die graphischen Symbole für binäre und digitale Funktionen sind Symbole zur Darstellung von Logikfunktionen (Logiksymbole), aus denen der Logikplan erstellt wird.

Die graphischen Symbole für Ablaufsteuerungen enthalten Symbole für Ablaufschritte, Übergänge und Befehlsblöcke

Symbole nach DIN19239:



Bei komplexen Funktionen benutzen die Hersteller unterschiedliche, von den Normen abweichende Symbole.

Die Symbole für komplexere Funktionen und für die Byte- und Wortverarbeitung sind den Handbüchern der SPS-Herstellern zu entnehmen.

Die Kennzeichen und Parameter der Operanden sind beim Funktionsplan die gleichen wie beim Kontaktplan.

Genau wie der Kontaktplan stellt auch der Funktionsplan die logische Verknüpfung zwischen den Ein- und Ausgangssignalen der SPS dar.

Die im Funktionsplan eingetragenen Eingangssignale fragen die Eingangsklemmen der SPS auf 1 bzw. H-Signal ab (anliegende Spannung z.B. 24 V).

Invertierte Eingangssignale fragen die SPS-Eingänge auf 0 bzw. L-Signal ab (anliegende Spannung 0 V).

Beim Entwurf des Logikplanes zur Beschreibung der Lösungsstruktur werden die Leitungen mit Signalnamen versehen. Dabei wird angenommen, dass ein betätigter Geber H-Signal, ein nicht betätigter Geber L-Signal liefert.

Um den Logikplan mit den Signalbezeichnungen der angeschlossenen Geber in den Funktionsplan mit den Eingangsbezeichnungen der SPS umformen zu können, ist wieder der Anschlussplan der Geber an die SPS erforderlich.

Dem Anschlussplan, in den die Geber in nicht betätigtem Zustand eingezeichnet werden, ist zu entnehmen, ob die betätigten Geber L- oder H-Signal liefern.

Falls erforderlich, sind dann die Eingangssignale der SPS im Funktionsplan zu invertieren.

BEISPIEL

Ein Lauftor ist mit einer SPS zu steuern.

Taste S1: Öffnen

Nach Betätigen der Taste öffnet das Tor, bis der Endschalter S4 „Tor ist Auf“ meldet.

Taste S2: Schließen

Nach Drücken der Taste schließt das Tor, bis der Endschalter S5 „Tor ist Zu“ meldet.

Taste S3: Stopp

Die Stopp-Taste schaltet die Torbewegung sofort ab.

Verriegelung

Wird der Lichtstrahl der Lichtschranke S6 unterbrochen, soll die Schließbewegung des Tores stoppen.

Beim Öffnen des Tores ist die Lichtschranke nicht im Eingriff. Trifft die Kontaktleiste S7 auf einen Widerstand, muss die Schließbewegung des Tores stoppen. Das Tor soll dann 2 Sekunden lang wieder öffnen, um ein eingeklemmtes Teil freizugeben.

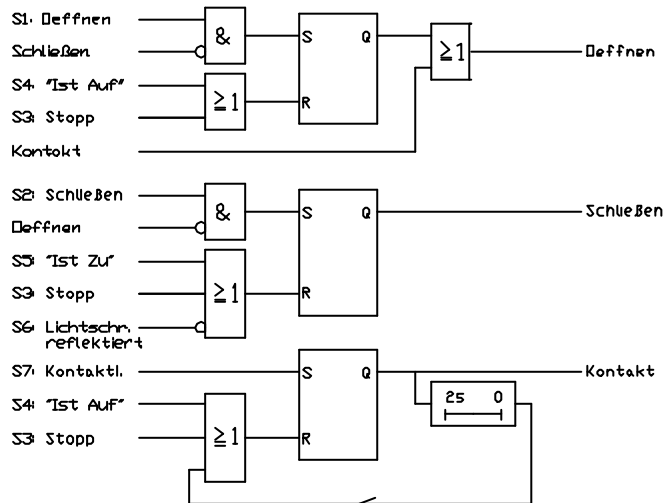
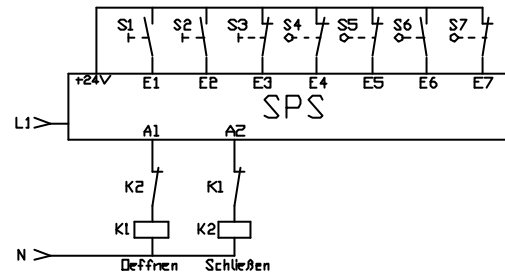
Signallampe H1:

Sobald sich das Tor bewegt, soll die Blinkleuchte eingeschaltet werden.

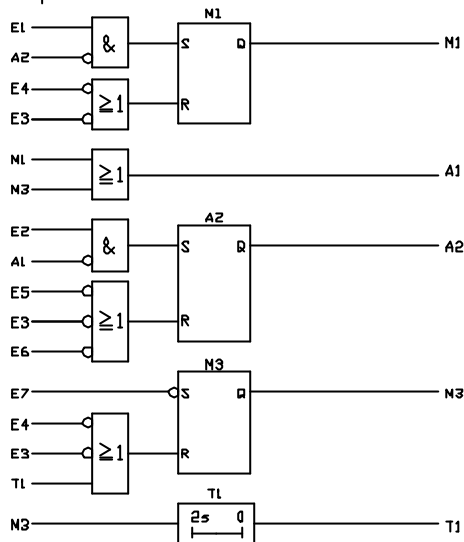
Antriebsmotor M1:

Der Drehstrommotor wird über zwei Wendeschütze geschaltet.

Gesucht: Logikplan, Anschlussplan und der daraus resultierende Funktionsplan.

Logikplan**Anschlussplan**

Funktionsplan



Anweisungsliste

Die Anweisungsliste ist eine prozessnahe Programmiersprache. Sie ähnelt einem Assemblerprogramm. Deshalb wird sie von Sachbearbeitern mit Programmiererfahrung bevorzugt. Aus steuerungstechnischer Sicht stellt die Anweisungsliste eine schaltalgebraische Gleichung dar, deren Glieder untereinander stehen.

Da die Anweisungsliste dem echten Prozessor-Maschinenprogramm am nächsten kommt, bietet sie dem Anwender die meisten Möglichkeiten. Es ist die freizügigste der drei SPS-Programmiersprachen. Die alphanumerische Schreibweise der Anweisungen erlaubt Abkürzungen für Operationen, die graphisch im Kontakt- oder Funktionsplan nicht darstellbar sind.

Die Anzahl der Operanden, die zu einem Verknüpfungsergebnis führen, kann beliebig groß sein, was in der graphischen Darstellungsweise des Kontakt- oder Funktionsplanes aufgrund der Bildschirmbreite nicht möglich ist.

Das Schreiben der Liste geht wesentlich schneller als der Entwurf des Schaltplanes mit Kontakten oder Logiksymbolen auf dem Bildschirm. Bei Kontakt- oder Logikplänen stören besonders die Verknüpfungsbeschränkungen, die bei allen Herstellern unterschiedlich vorgegeben werden.

Deshalb lässt auch nur die Programmiersprache, die den meisten Beschränkungen unterliegt, eine Rückdarstellung in beiden anderen Sprachen zu.

Die freie Anweisungsliste lässt sich daher nur in kleinen Teilen in einen Kontakt- oder Funktionsplan umwandeln.

Sprunganweisungen können in der Regel nur in der Anweisungsliste geschrieben werden.

Aufbau der Anweisungsliste

Eine Steuerungsanweisung setzt sich aus Operationsteil und Operandenteil zusammen. Der Operandenteil besteht aus Kennzeichen plus Parameter. Eine Steuerungsanweisung kann auch nur aus einem Operandenteil bestehen, wie z.B. U (.

Die Anweisungsliste besteht aus einer Folge von Steuerungsanweisungen.

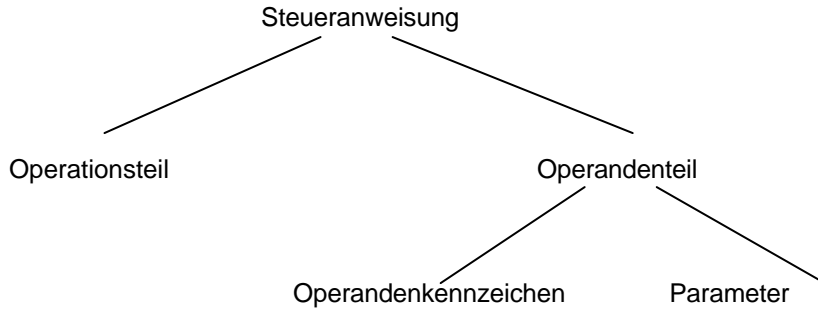
Beispiel einer Anweisungsliste:

```

UE1
UE2
OM3
UNA1
U (
UE3
OA2
)
UNM3
=A2

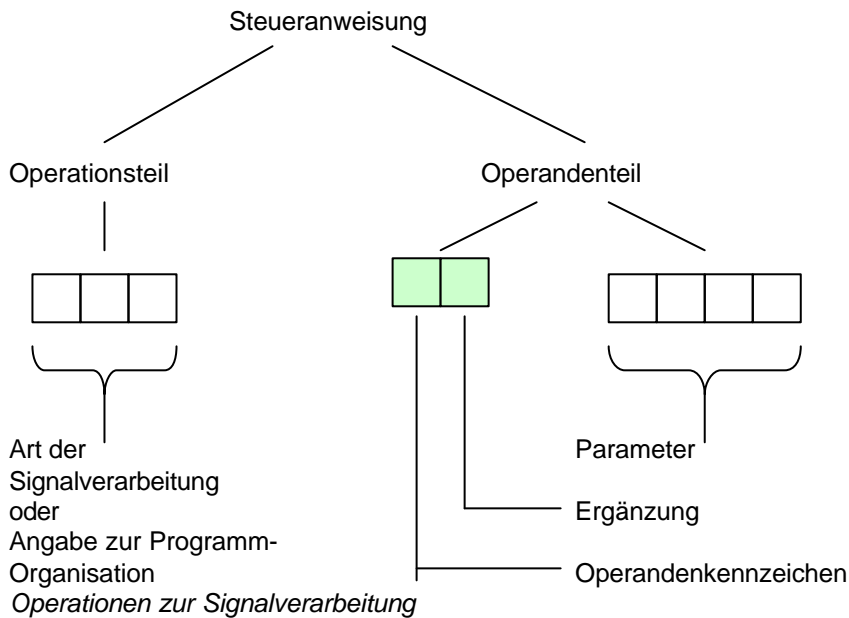
```

einfachste Form einer Steuerungsanweisung



Der Operandenteil kann durch eine Adresse ersetzt werden (z.B. Sprunganweisung SP200)
 Die verschiedenen Teile einer Steuerungsanweisung lassen sich durch ein Lesezeichen trennen.

allgemeine Form der Steuerungsanweisung



Operation	Bedeutung
U	UND
UN	UND NICHT
O	ODER
ON	ODER NICHT
S (oder SL)	Setzen eines Speichers
R (oder RL)	Rücksetzen eines Speichers
ZV	Zählen vorwärts
ZR	Zählen rückwärts
ADD	Addieren
SUB	Subtrahieren
MUL	Multiplizieren
DIV	Dividieren
GR	Vergleich Größer
GRG	Größer gleich
GL	Gleich
KL	Kleiner
KLG	Kleiner gleich
=	Zuweisung einer Verknüpfung zu einem Operanden

Operationen zur Programmorganisation

Operation	Bedeutung
NOP	Nulloperation (es wird keine Operation ausgeführt)
L	Laden (dient der Bereitstellung des ersten Operanden für nachfolgende Operationen)
(Klammer auf, zur Eröffnung eines Verknüpfungszweiges (In Verbindung mit U, UN, O, ON)
)	Klammer zu, zum Abschluss eines Verknüpfungszweiges
SP	Unbedingter Sprung, die Sprungadresse wird angegeben
SPB	Bedingter Sprung, die Sprungadresse wird angegeben
BA	Unbedingter Bausteinaufruf, die Bausteinbezeichnung wird angegeben
BAB	Bedingter Bausteinaufruf, die Bausteinbezeichnung wird angegeben
BE	Baustein Ende
PE	Programm Ende

Operandenkennzeichen

Operand	Bedeutung
E (I)	Eingangssignal
A (O)	Ausgangssignal
M	Merker (interne Verknüpfungsergebnisse, die nicht als Ausgang benötigt werden)
T	Zeitglied
Z	Zähler
K	Konstante

Ergänzungen zum Operandenkennzeichen (Formatangabe)

Ergänzung	Bedeutung
B	Byte (8 Bit) (der Operand ist 1 Byte)
W	Wort (2 Byte) (der Operand ist 1 Wort)
D	Doppelwort (der Operand ist 1 Doppelwort)
A	Analog (der Operand ist ein Analogwert)

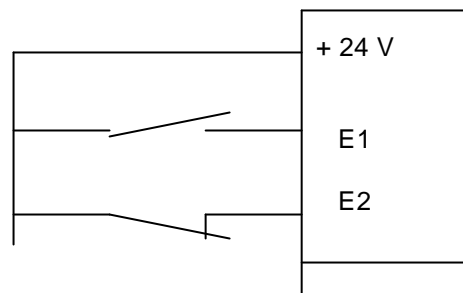
Wird kein Format angegeben, handelt es sich um einen 1-Bit-Operanden.

Eine Anweisung UE1 fragt den Eingang 1 der SPS auf 1- bzw. H-Signal ab.

Eine Anweisung UNE1 fragt den Eingang 1 auf 0- bzw. L-Signal ab. Dem Anschlussplan, der vor der Anweisungsliste zu erstellen ist, ist zu entnehmen, ob die an die SPS angeschlossene Geber im betätigten Zustand L- oder H-Signal liefern.

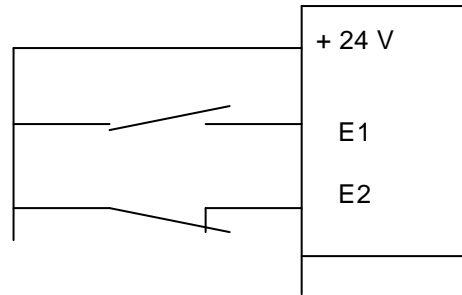
Die Geber sind im nicht betätigten Zustand (Ruhezustand) darzustellen.

Geber Anschluss:



UND-Verknüpfung der betätigten Geber

UE1
UNE2



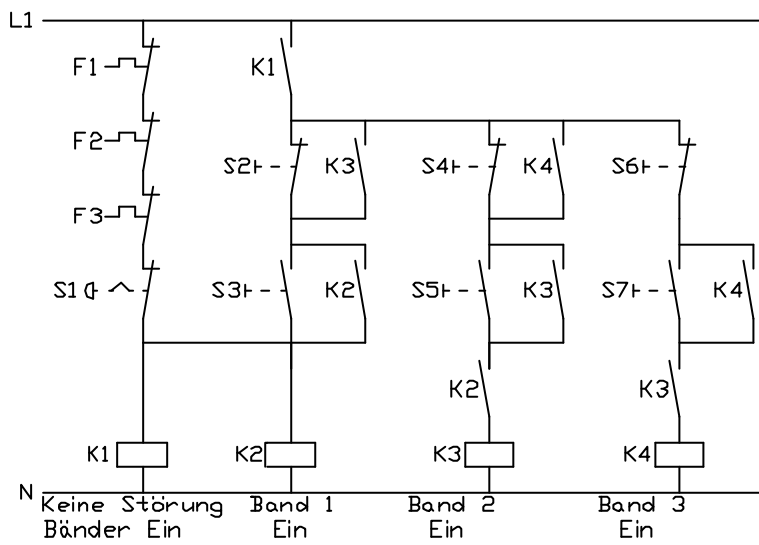
UND-Verknüpfung der nicht betätigten Geber
 UNE1
 UE2

BEISPIEL

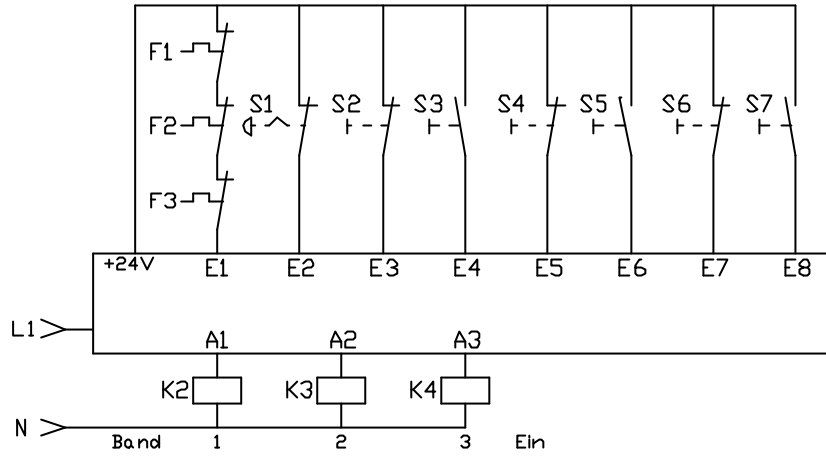
Folgeschaltung für drei Förderbänder

Der gegebene Steuer-Stromlaufplan soll mit einer SPS realisiert werden. Das SPS-Programm ist als Anweisungsliste zu schreiben.

Stromlaufplan



Anschlussplan



gesuchte Anweisungsliste:

UE1		
UE2		
=M2		
UM1	UM1	UM1
U(U(UE7
UE3	UE5	U(
OA2	OA3	UE8
))	OA3
U(U()
UE4	UE6	UA2
OA1	OA2	=A3
))	
=A1	UA1	
	=A2	

