Digitale Steuertechniken : ein Überblick

Autor(en): **Burri, U.**

Objekttyp: **Article**

Zeitschrift: **Bulletin des Schweizerischen Elektrotechnischen Vereins, des Verbandes Schweizerischer Elektrizitätsunternehmen = Bulletin de l'Association Suisse des Electriciens, de l'Association des Entreprises électriques suisses**

Band (Jahr): **75 (1984)**

Heft 7

PDF erstellt am: **21.09.2024**

Persistenter Link: https://doi.org/10.5169/seals-904388

Digitale Steuertechniken: ein Überblick

U. Burri

Für die Realisierung von Steuerungen stehen heute eine ganze Reihe verschiedener Techniken zur Verfügung, von der Relaistechnik bis zum in einer Hochsprache programmierten Computer. An einem einfachen Beispiel wird die Vielfalt der möglichen Techniken und Beschreibungsmethoden gezeigt. Der Aufsatz geht dann auf die Einsatzschwerpunkte der verschiedenen Techniken ein. Nebst den technischen Argumenten für die Wahl einer Technologie werden auch die Konsequenzen für das Realisierungspersonal aufgezeigt.

Pour la réalisation de systèmes asservis, on dispose actuellement d'un grand nombre de technologies, du relayage jusqu'à l'ordinateur programmé en langage évolué. En utilisant un exemple simple, l'auteur montre la multiplicité des techniques et méthodes de description possibles. L'article présente ensuite quelques arguments techniques pour le choix d'une de ces technologies ainsi que les conséquences pour le personnel chargé de réaliser et de maintenir les systèmes.

Adresse des Autors
Dipl. Ing. *Ulrich Burri*, Software-Schule Schweiz, Morgartenstrasse 2, 3014 Bern.

1. Rückblick

Die Steuerung von Industrieanlagen wurde während langer Zeit ausschliesslich mit Hilfe von Relais realisiert (Relaissteuerungen). Vor etwa 20 Jahren wurden die Relais durch Elemente aus der Halbleitertechnik ersetzt. Dazu dienten zuerst diskret aufgebaute Schaltungen mittels Transistoren und Dioden, später integrierte Schaltungen (Integrated Circuits, IC). Bei den Realisierungen mit Hilfe von IC hat sich vor allem die sog. TTL (Transistor-Transistor-Logik) durchgesetzt. Bei TTL-IC werden viele logische Funktionen, die mit Hunderten von Transistoren realisiert werden, auf *einem* IC oder Chip untergebracht. Die gesamte Logik arbeitet fest mit einer Spannung von 5 V und sehr kleinen Strömen. Die Signale können wegen ihrer elektrischen Störbarkeit nicht direkt in die industrielle Anlage geführt werden. Oft ist eine galvanische Trennung notwendig, d.h., dass die zu steuernde Anlage auf einem anderen Potentialniveau liegt als das Steuergerät; auch die verschiedenen Stell- und Messgrössen können unterschiedliches Potential haben. Die Signale müssen zudem je nach Vorschriften beim Kunden oder im Empfängerland auf andere Spannungen (24/48/60/220 V) und elektrisch weniger störempfindliche Stromstärken umgesetzt werden.

Alle genannten Techniken haben gemeinsam, dass echt parallel arbeitende Elemente verwendet werden. Dies bedeutet, dass der Projektierer überall in derselben Philosophie denken und projektieren kann. Eine Ausnahme bildet der Fall, wo die Steuerung den Anlagenzustand speichert, wofür eine Sequenzfolge zu realisieren ist.

Bei der Einführung von Prozessrechnern für die Automatisierung von Industrieanlagen (Process Control) hat es sich gezeigt, dass der Einsatz von Rechnern mit arithmetischem Befehlssatz für viele Probleme zweckmässig ist und auch die Lösung allgemeiner Steuerungsaufgaben zulässt. Es handelt sich dabei um sequentiell nacheinander ausgeführte Befehle, Befehle, die ganze Bytes (8 bit) oder Worte (16 bit) auf einmal ansprechen können und bei diesem Ansprechen sowohl addieren und substrahieren als z.T. auch multiplizieren und dividieren können, ohne dass dies in speziellen Standardprogrammen realisiert werden muss. Der Vorteil ist, dass bei Änderungen der Steuerung (Inbetriebsetzung oder spätere Änderung der Spezifikationen) nicht neu verdrahtet und gelötet, sondern nur das Programm umgeschrieben und neu in den Rechner geladen werden muss. Hierbei wird vom Rechner selbständig ein neues «Listing» angefertigt, d.h., dass die Dokumentation des Steuerschemas automatisch «à jour» gehalten wird.

Diese Lösung hat in der Praxis jedoch zu einem wichtigen Nachteil geführt. Die Art des Denkens in den sequentiellen Programmen ist von der früheren Denkart stark verschieden. Viele Mitarbeiter, deren Qualifikation den Anforderungen solcher Steuerungen angepasst war, haben sich in der neuen Technologie nicht mehr zurechtgefunden. Für Steuerungen, die früher von Berufsleuten realisiert wurden, sind jetzt Ingenieure z.T. mit Hochschulniveau notwendig.

Um diesen Nachteil zu beheben und trotzdem von den Vorteilen der frei programmierten Systeme profitieren zu können, wurden in den frühen siebziger Jahren von vielen Firmen sog. frei programmierbare Steuergeräte (PLC, Programmable Logic Controller) entwickelt. Diese enthalten einen Mikrorechner, der normal wie jeder Rechner sequentiell arbeitet, jedoch gegenüber dem Programmierer eine völlig andersartige Schnittstelle in Form einer bitorientierten Steuerspra-

che zur Verfügung stellt. Es handelt sich um sog. Bit-Rechner mit einer rein zyklischen Programmstruktur. Diese Steuersprachen gestatten dem Programmierer, wieder parallel zu denken und somit Steuerungen in seiner angestammten Denkart zu realisieren.

Im Rahmen dieser PLC sind eine Reihe verschiedener «Sprachen» entwickelt worden. In den USA hat man sich am Anfang dem Betriebselektriker am meisten angenähert, indem auf einem Fernsehbildschirm symbolisch in Relais programmiert wird. Dies hat den Nachteil, dass bei aufwendigen Steuerungen der Überblick erschwert wird, so dass diese Methode im Vergleich zum europäischen Ansatz mit logischen Operatoren und Boolschen Gleichungen auf einfachere Steuerungen beschränkt bleibt. Je nach Hersteller haben sich als Speichermedium für die Programme der konstante Kernspeicher (Core), das batteriegepufferte RAM-Memory oder der EPROM (Erasable Programmable Read-Only Memory) durchgesetzt. Für Industrieanlagen ist der Nachteil der etwas aufwendigeren Programmänderungen bei der Wahl von EPROM-Speichern gegenüber der höheren Langzeitdatensicherung und Verfügbarkeit der Systeme vernachlässigbar. Mit einem einmal geladenen EPROM wird der Rechner zu einem Gerät mit bleibenden Eigenschaften, wie dies bei der reinen Hardwarelösung der Fall ist.

In einer weiteren Entwicklungsphase hat es sich gezeigt, dass bei Steuerungsaufgaben häufig Rechnungen oder alphanumerische Ein-/Ausgaben auf schreibenden Geräten anfallen (Bedienschreibmaschinen oder Video-Bildsichtgeräte). Bei kleineren Anlagen kann man oft aus Kostengründen keinen speziellen Rechner für diese Aufgaben einsetzen. Daher kamen ab etwa 1978 die ersten Geräte auf den Markt, die zwar die Boolsche Steuerungssprache verstehen, daneben aber in der Lage sind, sog. arithmetische Befehle zu verarbeiten. Für diese Lösung hat sich der Ansatz der Boolschen Gleichungen bewährt. Damit ist die Möglichkeit gegeben, Steuerungen in klassischer paralleler Denkweise zu realisieren und gleichzeitig arithmetische Befehle auf demselben Rechner zu verarbeiten. Man spart damit die Kosten für einen zusätzlichen Rechner mit eigenen Prozessein- und -ausgängen, die z.T. schon zum PLC geführt werden. Derartige PLC mit zusätzlichem arithmetischem Befehlssatz sind auf dem Markt eingeführt.

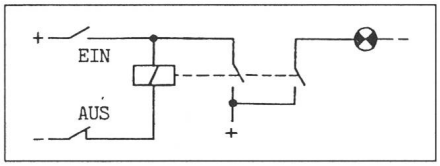


Fig. 1

Im Rahmen der weiteren Verbreitung der Datentechnik und des vermehrten Einsatzes von Rechnern bietet es sich trotz des Nachteils des seriellen Denkens an, anstelle von PLC direkt arithmetische Rechner einzusetzen. Dies gilt um so mehr, als die seit den sechziger Jahren auf dem Markt angebotenen sog. taskorientierten Betriebssysteme je Aufgabe paralleles Denken zulassen. Trotzdem ist der Erstaufwand für einen neuen Anlagetyp grösser als bei der Lösung mit PLC. Der Grund liegt darin, dass durch Informatiker zuerst ein für diese Art von Anlagen spezifisches Softwarepaket erstellt werden muss und erst dieses Softwarepaket eine einfache Realisierung der Anlage durch Berufsleute zulässt. Da die Hardwarekosten für einen solchen Rechner praktisch identisch sind mit den Hardwarekosten für einen PLC, bietet sich ein solches System vor allem für grössere Serien ähnlicher Anlagen an, wo die Erstkosten weniger ins Gewicht fallen. Im allgemeinen ist beim arithmetischen Rechner die Effizienz vergleichbarer Hardware infolge einer schlechteren Ausnützung geringer. Wegen fallender Preise der Hardware ist dieser Nachteil jedoch schon heute nicht mehr relevant. Die schlechtere Ausnützung kommt vor allem daher, dass viele Arbeiten, wie vor allem das Verschieben von einzelnen Bits in eine für die logischen Operationen übereinstimmende Position (Rotationsbefehle), die vom PLC in schnellen Mikroprogrammen bearbeitet werden, per Software (in PL/M geschrieben) durch den Rechner ausgeführt werden.

Im folgenden wird die Vielfalt der möglichen Darstellungsarten und Technologien an einem einfachen Beispiel beschrieben und diskutiert.

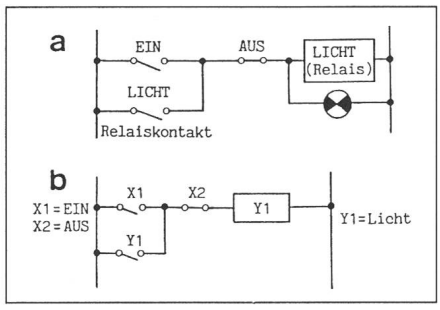


Fig. 2

2. Verschiedene Arten der Beschreibung und Realisierung von Steuerungen

Als Beispiel dient folgender einfacher Steuerungsfall: Das Licht in einem Zimmer ist ein- und auszuschalten. Als Bedienungsorgane sind eine EIN-Taste und eine AUS-Taste (beide ohne Verriegelung) zu verwenden. Wenn die EIN-Taste aktiviert ist, soll die Lampe leuchten. Wenn die AUS-Taste aktiviert wird, soll die Lampe gelöscht werden. Falls EIN und AUS gleichzeitig aktiviert werden, soll die Lampe dunkel sein.

Diese einfache Aufgabe braucht schon relativ viel Text. Daher ist es nicht möglich, komplexe Aufgaben auf diese Weise komplett zu formulieren.

2.1 Direkte Relaistechnik

Ein Techniker, der gewohnt ist, mit Relais umzugehen, würde das Problem wahrscheinlich mittels Figur 1 beschreiben. Diese Beschreibungsform ist sehr lösungsnah, jedoch nur schwer auf andere Lösungsformen als Relaistechnik zu übertragen.

2.2 Logischer Relaisplan (Stromlaufplan)

Die Art der Darstellung von Figur 2a ist schon übersichtlicher. Unter Verwendung von Variablennamen ist sie relativ einfach in andere Realisierungstechniken übertragbar. Sie findet als sog. Stromlaufplan in den meisten konventionellen Steuerungsplänen Verwendung (Fig. 2b).

2.3 Pseudo-Relais-Darstellungen

Derartige Darstellungen (Fig. 3) sind in verschiedenen PLC, vor allem

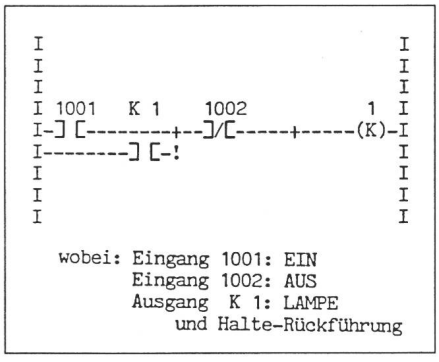


Fig. 3

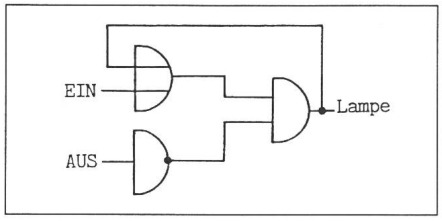


Fig. 4

in den USA verbreitet. Es wird direkt in dieser Darstellung auf einem Bildschirm programmiert. Sie stellt ein Entgegenkommen an die Betriebselektriker dar, die mit der Darstellung von Figur 2 vertraut sind.

2.4 Logische Symbole

Diese Darstellung (Fig. 4) ist in der IC-Technik üblich. Sie wird vor allem von Ingenieuren angewandt, die die Steuerung mit sog. TTL-Logik realisieren.

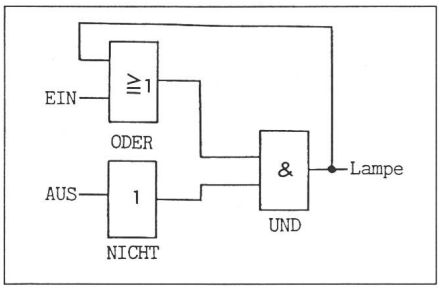


Fig. 5

2.5 Funktionssymbole

Mit Funktionsplänen und Funktionssymbolen nach DIN 40 719, die ein sehr verbreitetes Werkzeug für das Beschreiben von Schaltungen mit digitalen Schaltkreisen sind, erhält man die Darstellung von Figur 5. Sie stellt eine Weiterentwicklung der Darstellung von Figur 4 dar.

2.6 Entscheidungstabelle

In einem ersten Entwurf werden alle möglichen Kombinationen aufgeführt (Tabelle I). Durch Zusammenfassung erhält man die vereinfachte Tabelle II; darin bedeutet –: Y *oder* N.

2.7 Funktionsschema

Diese Darstellung (Fig. 6) stellt ein Entgegenkommen an die Betriebselektriker dar, die mit der Darstellung von Figur 2 vertraut sind. Sie ist daher als europäischer Ansatz der Darstellung von Figur 3 zu verstehen.

Tabelle I

	Regelteil (Rules)							
	R1	R2	R3	R4	R5	R6	R7	R8
Bedienungsteil (Wenn's) (Zustände, Ereignisse)								
EIN-Taste gedrückt	Y	Y	Y	N	N	N	N	Y
AUS-Taste gedrückt	Y	Y	N	N	N	Y	Y	N
Licht AUSgeschaltet	Y	N	N	N	Y	Y	N	Y
Aktionsteil (Dann's)	(Aktionsanzeigeteil)							
Licht EINschalten								X
Licht AUSschalten		X					X	
NICHTS tun	X		X	X	X	X		

Tabelle II

	R1	R2	R3	R4	R5	R6
EIN-Taste gedrückt	Y	Y	N	–	–	N
AUS-Taste gedrückt	Y	N	Y	N	Y	N
Licht AUSgeschaltet	N	Y	N	N	Y	Y
Licht EINschalten		X				
Licht AUSschalten	X		X			
NICHTS tun				X	X	X

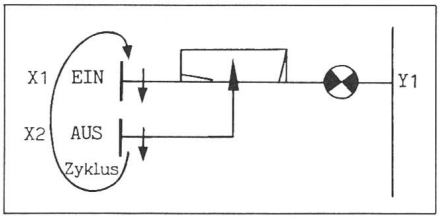


Fig. 6

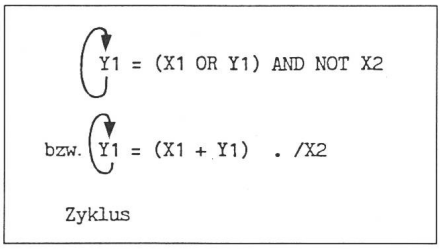


Fig. 7

2.8 Boolsche Gleichungen

Die Auflösung der Darstellung von Figur 6 in die Sprache, die von europäischen PLC verstanden wird, wird je nach Hersteller zu den Gleichungen von Figur 7 oder der Darstellung von 2.9 führen. Zykluszeit bedeutet Auflösung der «Quasi-Gleichzeitigkeit» z. B. 20 bis 100 ms.

2.9 Boolsche Darstellung in sequentieller Form

```
STR X1      ; Store
OR  Y1
AND NOT X2
OUT Y1      ; Output
ANDX1
OR  Y1
ANC X2      ; AND complement
STO Y1      ; Store als Output
```

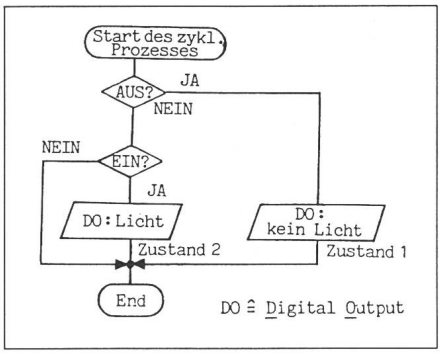


Fig. 8

2.10 Flussdiagrammtechnik in einem zyklisch gestarteten Prozess

Diese Darstellung (Fig. 8) wurde vor allem bei der Realisierung in einem Computer ohne Betriebssystem (zyklisches Arbeiten) oder mit Betriebssystem und zyklisch gestarteten Prozessen verwendet.

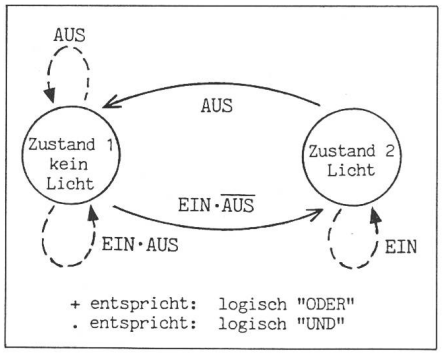


Fig. 9

2.11 Allgemeines Zustandsdiagramm

Diese Darstellung (Fig. 9) findet keine praktische Verwendung und dient lediglich als Ergänzung zur Beschreibung im Klartext.

2.12 State-Event-Technik

Es handelt sich hier (Fig. 10) um ein gutes Werkzeug zur Erarbeitung eines Lösungskonzeptes, das jedoch relativ weit von einer praktischen Realisierung entfernt ist. Die tabellarische Form dient auch mehr als Ergänzung zur klartextlichen Beschreibung.

In dieser Darstellungsart als Statusübergangsdiagramm sind immer Zustand (State), Ereignis (Event) und Zustandsübergang (Aktion) gemeinsam zu betrachten.

Tabelle III

(OLD) Status: Zustand alt Lampe	Ereignisse: Tasten aktiviert EIN	AUS	(NEW) Status: Zustand neu Lampe
kein Licht	–	ja	kein Licht
kein Licht	ja	–	Licht
kein Licht	ja	ja	kein Licht
Licht	–	ja	kein Licht
Licht	ja	–	Licht
Licht	ja	ja	kein Licht

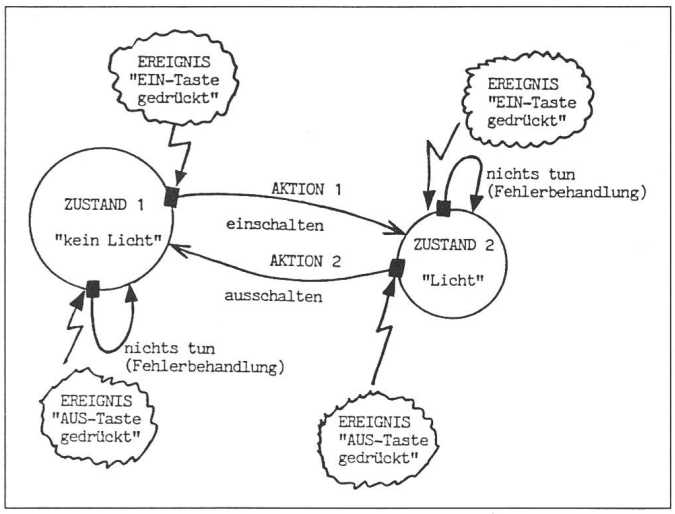
Tabelle III zeigt die tabellarische Form des Statusübergangsdiagramms.

2.13 Höhere Programmiersprache (z.B. Pearl oder Pascal)

Definitionen der Ein- und Ausgänge:

INPUT		AUS	EIN
OUTPUT			Licht
	...	bit 1	bit 0

Fig. 10



Eigentliches Programm:

```
VAR Input,Output   BINARY
                   BOOLEAN
                   (evtl. INTEGER)

WHILE true DO

BEGIN

IF (Input = '00000010'B) OR
   (Input '00000011'B)
   THEN Output:
   = '00000000'B;

IF (Input =  '00000001'B) THEN
   Output:= '00000001'B;

END.
```

Diese Darstellung eignet sich vor allem für relativ einfache Steuerungen, die in einer Hochsprache realisiert werden sollen.

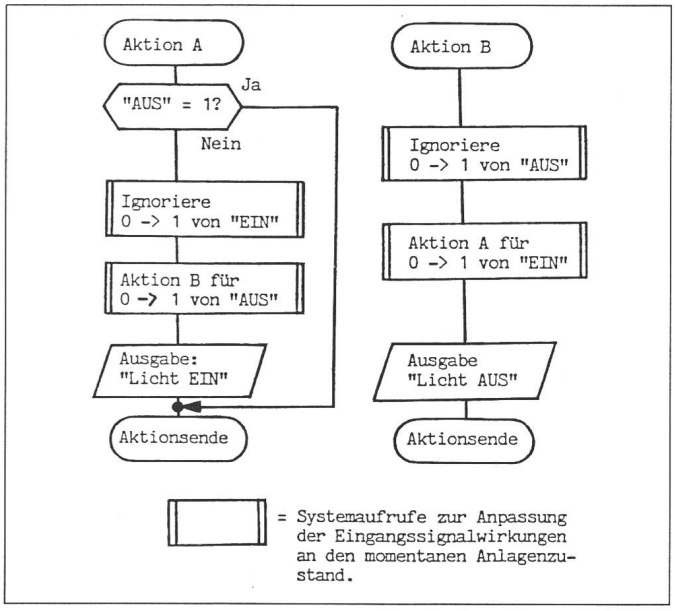
2.14 Macro-11 Assembler

```
While:
        BIT   #^B00000010,@#INPUT
        BEQ   1$
        BIC   #^B00000001,@#OUTPUT
        BR    WHILE
1$:     BIT   #^B00000001,@#INPUT
        BEQ   WHILE
        BIS   #^B00000001,@#OUTPUT
        BR    WHILE
```

2.15 Ereignisorientierte Darstellung (Sequenzmonitor)

Ereignisse haben je nach Anlagenzustand den Start spezifischer Aktionen zur Folge. Sie bilden den Eingang für die Darstellung von Sequenzen. In Figur 11 gilt folgender Initialzustand der Eingangssignalwirkungen:

Fig. 11



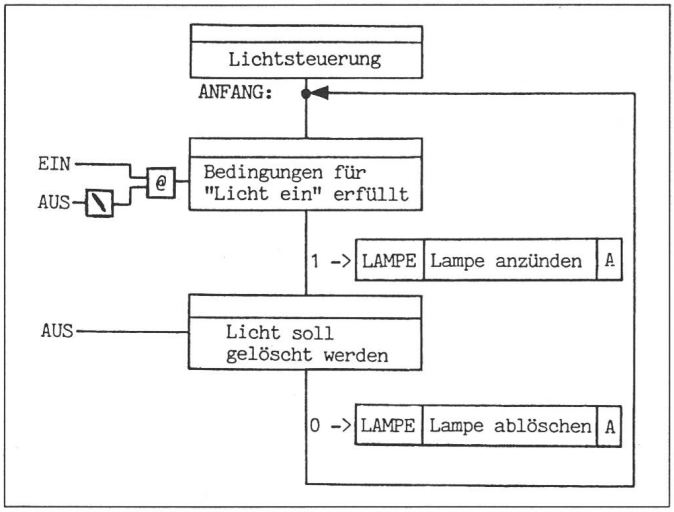


Fig. 12

0 → 1 Flanke des Signales EIN hat Start der Aktion A zur Folge.
0 → 1 Flanke des Signales AUS wird ignoriert.
1 → 0 Flanken beider Signale werden ignoriert.

Mit dieser Darstellung (Fig. 11) können komplexe Steuerungen realisiert werden. Dafür ist der Aufwand, bis überhaupt eine Steuerung realisiert werden kann, grösser. Die aus dieser Darstellung resultierende Problemlösung zeichnet sich vor allem durch grosse Änderungsfreundlichkeit aus.

2.16 Steuerungsplan nach DIN

Diese funktionsorientierte Darstellung ist in Figur 12 zu sehen. Es handelt sich um eine Weiterentwicklung von 2.15, mit der auch Regelungen ausgeführt werden können.

Die alphanumerische Darstellung in der entsprechenden Steuersprache hat folgende Form:

```
ANFANG:  SCHRITT   EIN     @\ AUS
         1 = LAMPE A

         SCHRITT   AUS
         0 = LAMPE A
         -> ANFANG
```

3. Entscheidungskriterien für die verschiedenen Lösungsarten

Im Prinzip können mit den verschiedenen beschriebenen Methoden beliebige Steuerprobleme beschrieben werden. Je nach der für die Realisierung gewählten Technologie drängt sich jedoch die eine oder andere der aufgeführten Beschreibungsarten auf. Die am meisten verbreiteten Technologien für die Realisierung können wie folgt zusammengefasst werden:

- Relaistechnik
- Diodenmatrizen
- IC-Technik (TTL)
- frei programmierbare Steuergeräte, sog. PLC, mit reiner «Bit-Programmierung», d.h. nur mit logischen, Boolschen Verknüpfungen; Bit-Prozessoren
- frei programmierbare Steuergeräte mit zusätzlich arithmetischer Verarbeitung numerischer Grössen, im allgemeinen nur mit Festkommadarstellung auf 8-Bit-Werten pro Assembler-Instruktion
- frei programmierbare arithmetische Rechner mit Byte-(8 bit) oder Wort-(16 bzw. 32 bit)Verarbeitung numerischer Grössen

Je nach Aufgabe kommen im letzten Fall einfache Rechner mit nur Festkommaverarbeitung auf 8, 16 oder 32 bit (Binärdarstellungen, früher z.T. auch BCD-Darstellung) oder komfortablere Rechner mit Gleitkommaverarbeitung auf 32 oder 64 bit in Frage. Dies ist vor allem eine Frage des zu verarbeitenden Datenumfanges und der erforderlichen Rechengeschwindigkeit. Bei diesen numerischen Rechnern stellt sich ferner die Frage nach den optimalen Programmiersprachen:

- Assemblersprachen
- höhere Programmiersprachen wie z.B. Pascal
- eigentliche Steuersprachen, die sich auf die verschiedenen aufgeführten Darstellungsformen (Funktionssymbole, Steuerungspläne, DIN-Normen) abstützen.

Als Kriterien für die Wahl einer Methode müssen verschiedene Gesichtspunkte mitberücksichtigt werden:

3.1 Komplexität und Verknüpfungsgrad

Unter dem Verknüpfungsgrad wird im allgemeinen die Verschachtelungstiefe verstanden. Dies bedeutet, wie oft die Ein- oder Ausgänge im Fall einer Boolschen Darstellung im Mittel pro Gleichung auftreten (Anzahl Gleichungselemente dividiert durch die Anzahl Ein- und Ausgänge).

Im allgemeinen lassen sich heutige Steuerungen in Relaistechnik nur noch bei einer Verschachtelungstiefe unter 3–4 und bei weniger als 30–40 Ein- und Ausgängen rechtfertigen. Bei grösseren Steuerungen ist eine programmierbare Technik nicht nur billiger, sondern auch zuverlässiger (an der genannten oberen Grenze 90–160 Relais, die eine relativ grosse Störungsquelle bedeuten) und übersichtlicher. Zudem wird bei programmierbaren Steuerungen der Preis mit tieferem Verknüpfungsgrad nicht mehr grösser: dieser benötigt nur zusätzliche Speicher. Bei programmierbaren Steuerungen kosten praktisch nur noch die Ein- und Ausgänge.

3.2 Alphanumerische Bediengeräte

Falls zusätzlich alphanumerische Geräte bedient werden sollen, sind Steuerungen in Relaistechnik oder festverdrahtet (TTL, Diodenmatrizen) nicht mehr brauchbar. Ein Steuergerät mit sog. Byte-Verarbeitung kann die alphanumerische Ein- und Ausgabe als Nebenprodukt «ansteuern» und ist damit auf jeden Fall vorzuziehen.

3.3 Qualifikation der Mitarbeiter

Neben den zur Verfügung stehenden Geräten spielt auch die Qualifikation und Erfahrung des Personals, das die Anlage realisieren oder warten soll, eine grosse Rolle. Im allgemeinen muss mit höherer Technik auch höher geschultes Personal eingesetzt werden. Figur 13 zeigt den qualitativen Zusammenhang. Die Wellenlinie in der Mitte der Figur ist ein Grenzgebiet, das trendmässig von beiden Seiten (Pfeile) aufgegriffen wird, so dass sich dort die Grenzen verwischen.

3.4 Wartbarkeit und Verfügbarkeit der Anlage

Die Wart- und Verfügbarkeit einer Anlage ist weitgehend eine Frage des Speichermediums, auf dem die System-Software gespeichert ist. Bei der

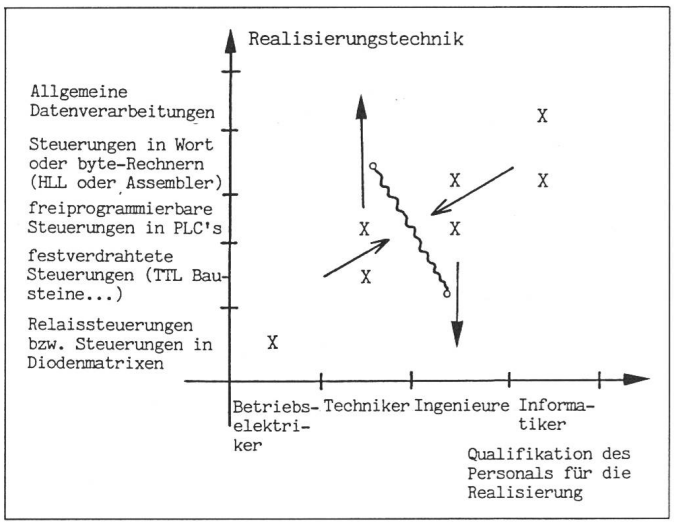


Fig. 13

Realisierung mit einer festverdrahteten Technik (verdrahtungprogrammiert) bzw. einer Speicherung auf einem zur Laufzeit nicht überschreibbaren Speicher (PROM, EPROM, ROM) wird eine einfache Wartung und eine hohe Verfügbarkeit der Anlage erreicht. Es werden keine anfälligen Massenspeicher-Geräte wie Magnetbänder/Plattenspeicher benötigt. Ferner sind keine datenverarbeitungsspezifische Arbeiten, wie z.B. das «Booten» (Laden der Programme in RAM) notwendig. Die Wartung der Anlage kann von nicht in Datenverarbeitung geschultem Personal (keine Operateure) übernommen werden.

3.5 Mathematische Anforderungen an die Anlage

Es ist klar, dass zusätzlich zu den Steuerungsaufgaben zu lösende mathematische Probleme die Wahl der Werkzeuge weitgehend bestimmen. Für Steuerungsgeräte, die z.B. gleichzeitig eine DDC-Regelung realisieren sollen, kommen nur Rechner, die mathematische Verknüpfungen erlauben, in Frage. Hierbei genügt im allgemeinen eine Arithmetik auf 8-Bit-Worten nicht mehr, da die Zeitanforderungen meistens zu hoch sind.