

Roboter selbst steuern

Komplizierte
Technik
einfach
erklärt

Sonderheft Nr.255

Preis DM 18.-

öS 146.-, sfr. 18.-

ELO



Elektronik verstehen, Computer verstehen – mit der ELO fängt der Spaß an moderner Technik erst so richtig an.

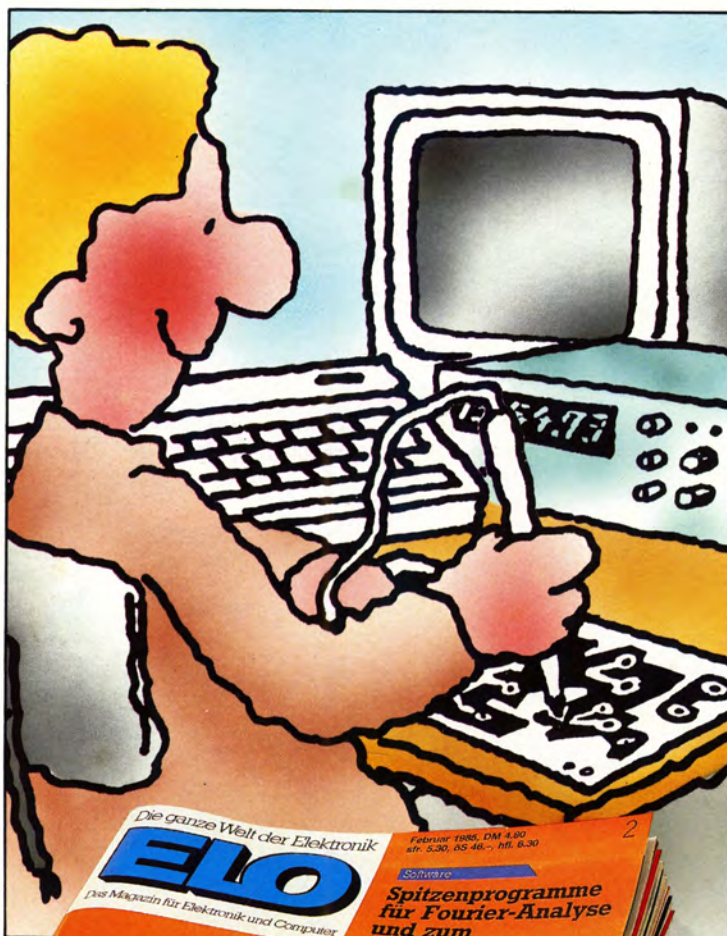
Weil Sie in der ELO leicht zu realisierende **Bauanleitungen** finden, die Ihnen Elektronik praktisch nahe bringen und oft **neue Einsatzbereiche für den Computer** erschließen: Messen, Steuern, Regeln, Erfassen, Auswerten.

Und weil ELO Ihnen helfen wird, Hardware in Ihrem Sinne zu verändern – auch als Anfänger.

Daneben finden Sie leicht verständliche **Beiträge zu Grundlagen der Elektronik**, die sehr schnell dazu beitragen, technische Zusammenhänge zu erkennen und zu nutzen.

Im Magazinteil der ELO und in ELO-Reports wird berichtet, **was mit Elektronik und Computern alles möglich ist.**

In der Forschung, in der Industrie, im Verkehrswesen, im Umweltschutz...



Die Rubrik ELO-Service bringt regelmäßig **Marktübersichten**, die Ihnen wichtige Orientierungshilfe im immer unüberschaubarer werdenden Markt elektronischer Produkte sein werden.

Und natürlich finden Sie **ELO-Testberichte**: Über Schach- und Homecomputer, über Video und HiFi-Geräte, über Meßgeräte und alles, was damit im Zusammenhang steht.

Die ELO verbindet die Elektronik mit dem Computer. Wer endlich richtig einsteigen möchte und den Spaß an moderner Technik neu entdecken will, für den gibt's die ELO für DM 5,50 an allen größeren Zeitschriften-Verkaufsstellen.

Noch besser: Sie machen von unserem vorteilhaften Kennenlern-Angebot Gebrauch.

Eine Kennenlern-Karte finden Sie an der Umschlagklappe.

ELO

Das Magazin für Elektronik und Computer

Lernen am Modell



Brita Eder
Redakteurin der ELO

Sich mittels technischer Kenntnisse einen mustergültigen und starken Helfer nach seinem Bilde zu erschaffen, hat den Menschen schon immer gereizt. Die abenteuerlichsten Konstruktionen entstanden, selbst in der Literatur fanden Homunculi einen Platz. Neben der Faszination ein so kompliziertes Gebilde wie den Menschen „nachzubauen“ sind es schlicht ökonomische Gründe, die dazu führten, bestimmte Tätigkeiten von Maschinen ausführen zu lassen. Der Einsatz von Robotern befreit den Menschen von schweren und gefährlichen Tätigkeiten. Doch im Gegensatz zu einfachen Automaten können Roboter komplizierte und wechselnde Arbeiten erledigen. Sein „Revier“ sind nicht nur die automatische Fabrik und der Schweißapparat, er ersetzt z. B. Schwerstbehinderten einen Teil der verlorenen Beweglichkeit.

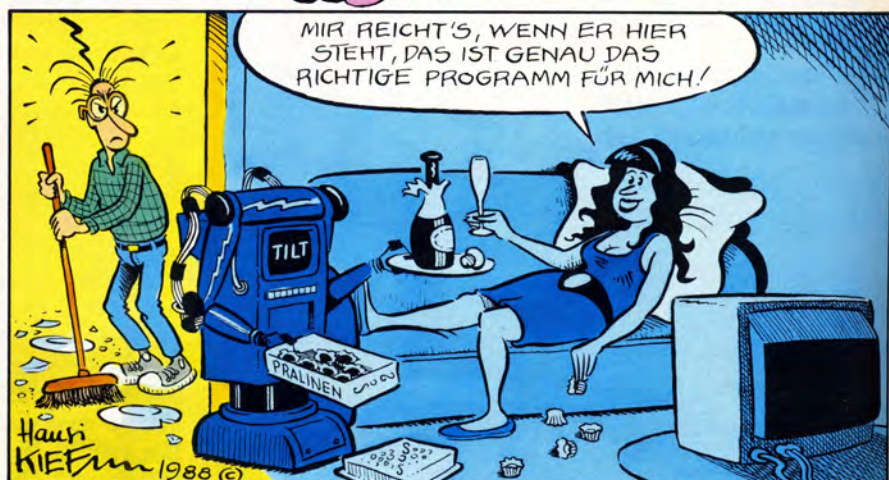
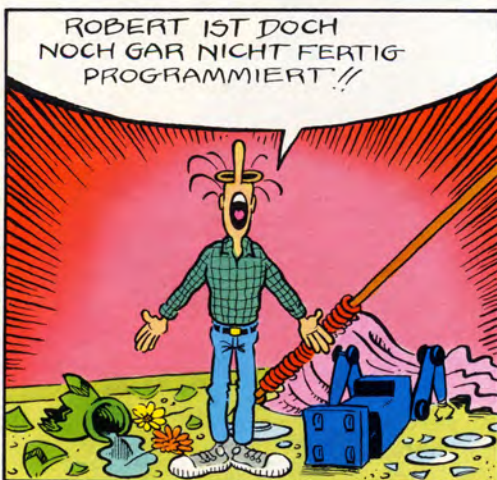
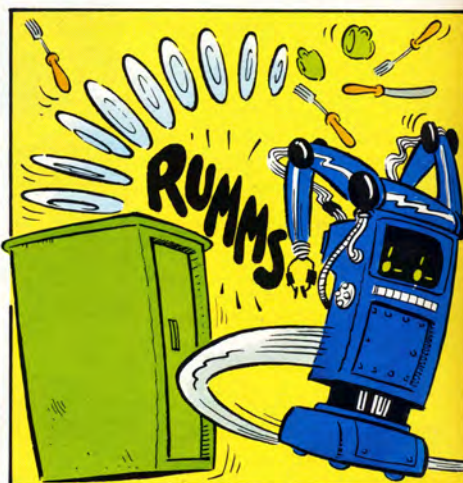
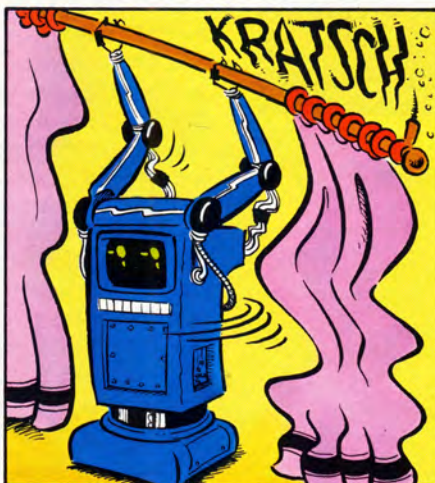
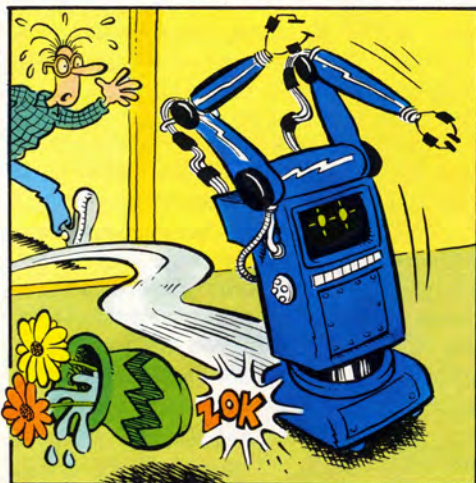
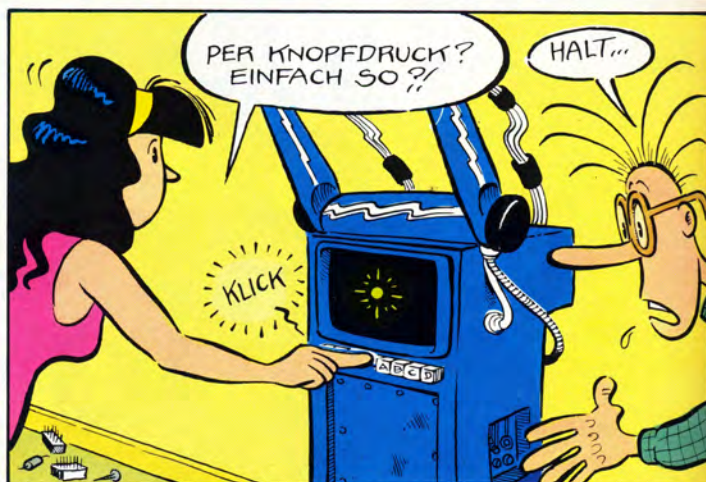
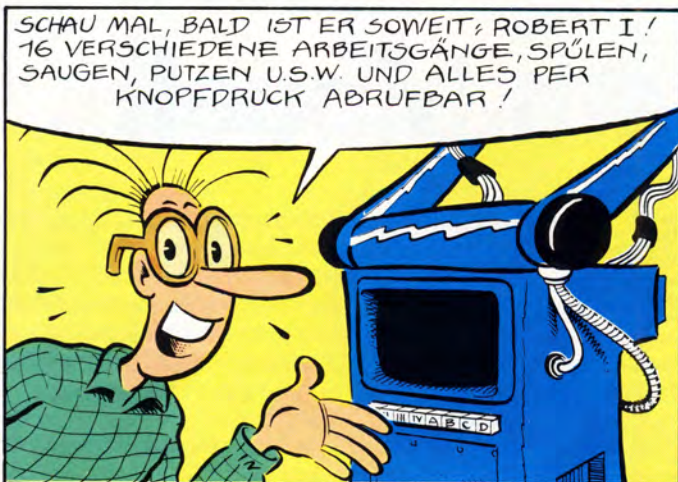
Über den rasant wachsenden Markt besteht kein Zweifel. Wer sich rechtzeitig darauf einstellt, indem er zumindest die grundlegenden Kenntnisse erwirbt, ist im Vorteil.

Natürlich muß der Umgang mit den „Handhabungsautomaten“ erst gelernt werden. Der richtig geplante Einsatz entscheidet über die Renta-

bilität der teuren Maschinen. Doch vor allem sollte jeder, sei er nun Schüler, der den Eintritt in das Berufsleben noch vor sich hat, oder Fertigungsleiter, der irgendwann mit der neuen Technik konfrontiert wird, wissen, wie ein Roboter funktioniert.

Anhand eines Baukastensystems wird Schritt für Schritt in die Funktionsweise und Programmierung einfacher Roboterkonstruktionen eingeführt. Die Programme sind für alle gängigen Home- bis Personalcomputer aufgeführt. Die erworbenen Kenntnisse befähigen dann zum selbständigen Weiterführen bis zum Aufbau einer Fertigungsstraße im Modell.

Brita Eder



Inhalt

Vorwort	3
OLE ELO	4
Inhalt	5
Der eiserne Golem	6
Roboterlehrlinge	10
Anschluß an Sinclair	13
Die stummen Diener	14
Die richtige Verbindung	20
Fischers starker Arm	26
Mehr Ein- und Ausgänge für den Apple II	28
Ampelsteuerung	29
Grün auf Knopfdruck	36
Software-Werkzeug	44
Motoren werden gesteuert	54
Werkzeug-Roboter	56
Der Roboter lernt	64
Die Robotfabrik	72

1987

Franzis-Verlag GmbH, Karlstr. 37-41, 8000 München 2

Sonderheft der Zeitschrift ELO

Redaktion und für den Text verantwortlich: Brita Eder

Herstellung und Layout: Günter Ropertz

Sämtliche Rechte – besonders das Übersetzungsrecht – an Texten und Bildern vorbehalten. Fotomechanische Vervielfältigung nur mit Genehmigung des Verlages. Jeder Nachdruck, auch auszugsweise, und jede Wiedergabe der Abbildungen, auch in verändertem Zustand, sind verboten.

ISSN 0172-2786

Gesamtherstellung Franzis-Druck GmbH, Karlstr. 35, 8000 München 2

ZV-Artikel-Nr. 25503

Druck-Nr. F/ZV/1287/1301/6'

Roboter

Der eiserne Golem

von drei Robotern heute in der Automobilindustrie eingesetzt werden.

Doch sind die Einsatzgebiete natürlich sehr viel weiter gestreut. Roboter findet man als „Handlanger“ in einer Schmiede (Bild 3), als „Packer“ im Versand oder sogar als „Monteur“, der selbständig mehrere Teile zu einem Stück zusammenfügt.

Zur Zeit noch in der Erprobungsphase befinden sich Roboter für die Chipfertigung. Denn für die Bedingungen der Reinraumfertigung erscheinen Menschen trotz aller Vorsicht als „Dreckschleuder“ und sind damit Fehlerquelle.

Roboter sind vielseitig

Betrachtet man sich die bereits erwähnten Film-Roboter oder sieht sich einem 5-Meter-Riesen, der tonnenschwere Eisenteile stapelt gegenüber, fällt es einem mit Sicherheit sehr schwer, eine das alles umfassende Formulierung zu finden.

Selbstverständlich gibt es eine offizielle Definition des VDI über den Roboter. Wie gewohnt in leicht verquastem Deutsch, doch erfreulich kurz und eindeutig:

Industrieroboter sind universell einsetzbare Bewegungsautomaten, mit mehreren Achsen ausgestattet, deren Bewegung hinsichtlich Bewegungsfolge und Wege bzw. Winkeln frei, d.h. ohne mechanischen Eingriff, programmierbar und gegebenenfalls sensorgeführt sind. Sie sind mit Greifern, Werkzeugen oder anderen Fertigungsmitteln ausrüstbar und können Handhabungs- und Fertigungsaufgaben erfüllen.

Mit dieser Begriffsbestimmung ist eine deutliche Abgrenzung z.B. gegenüber einer NC-Fräsmaschine gegeben. Denn die arbeitet zwar automatisch, kann aber nur Fräsen, ist also nicht universell einsetzbar.

Das möglichst leichte Umstellen eines Roboters von einer Tätigkeit zur anderen

Der Roboter hat ein nüchternes Gesicht bekommen. Aus der Science-Fiction-Horror-Maschine ist der zur physischen Arbeit eingeteilte Bruder des Computers geworden. Die Team-Arbeit der beiden Spezialisten macht den Roboter heute zum universell einsetzbaren „Bewegungsautomaten“. Mit der wachsenden Bedeutung der Roboter müssen auch immer mehr Menschen wissen, was ein Roboter ist und kann, hier können Sie es erfahren.

Mit dem Reizwort Roboter verbinden sich Vorstellungen von nützlichen Idioten oder Arbeitsplatzvernichtern – Erleichterung oder Bedrohung. Eine technische Entwicklung, die die alten Maschinenstürmer in manchen Köpfen zu neuer Ehre gelangen läßt. Aber auch Fortschritt, den keine Konkurrenzfähigkeit be-

mühte Fabrik ignorieren darf.

Das mit dem Golem ist höchstens ein dogmatisch gefärbtes Bild. Denn eine automatische Schweißstraße ist nicht im entferntesten mit einem menschenähnlichen Wesen zu vergleichen und ein Präzisionshandhabungsautomat hat manchmal Abmessungen, die unter

20 cm liegen. Daß sich trotzdem Bezeichnungen finden, die dem zwischenmenschlichen Sprachgebrauch entstammen, „Maschinenmensch, Kollege Roboter“ usw., liegt an ihrem Einsatzgebiet. Roboter arbeiten heute dort, wo früher Menschen ihren Arbeitsplatz hatten und ahmen deren Tätigkeit nach.

Eine sichtbare Folge – durch Robotereinsatz dürfte das Montagsauto inzwischen tot sein, denn kein vom abendlichen Kneipenbummel gestreßter Arbeiter wird eine löchrige Schweißnaht ziehen.

Schweißen, Bohren und Transport sind denn auch die bevorzugten Arbeitsgebiete der Automaten, und so ist es kein Wunder, daß zwei

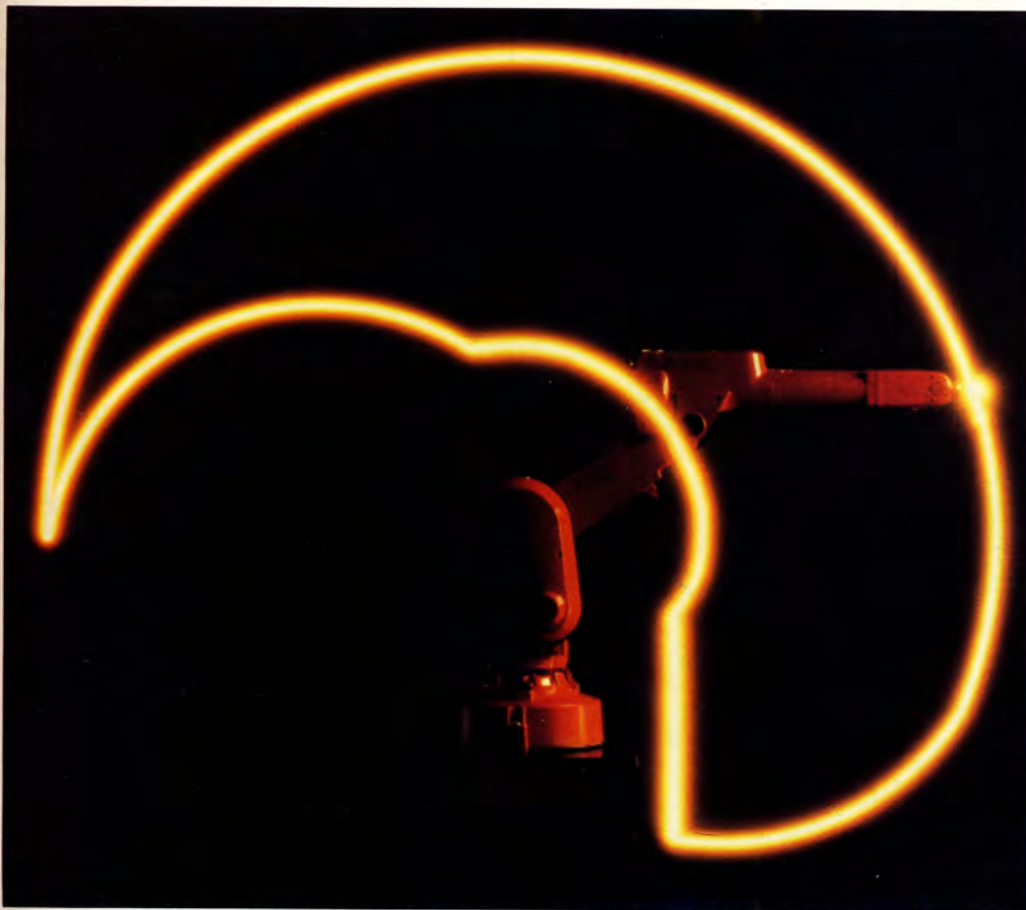


Foto: ASEA Industrie-Roboter

Bild 1: Beweglich und mit einer großen Reichweite ausgestattet ist solch ein Roboterarm.

muß also bereits ein Konstruktionsmerkmal sein, sowohl von seiner Hardware, also dem Bewegungsapparat, wie von der Software, d.i. das unkomplizierte Lehren seiner neuen Arbeit.

Roboter brauchen Arme und Achsen

So vielfältig wie die Einsatzgebiete sind auch die Konstruktionsmerkmale der Robotertypen. Trotzdem erkennt man die meisten auf einen Blick, das sind dann die auf Seite 8 zu sehenden **Knickarm-Roboter**, daß sie am weitesten verbreitet sind. Nach diesem Konstruktionsprinzip arbeiten auch alle uns bekannten Schulungsroboter, denn dieser Robotertyp ist relativ einfach konstruiert und sehr beweglich (**Bild 1**). Er verfügt im allgemeinen über fünf bis sechs Achsen.

damit werden:

- die Grundkonstruktion

- mit senkrechter Drehachse horizontal gedreht,
- der Hauptarm und das darüberliegende Armgelenk haben horizontale Drehachsen

- im oberen, hohlen Arm befindet sich ein Rohr, das verschiebbar ist, daran schließen sich die drehbaren Handgelenke an, zum Schluß folgt das werkzeugspezifische Gelenk, das z. B. die Zangenbewegung der Greifer ermöglicht.

Im allgemeinen wird dieser Robotertyp stationär betrieben. Um seinen recht großen Bewegungsspielraum auszunutzen, muß er fest verankert sein.

Nach den Bewegungsanforderungen richtet sich auch die Platzierung des Roboters, er muß durchaus nicht aufgestellt werden, es ist im Gegenteil oft günstiger, ihn an der Decke zu montieren, so ist er voll beweglich und steht (sich) nicht im Wege.

Der **Röhren-Roboter** kann

kompliziertere Arbeiten ausführen. Seine Hauptdrehachse liegt horizontal und ist etwa im Schwerpunkt der Konstruktion angeordnet (**Bild 3**).

Als Transportgerät wird der **Linear-Roboter** vorwiegend eingesetzt. Sein wichtigstes Konstruktionsmerkmal ist der lange, verschiebbare Horizontalarm und die erhöhte Tragfähigkeit, er wird vorwiegend zum Zulegen von Werkstücken gebraucht.

Die Verwandtschaft mit Kränen sieht man dem **Portal-Roboter** deutlich an, der dementsprechend vorwiegend für Transportarbeiten eingesetzt wird.

Diese geschilderten Robotertypen sind nicht voll beweglich, sie arbeiten stationär, bzw. die zum Transport eingesetzten Roboter benutzen allenfalls Schienen.

Doch vorwiegend für den menschenfernen Einsatz, also in gefährlicher z. B. durch Strahlung verseuchter Um-

gebung, oder in der Raumfahrt werden Roboter verwendet, die sich selbst an ihren jeweiligen Einsatzort bewegen können. Wenn das Gelände planiert ist und Straßen vorhanden sind, sind 3 – 4 Räder geeignet, bei unebenem Gelände helfen 6 Räder oder Ketten.

Wie schon bei anderen technischen Problemen hat man auch beim Konstruieren des Bewegungsapparates versucht, die Natur nachzuahmen. Wurden Menschen zum Vorbild genommen, mißlang das. Die zweibeinige Bewegung eines Körpers, der zudem seinen Schwerpunkt (Mollenfriedhof) relativ weit oben hat, ist mit künstlichen Mitteln zur Zeit nicht befriedigend zu lösen.

Besser funktioniert es, wenn weniger Schwebephasen notwendig werden, also z.B. die Bewegung eines sechsbeinigen Insektes zum Vorbild genommen wird. Da werden jeweils drei Beine bewegt, während die anderen drei Beine stützen.

Roboter müssen auf die Schulbank

Bevor der Roboter seine Arbeit aufnimmt, muß ihm erstmal beigebracht werden, was er zu tun hat. Wäre dieses Lehren so schwierig, daß für jede Änderung des Arbeitsganges ein Programmierer beschäftigt werden muß, wäre eine Grundforderung für den Einsatz von Robotern schon verfehlt. Roboter sollen möglichst flexibel sein, und dazu müssen sie auch von Nichtspezialisten programmiert werden.

Ein Programmiergerät faßt große Gruppen von Anweisungen zu einem einzigen Tastendruck zusammen. Oft werden nur Symbole für den Bewegungsablauf angegeben. So kann man in wenigen Schritten einen Roboterarm dazu veranlassen, sich zunächst um 32° zu drehen, dann den Arm von der Senkrechten aus anzuheben, den Greifer öffnen, den Arm senken, den Greifer schließen,

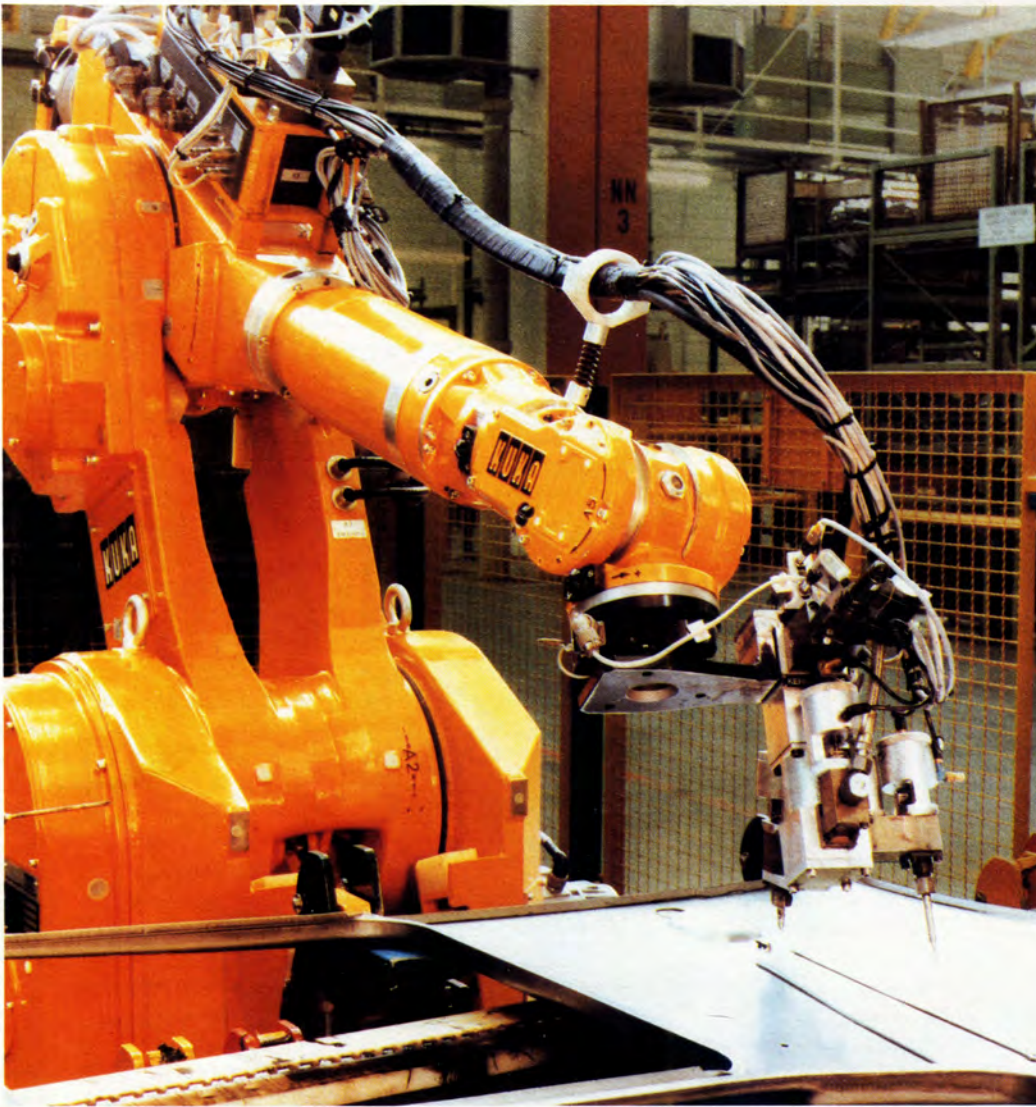


Bild 2: Komplizierte Arbeiten, wie das Kleben und exakte Zusammenfügen von Lkw-Türen beherrscht ein gut instruierter Roboter.

das Werkstück umfassen, anheben, drehen, senken, ablegen. Die Eingabe erfolgt meist im Dialog, so wird die korrekte Reihenfolge immer eingehalten. Außerdem wird der Anwender auf offensichtliche Fehler seiner Programmierung gleich hingewiesen. Bewegungsabläufe, die nicht millimetergenau sein müssen, können sogar mit dem Steuerknüppel eingegeben werden. Mit Recht schreibt ein Roboterhersteller: „Programmieren soll Spaß machen“. Denn nur wenn das Bedienungspersonal vor Ort problemlos zu-rechtkommt, wird der Roboter optimal genutzt.

Fixpunkte helfen bei der Programmierung

Das Programmieren komplizierter Bewegungen wird erheblich vereinfacht, wenn unterschiedliche Koordinatensysteme verwendet werden. Denn Drehbewegungen lassen sich leichter in Zylinderkoordinaten angeben. In allen Arbeitsgängen gibt es Fixpunkte, die immer wieder angefahren werden. Wenn man einige dieser Punkte dauerhaft speichern und von jeder Position aus ohne weitere Angaben machen zu müssen, ansteuern kann, spart man Programmierzeit. Die meisten Industrieroboter kennen eine Reihe solcher, TCP genannten Arbeitspunkte.

Eine andere Programmiermethode nimmt sich wiederum den Menschen zum Vorbild. Das Nachahmen menschlicher Tätigkeit wird am Beispiel von Farbspritzarbeiten besonders deutlich. Der Mensch, der diese Arbeit bisher machte, führt einen Roboterarm so über das zu bearbeitende Teil, wie er selber es tausendmal gemacht hat. Der Roboter, bzw. der ihn steuernde Computer merkt sich den Bewegungsablauf und wird ihn in Zukunft immer wieder exakt ausführen.

Das Nachführen (teach-in) zeichnet sich zwar durch ei-

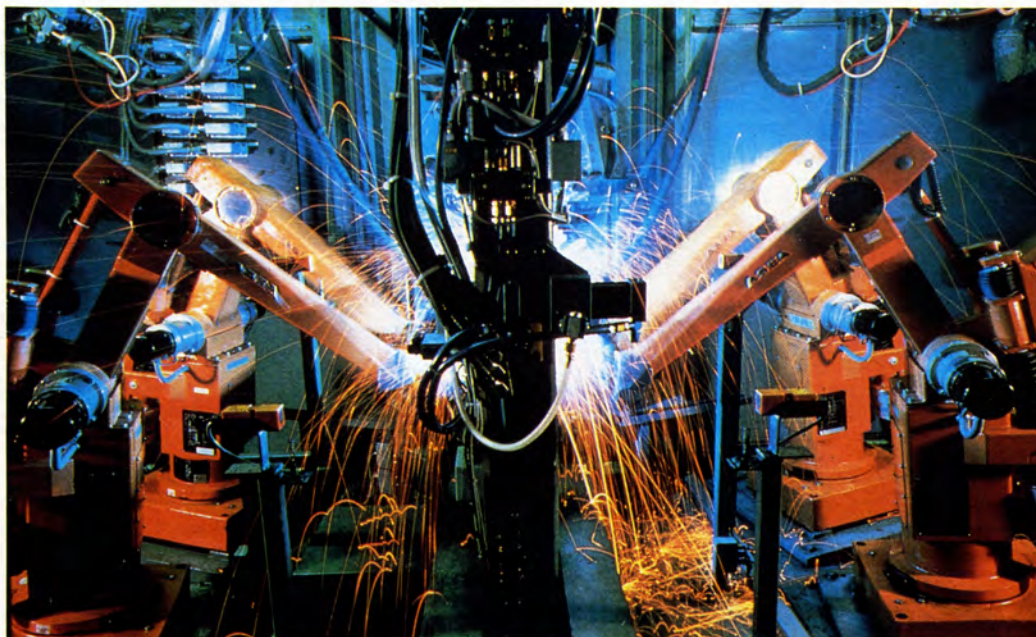


Bild 3: Ein Roboter braucht keine Handschuhe um Schweißarbeiten auszuführen.

Foto: kuka

Foto: ASEA Industrieroboter

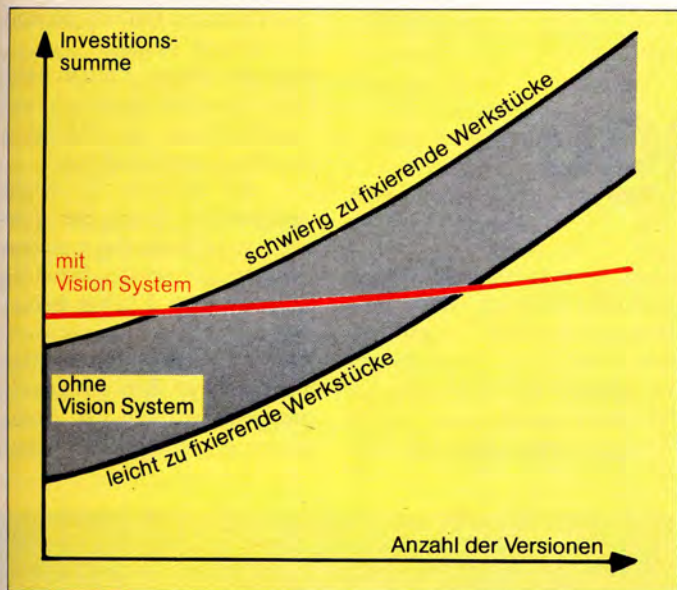


Bild 4: Je schwieriger es ist, ein Werkstück zu erkennen und je mehr Versionen davon existieren, desto eher lohnt es sich, einen „sehenden“ Roboter einzusetzen.

ne sehr einfache Handhabung aus, stellt jedoch eine Primitivversion der Programmierung dar. Und an diesem Beispiel läßt sich auch der weniger positive Aspekt des automatischen Nachahmens zeigen. Angenommen, das Teil liegt in einer etwas verschobenen Position, oder die Farbzusammensetzung ist leicht verändert (flüssiger); der Mensch an diesem Arbeitsplatz konnte darauf reagieren, der Roboter bemerkt nicht, daß das Teil nun nur unvollkommen gespritzt ist. Das führt uns zum nächsten Punkt; wieviel muß ein Roboter von seiner Umwelt erkennen können?

Mit Sensor und Kamera fühlen und sehen Roboter

Störanfällig und umständlich ist die Art, wie die meisten Roboter auf Teile zugreifen. Die benötigten Teile müssen immer an einem genau festgelegten Ort zu finden sein. Durch Toleranzen des Roboters oder durch Erschütterungen ist es möglich, daß ein Fehlgriff erfolgt. Vor allem bei Kleinteilen, die in größeren Stückzahlen aufliegen (Bild 4). Um trotzdem ohne mensch-

lichen Zuarbeiter auszukommen, wurden Kameras und Video-Digitalisierer in Roboter eingebaut. Eine Schwarzweiß-TV-Kamera erzeugt ein Bild, dessen Ausgangssignal von einem Computer verarbeitet wird. Nicht nur die Umrisse, auch Graustufen werden so erfaßt, daß normale Schwankungen der Beleuchtung keinen Einfluß haben. Die „Sehfähigkeit“ der Roboter ist aber noch sehr eingeschränkt, die zum Vergleich herangezogene Information umfaßt im Prinzip nur die Konturen des Objektes.

Siemens hat jedoch einen Graubildsensor entwickelt, mit dem 68 Teile trotz variierender Beleuchtung identifiziert werden können, auch wenn sie übereinander liegen. Der Sensor ermöglicht sogar ein Erkennen der Oberflächenstrukturen.

Um das zu realisieren, müssen auch die feineren Abstufungen der Grauwerte gespeichert und verglichen werden. Die Identifizierung dauert ca. 0,5 bis 2,5 Sekunden. Das ist für einen industriellen Einsatz allerdings noch viel zu langsam.

Häufig sind Drucksensoren in den Greiferhänden der Roboter integriert. Emp-

findliche Teile werden mit der zum Halten notwendigen Kraft umfaßt, ohne zerquetscht zu werden. Da sind auch die buchstäblichen rohen Eier kein Problem mehr. Interessant und von steigender Bedeutung ist der Einsatz von Robotern in der Endkontrolle. Mit Röntgenaugen kann ein fahrbarer Roboter in Rohre krabbeln, um die Schweißnähte zu überprüfen. Mittels Ultraschallsensoren kann die Oberfläche von Werkstücken kontrolliert werden.

Werkzeuge machen Roboter zu Facharbeitern

Die bekannteste Ausführung der „Roboterhand“ ist der vielseitig verwendbare Greifer. Seine Zangen können direkt als Werkzeug eingesetzt werden, bzw. solches halten. Für besondere Arbeiten wird das Endstück des Roboterarmes gegen ein Spezialwerkzeug ausgetauscht. Zum Punktschweißen ist die gesamte Schweißanlage im Roboter vorhanden, je nach Aufbau der Mechanik sind die Roboter besser als ein Mensch in der Lage, auch schwer zugängliche Teile genau zu verschweißen.

Beim Vernieten werden die Vorteile des Roboters besonders deutlich. Denn er kann mehrere Arbeitsgänge in einem Zug durchführen. Der Roboter muß nicht nach den Einzelteilen greifen, denn die werden über sein im Arm integriertes Magazin zugeführt. Bohren, Nieten einpassen und mit Hydraulikschlag zusammenpressen dauert etwa drei Sekunden. Besonders feines und genaues Schneiden ist mit einem Laser in Roboterhand möglich. Ein Elektromagnet statt der Greifer kann Metallteile ganz unabhängig von ihrer Größe und Form transportieren.

Als Schlußfolgerung aus den Informationen über den derzeitigen Stand der Robotertechnik läßt sich einiges Positive ablesen:

Arbeit unter schwierigen und gesundheitsschädlichen Bedingungen wie Staub, Lärm, Hitze und Strahlen, sollte und kann in Zukunft ruhig dem Roboter überlassen werden.

Statt vieler spezialisierter großer Maschinen, die beim Umstellen der Produktion evtl. sogar überflüssig werden, können Roboter viele Aufgaben übernehmen.

Ein äußerst publikumswirksamer Anwendungsfall ist der Einsatz des Roboters im Haushalt. Doch obwohl er immer wieder propagiert wurde – eine Reihe von Prototypen existieren – konnte er sich nicht durchsetzen. Offensichtlich verlangt der Haushalt wesentlich größere Flexibilität als der durchorganisierte und wohlgeordnete Produktionsbetrieb.

Diese Flexibilität und die benötigten Sensoren machen den Haushaltsroboter so teuer, daß Hausmädchen und Butler immer noch kostengünstiger und zudem repräsentativer sind. Die zum erschwinglichen Preis angebotenen Roboter können nur wenige Spezialaufgaben wie Servieren und Staubsaugen, d. h. ihr „Nutzen“ liegt eher im Spaß an der Sache.

Menschenleere Fabrihallen werden aller Wahrscheinlichkeit nach trotzdem eine Ausnahme bleiben. Der mit zahlreichen Sinnen und einem beweglichen Verstand ausgestattete Mensch läßt sich doch nur in geringen Teilen seiner Funktion nachahmen und wird nie ganz ersetzt werden können. Es werden durch den Roboter sogar einige neue Arbeitsplätze geschaffen. In der Roboterfertigung liegen Westeuropa und USA gemeinsam an der Spitze.

Brita Eder

Buch zum Thema:

Künstliche Intelligenz und Robotik

von Thomas Winzer, DM 38,-, Franzis-Verlag, München, ISBN 3-7723-8381-5

Roboter in der Facharbeiterausbildung

Roboterlehrlinge

Roboter werden immer flexibler. Ihr Einsatz wird in absehbarer Zeit für viele Firmen zu den entsprechenden Investitionen führen. Mindestens genauso wichtig ist die Investition in die „Bediener“-Ausbildung. Eine Berufsschule in Kempten macht den Anfang.

Man könnte es sich recht einfach machen und den Schulen einfach sagen, na nun macht mal. Doch damit ist noch lange nicht klar, in welchen Berufen und damit Ausbildungswegen der Ro-

boter Einzug halten soll und in welchem Maße.

Erst seit einem halben Jahr läuft der Schulversuch zur „Ausbildung an flexiblen Handhabungssystemen“, wie der Roboter so schön und treffend heißt. Der Versuch wird vom Bayerischen Staatsministerium für Unterricht und Kultus gefördert.

Vier Ausbildungsplätze wurden mit der notwendigen Ausrüstung, also Computer einschließlich Zube-



hör und Übungsrobotern, versehen (**Bild 1**). Damit soll ausprobiert werden, wie der Roboter in den Lehrplan der Ausbildung zum Industriemechaniker eingefügt werden kann.

Diese Berufsgruppe eignet sich deshalb für ein derartiges Pilotprojekt, weil hier die bisherigen Berufe wie Mechaniker, Feinmechaniker, Maschinen- und Betriebsschlosser zusammengefaßt wurden. Aus dieser Gruppe dürften deshalb die

zukünftigen „Bediener“ kommen. Aber auch die Gruppe der Zerspanungsmechaniker wird in Zukunft verstärkt mit Robotern zu tun haben.

Zum Aufgabenbereich eines Industriemechanikers gehört „das Einrichten, Inbetriebnehmen, Steuern,

Bild 1:
Die Zukunft beginnt in der Schule. Ein Computer/Roboter-Arbeitsplatz gehört dazu.



Überwachen und Warten von automatisierten Produktionsanlagen und Fertigungssystemen in Betrieben mit spangebender, spanloser oder montierender Fertigung.“

Das sind nun auch die klassischen Einsatzgebiete des Roboters, dessen Vorteile, wie Einsatz unter harten Bedingungen und große Flexibilität, hier zum Tragen kommen.

Zudem sind Roboter eine Weiterentwicklung der NC-Maschinen. Die haben in den letzten Jahren in der industriellen Produktion ihren Siegeszug angetreten. Eine ähnliche Entwicklung könnte dem Roboter bevorstehen.

Schwerpunkte für die neue Ausbildung am Roboter

Neben den Grundlagen, also Aufbau eines Roboters, Einsatzmöglichkeiten, Programmierung und Simulation, ist für die Ausbildung des Industriemechanikers das künftige Arbeitsgebiet von besonderer Bedeutung. Im beruflichen Alltag wird er weniger damit beauftragt sein, ein Robotersystem komplett zu installieren, als vielmehr ein vorhandenes zu „pflegen“. Das heißt, die Mechanik und Elektronik zu überwachen und notfalls zu reparieren, sowie Anpassung und Optimierung der Software vorzunehmen. Ebenso wie beim Einsatz in der Industrie ist auch im

Schulbetrieb als erstes die „Programmiersprache“, das Teach-in, erprobt worden. Die anderen Methoden wie PTP- oder CP-Programmierung erfordern Erfahrung und größere Kenntnisse.

Teach-in bedeutet jedoch nicht, daß man wie bei einem Kleinkind die zum Essen mit dem Löffel erforderlichen Handgriffe beibringt, indem man seine Hand führt. Es bedeutet vielmehr, daß der Greiferarm im Zusammenspiel mit den Bewegungsmöglichkeiten Schritt für Schritt an die geforderten Punkte „herangetastet“ wird. So können z. B. Schweißpunkte mit hoher Präzision festgelegt und die erforderlichen Bewegungsschritte im Computer registriert werden. Die dort gespeicherten Arbeitspunkte lassen sich dann beliebig oft wieder anfahren.

Spielroboter sind zu wirklichkeitsfremd

Um eine möglichst große Nähe des Schulversuches zur Arbeitswelt zu erreichen, wurde ein Roboter ausgewählt, der in allen wichtigen Funktionen einem Industrieroboter entspricht, d. h. 5 bis 6 Achsen besitzt und frei programmierbar ist (**Bild 2**).

Obwohl die Roboter erst vor kurzem in die Kemptener Berufsschule einzogen, ließen sich schon eine Reihe von Aufgabenstellungen mit ihnen erproben.

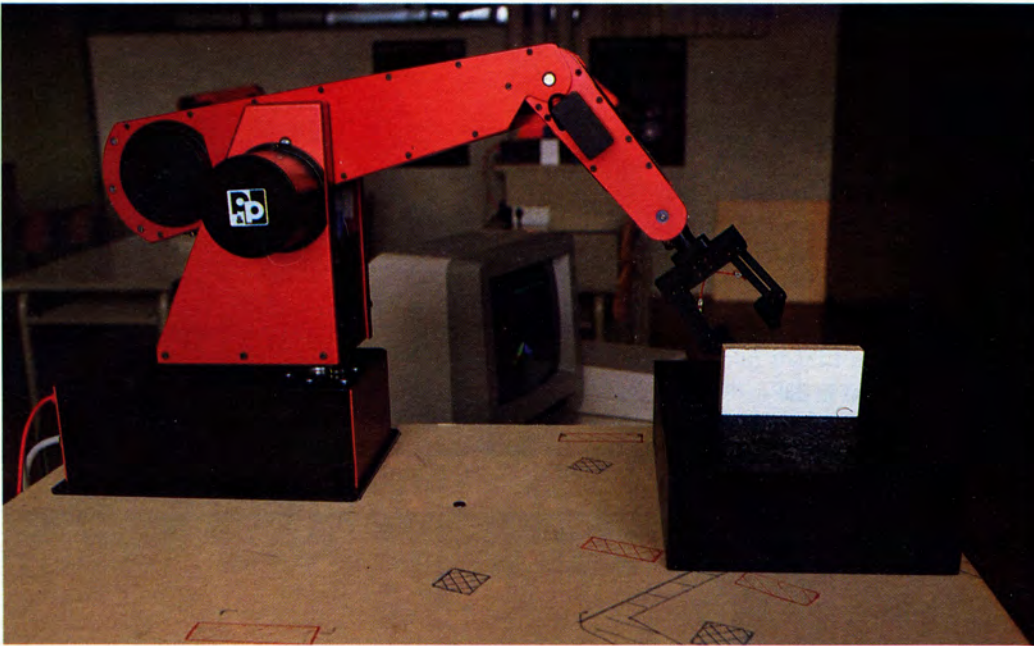


Bild 2:
Ein für Übungsaufgaben
konzipierter kleinerer Bruder
der großen Roboter.
Mechanisch stabil muß er
schon sein, sonst hat er ein
zu kurzes Schulleben.

Versuche, die noch näher an den tatsächlichen Roboter-einsatz angelehnt sind, werden zur Zeit noch entwickelt, doch schon jetzt läßt sich deutlich sehen, daß der Einsatz des Roboters nicht nur in der Industrie, sondern auch in der Ausbildung große Chancen eröffnet. Die mathematische Behandlung der Roboterkinematik wurde in diesem Fall bewußt ausgeklammert; vielmehr soll an gezielten Aufgabenstellungen die Bewegungsmöglichkeit und der Bewegungsspielraum des Roboters „erfahren“ werden. Es gibt eine mathematische Theorie der Roboterbewegungen, die sicherlich nützlich und genau ist, aber dem Schüler wenig Spielraum läßt. Und zwar Spielraum im wörtlichen Sinn. Erst wenn man sieht, wie der Roboter ein Werkzeug faßt und transportiert, entwickelt sich ein Gespür dafür, wie dieser Bewegungsablauf eleganter anzupacken wäre, oder wie kompliziertere Aufgaben gelöst werden könnten.

Fotos: G. Lang

Der Rob2 unterstützt das visuelle und schnell umsetzbare Lernen durch eine Programmumgebung, deren Handhabung nur wenig theoretischen Background verlangt. Der gerade erreichte Stand der Achsen und Arme läßt sich am Bildschirm gut verfolgen. Noch näher am Industrieroboter ist der neue Rob3, der ähnlich wie seine großen Brüder über ein Bedienpanel verfügt. An die 8 digitalen und 2 analogen Ein- und Ausgänge lassen sich Sensoren anschließen, die dem Roboter ein Bild seiner Umwelt vermitteln.

OStR Günter Lang
Brita Eder

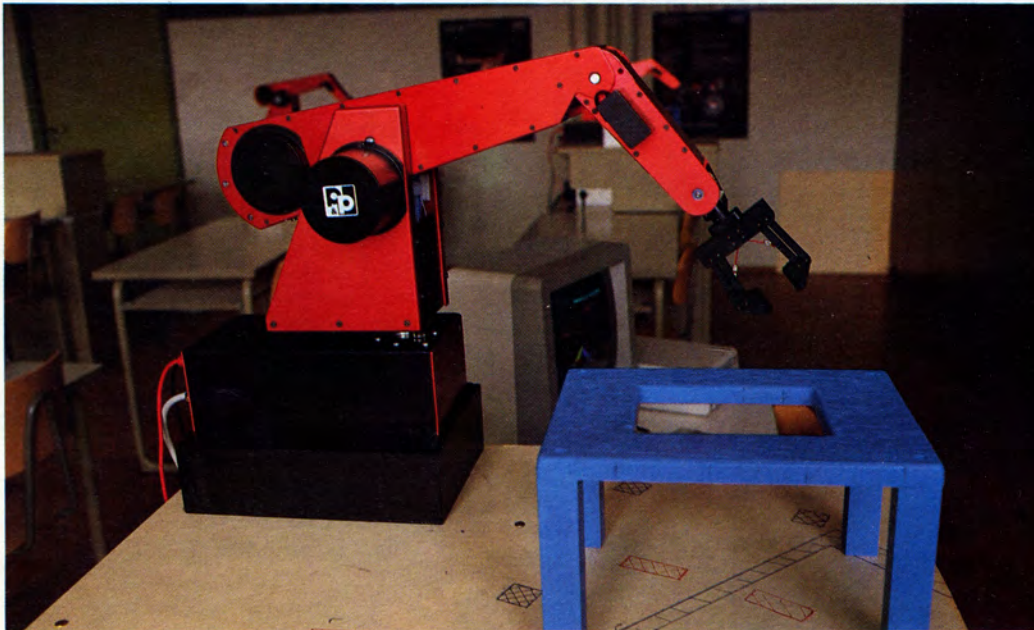


Bild 3: Das Beladen senkrechter und waagerechter Ladeluken läßt sich mit diesem Tisch üben.

In der Unterrichtsmethodik kam es vor allem darauf an, bestimmte Begriffe aus der Robotertechnik am „lebenden“ Objekt zu erläutern. Zum Beispiel wird anhand eines speziell entwickelten Tisches die Beladung von waagrecht oder senkrecht angeordneten Ladeluken simuliert (Bild 3).

In Bild 4 ist eine erheblich schwierigere Aufgabe gezeigt. Die unterschiedlichen Bohr- und Werkzeugdurchmesser erfordern eine recht hohe Positioniergenauigkeit.

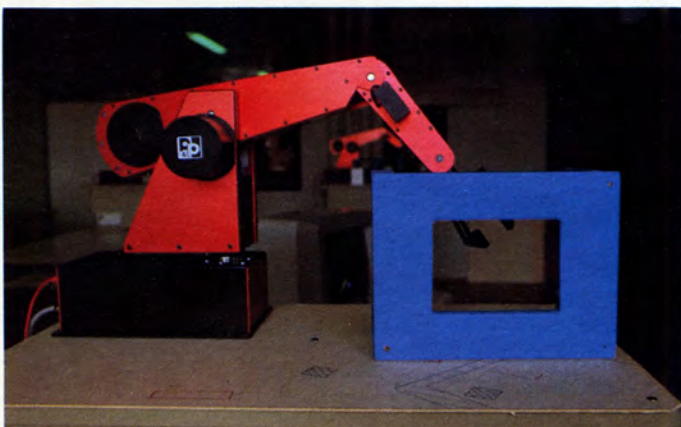


Bild 4: Feines Positionieren wird verlangt, wenn man die Werkzeuge richtig unterbringen will.

Roboter an Sinclair Computern

Wenn ein Bericht über einen interessanten Einsatz von Computern veröffentlicht wird, freuen sich die Leser, die gerade denjenigen Computer ihr eigen nennen, auf den das Programmlisting paßt. Die Besitzer anderer Computer schauen eher unglücklich drein. Gewiß, das Programm ist in BASIC geschrieben, aber was *genau* bewirkt die Funktion XYZ auf dem fremden Computer? Ein Anleitungsbuch steht ja nicht zur Verfügung, wo man nachschlagen könnte und damit eine Übersetzung versuchen könnte.

Schlimmer gar noch, wenn auch Schaltungen für den Computer angegeben werden. Eine Anpassung wird in den meisten nur schwer möglich sein. So geschehen bei der Serie „Einführung in die Robotik“. Zwar werden dort die Besitzer von C64/C128 und CPCs versorgt – in einigen Punkten wird eine Hilfe auch für Acorn-, Apple-, IBM- und NDR-Klein-Benutzer geboten –, es fehlt jedoch etwas für die Sinclair-Freunde. Mit der hier angebotenen Information soll das Versäumte nachgeholt werden.

Im ersten Schritt müssen die Sinclair-Computer mit etwas nachgerüstet werden, was bei den oben genannten Computern vorausgesetzt werden konnte: ein frei programmierbarer Parallelanschluß jeweils für Ein- und Ausgabe. Wir greifen hier auf Bewährtes zurück. In der Funkschau (ab Heft 4/84) wurde die „Daten-Drehscheibe“, eine universelle Ein- und Ausgabeschnittstelle für Experimente vorgestellt. Wer Mühe hat, diese Hefte zu besorgen sollte gleich zu den Funkschau-

Sonderheften „ZX-81-Kochbuch“ greifen; er wird dort auch noch eine Vielzahl von Anwendungen vorfinden. Die Platinenvorlagen sind ebenfalls abgebildet. Es dürfte nicht zu schwer sein, die Platine herzustellen, denn sie ist nur einseitig aufgebaut.

Einige Schwierigkeiten gab es mit der Adressierung der Platine, die aber nicht in der Platine sondern in der unvollständigen Adreßdekodierung des ZX-81 zu suchen waren. Wir haben uns dieses Problems nochmals angenommen und schlagen eine Adressierung vor (s. Bild), die nicht nur dem ZX-81 sondern auch noch dem ZX-Spectrum gerecht wird.

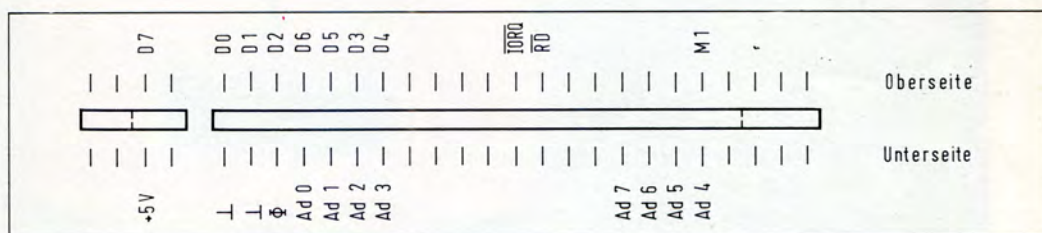
Hier die Hintergründe:

Bei den IN- und OUT-Befehlen des Z80-Mikroprozessors, der diese Computer antreibt, wird das Signal *IORQ* gesetzt, um den Peripheriezugriff von den Speicherzugriffen zu unterscheiden. Gleichzeitig werden auch noch die Adreßleitungen A0...A7 gesetzt, so daß der Mikroprozessor 256 verschiedene Peripherieregister auseinanderhalten kann (unter bestimmten Bedingungen werden sogar A0 bis A15 gesetzt). Dies funktioniert jedoch nur, wenn auch ein Adreßdecoder verwendet wird, der in beiden Computern jedoch eingespart wurde. So bleibt eine Adressierung nur dann eindeutig,

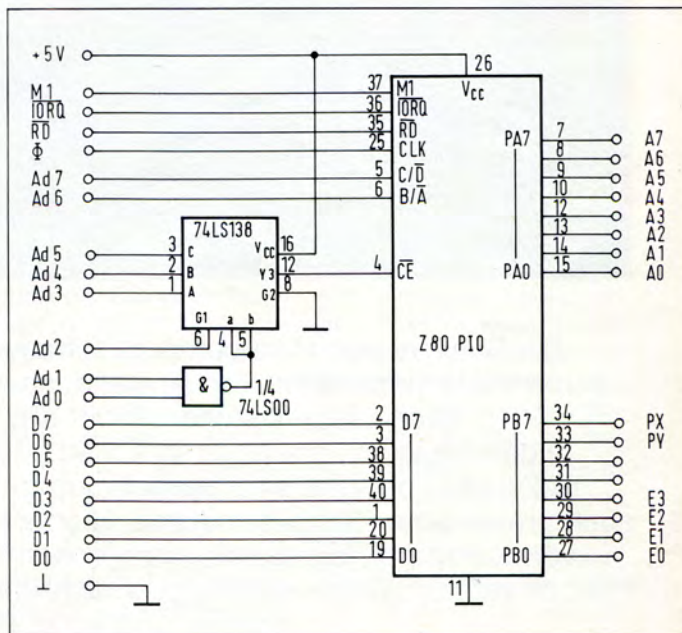
wenn immer nur genau ein Bit aktiviert wird, in diesem Falle auf Low gesetzt wird. Der Preis der Einsparung: man kann nur noch 8 statt 256 Register adressieren. Wenn aber mehr nicht vorgesehen sind, ist das Verfahren ja durchaus legitim. Ein Beispiel: Wird z. B. beim ZX-Spectrum in einem OUT-Befehl A0 auf Low geschaltet, so wird ein Register adressiert, das den Lautsprecher, den Ausgang für den Cassettenrecorder und die Farbe des Bildschirmrahmens steuert. Dieses Register wird mit jeder geraden Adresse erreicht, aber nur ohne Nebenwirkungen auf andere Register unter Adresse $FE_{16} = 254$.

Beim ZX81 ist ein Low-Signal auf den Adreßleitungen A0 und A1 schon belegt, mit A2 = Low wird der Drucker angesprochen. Das gleiche gilt auch für den ZX-Spectrum. Beim Spectrum sind

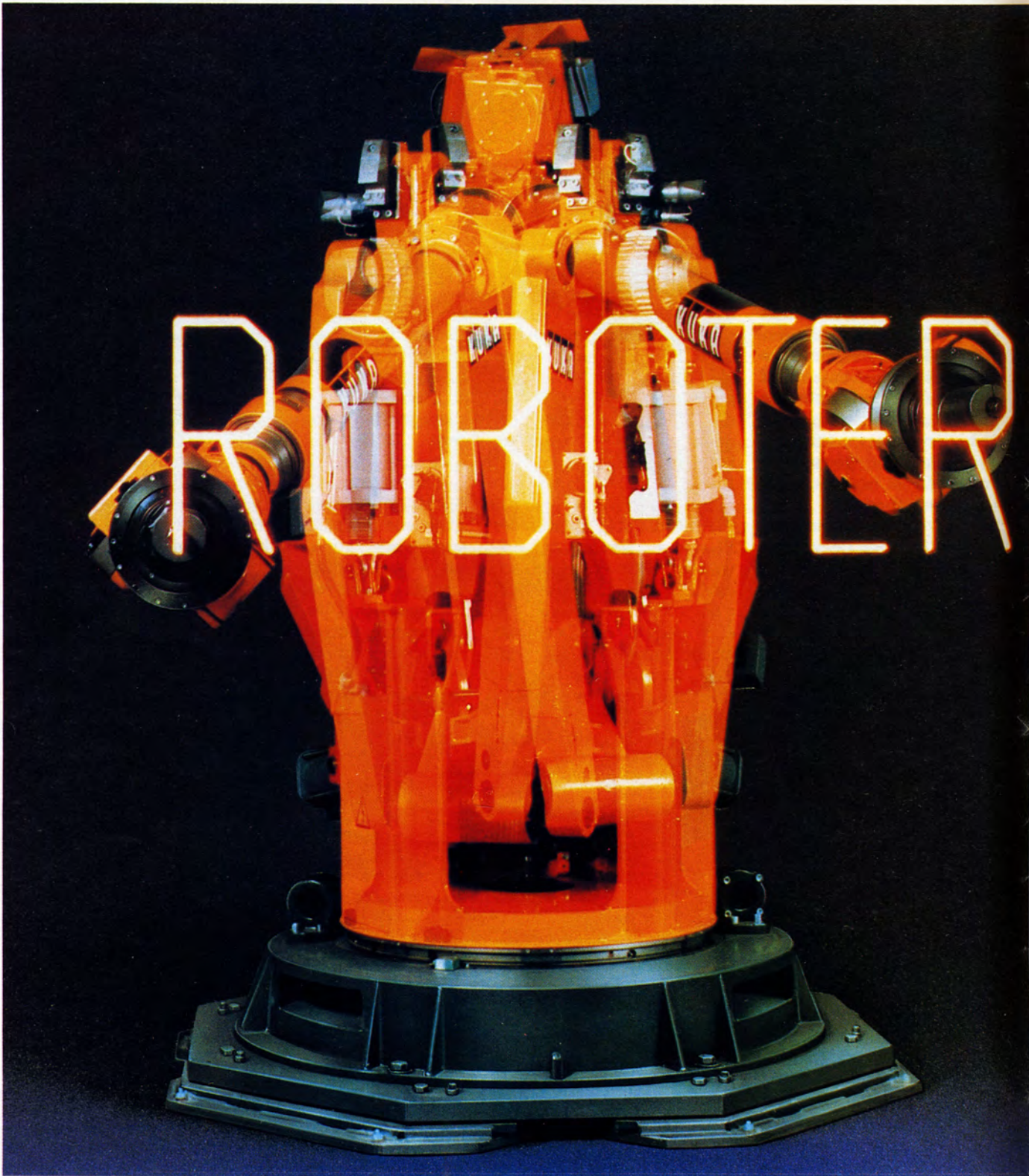
Fortsetzung auf Seite 28



▲ Pin-Belegung des USER-Port-Steckers beim ZX-Spectrum.



Die Daten-Drehscheibe ► des ZX-81. Die hier abgebildete Pin-Belegung paßt sowohl für den ZX-81 als auch für den Spectrum.



Die Besitzer von Heimcomputern fragen immer mehr nach sinnvollen Anwendungen ihrer Zauberkistchen. Nicht allein Videospiele, Programmiersprachen oder raffinierte Tool-Kits sind gefragt, sondern auch praxisnaher Computereinsatz. Hierzu zählt die Robotik, eine Synthese aus dem Einsatz von Computer, Elektronik und Mechanik. Diese Serie macht Sie mit den Grundlagen der Robotik vertraut. Dabei werden gebräuchliche Computer, einfache Schaltungen, kleine Modell-roboter und kurze Programme eingesetzt. Ein faszinierendes Gebiet!

Einführung in die Robotik:

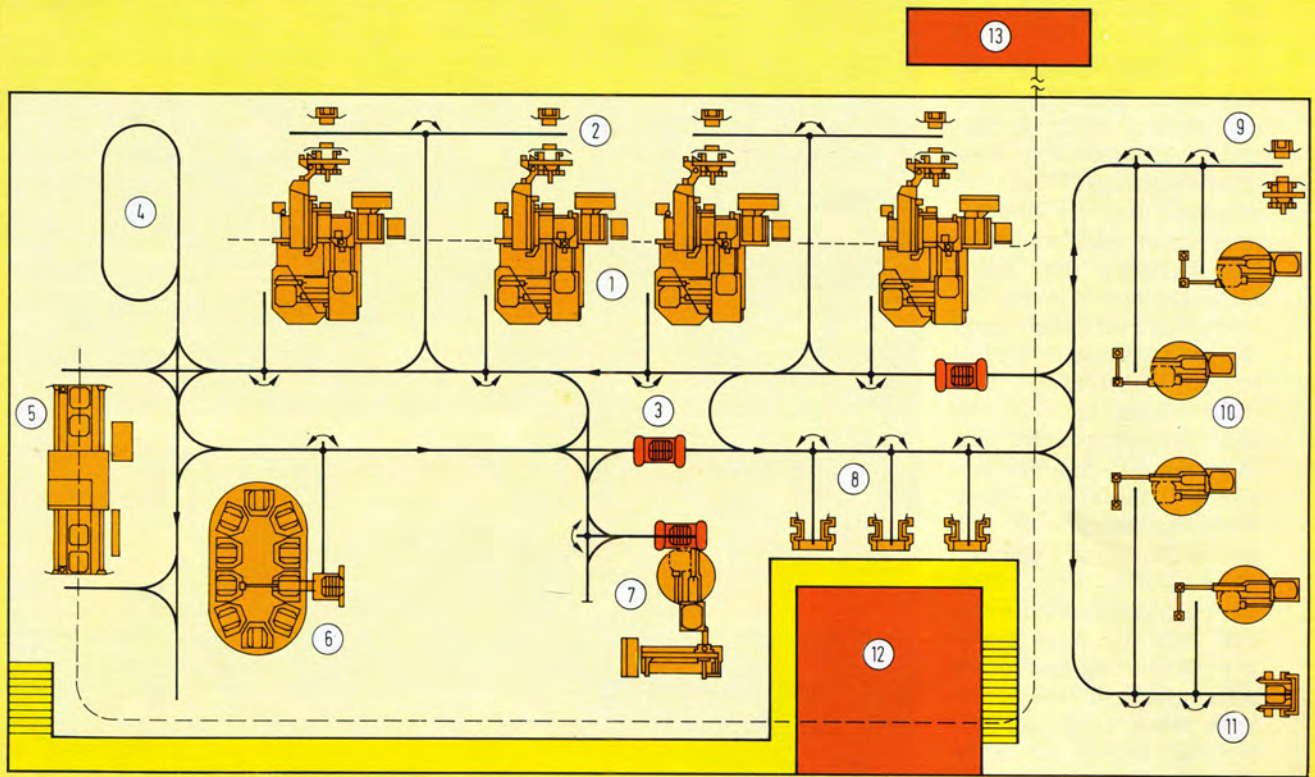
Die stummen

TEIL 1

Bild 1:
Grundrißplan eines flexiblen
Fertigungssystems. Die
Zahlen stehen für folgende
Anlagen:
(1) vier Bearbeitungs-
maschinen,
(2) vier Übergabestationen,
(3) drei computergesteuerte
Flurförderfahrzeuge,

(4) Serviceplatz der
Flurförderfahrzeuge,
(5) automatische
Teilerreinigung,
(6) Zwischenspeicher für
zehn Werkstücke,
(7) Koordinatenmeßmaschine,
(8) Zwischenlager für
Werkzeuge,

(9) Übergabeplatz für
Werkzeuge,
(10) vier Be- und Entlade-
roboter,
(11) Übergabeplatz für Paletten,
(12) Steuerzentrale und
Computerraum,
(13) Versorgungs- und
Entsorgungsleitungen unter
dem Hallenboden.



Der Roboter schreibt seinen Namen „in die Luft“. Dadurch scheint er zwei Arme zu haben. (Aufnahme: Kuka)

Die Fabrik der Zukunft wird eine Vielzahl hochtechnischer Geräte beinhalten, die alle zusammengenommen ein sogenanntes flexibles Fertigungssystem bilden. Schauen wir uns **Bild 1** an, das im Grundriß ein solches flexibles Fertigungssystem zeigt. Es handelt sich um eine Maschinenbaufirma, die komplizierte Formwerkzeuge für Kunststoff-Spritzgußmaschinen herstellt.

Zur eigentlichen Herstellung dienen vier Werkzeugmaschinen (1), die eine Vielzahl von Bearbeitungsvorgängen durchführen können, da sie in einem Magazin einen Vorrat von bis zu 90 Werkzeugen halten, Bohrer, Fräser, Stähle usw. Diese Bearbeitungsmaschinen sind mit Übergabestationen (2) aus-

stattet, an denen die Werkstücke aufgespannt, aber auch wieder abgegeben werden. Der Zu- und Abtransport erfolgt durch automatische Flurförderfahrzeuge (3), von denen sich drei in der Halle tummeln. Sie folgen den im Fußboden verlegten Induktionsschleifen und können auch an den mit Doppelpfeil bezeichneten Stellen Kehrtwendungen machen. Mit (4) ist der Park- und Serviceplatz der Flurförderfahrzeuge bezeichnet.

Die bearbeiteten Teile werden in einer Waschanlage (5), die ebenfalls vollautomatisch arbeitet, gereinigt und in einer Vermessungsapparatur (7) auf Maßhaltigkeit überprüft. Um die Maschinen genügend auslasten und Spitzen abbauen zu können, befindet sich bei (6) ein Zwischenlager

in Karussellform. Die Flurförderfahrzeuge transportieren aber nicht nur die Werkstücke, sondern versorgen die Bearbeitungsmaschinen auch mit Werkzeugen, falls diese abgenutzt sind. Die Werkzeuge stehen an Position (8) parat und können an Position (9) ausgetauscht werden.

Die Werkstücke wiederum sind auf standardisierten Paletten montiert, so daß sie einheitlich von all diesen Maschinen aufgenommen werden können. Die Paletten werden bei (11) eingeschleust bzw. entnommen, während vier Roboter (10) die Paletten be- und entladen. Gesteuert wird die ganze Fabrik von dem Computerraum (12), der der Übersichtlichkeit wegen erhöht angebracht ist. Letztendlich sind die ganzen Versorgungsleitun-

Diener

gen und die Wiederverwertung des Abfallmaterials (13) unter dem Hallenboden verlegt.

So futuristisch sich die Beschreibung anhören mag, die Fabrik ist bereits in Betrieb. Es handelt sich um den Kunststoffformenbau der Firma Cincinnati Milacron, selbst ein Roboterhersteller.

Daher also, werden Sie sich sagen. Worin liegt nun der Nutzen einer so modern ausgestatteten Produktionsanlage? Ist es eventuell nur ein Renommierobjekt oder ein Ausstellungsraum des Roboterherstellers? Wie der Betriebsleiter versichert, liegt der Hauptnutzen in der höheren Flexibilität. Was diese Firma herstellt, ist kein Massenprodukt, das in großen Stückzahlen von einem Band laufen kann. Vielmehr handelt es sich gar nicht selten sogar um Einzelstücke. Etwa 3500 verschiedene Produkte eignen sich für diese Fertigungslinie. Lag die Vorbereitungszeit für ein derartiges Stück wegen der Teilvervielfalt bisher bei ca. 20 Tagen, so kann sie nun auf einige Stunden abgekürzt werden.

Wenn auch der Betrieb in diesem flexiblen Fertigungssystem mit geringer Mannschaft oder z. T. sogar unbemannt durchgeführt werden kann, so steckt doch eine ganze Menge Arbeit dahinter. Die Arbeit verbirgt sich in all den Steuerungsanlagen, den Automaten und nicht zuletzt in dem riesigen Softwarepaket, das diese Fertigung in Gang hält. Es wäre sicherlich interessant, aufzurechnen, wieviele Arbeitsplätze entfallen und wieviele neu geschaffen wurden; die Zahlen liegen uns in diesem Fall jedoch nicht vor.

Eines aber kann der Leser, der die Beschreibung dieser Anlage genau studiert hat, feststellen: die neuen Arbeitsplätze werden sicherlich eine wesentlich höhere Qualifikation erfor-

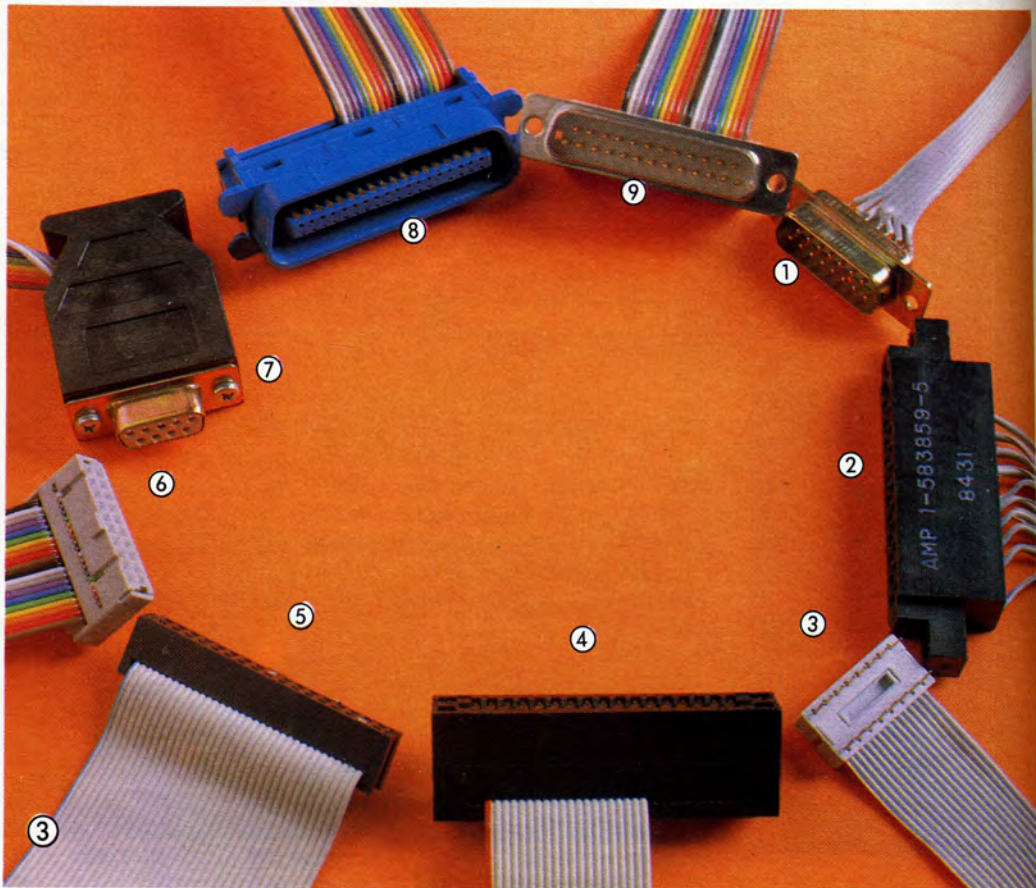
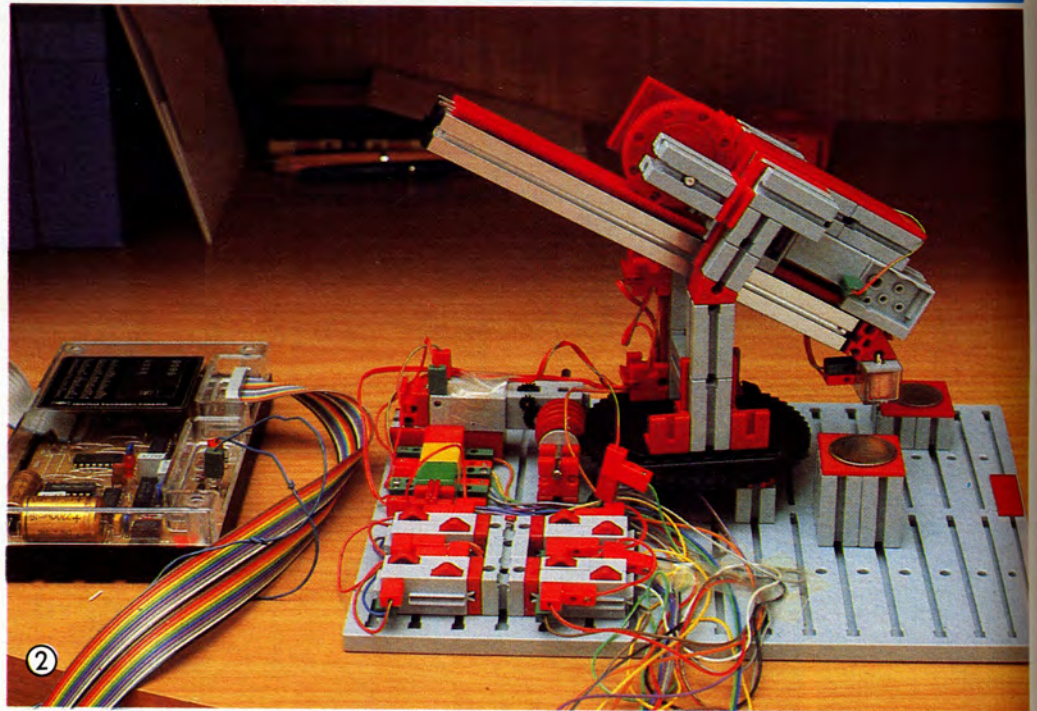


Bild 2:
Modell eines Industrieroboters, aufgebaut aus fischertechnik computing.

Bild 3:
Eine kleine Auswahl von Steckersystemen zum Experimentierschluß. Die dazugehörigen Computertypen: (1) Acorn analog in IBM ein, (2) Commodore aus, (3) Apple II, (4) Schneider CPC 464 und 664 aus, (5) NDR-Kleincomputer, (6) Acorn digital in, (7) Schneider/Commodore, (8) Schneider 6128 aus, (9) IBM aus.

Für
viele Computertypen
geeignet:
ELO-Robotik

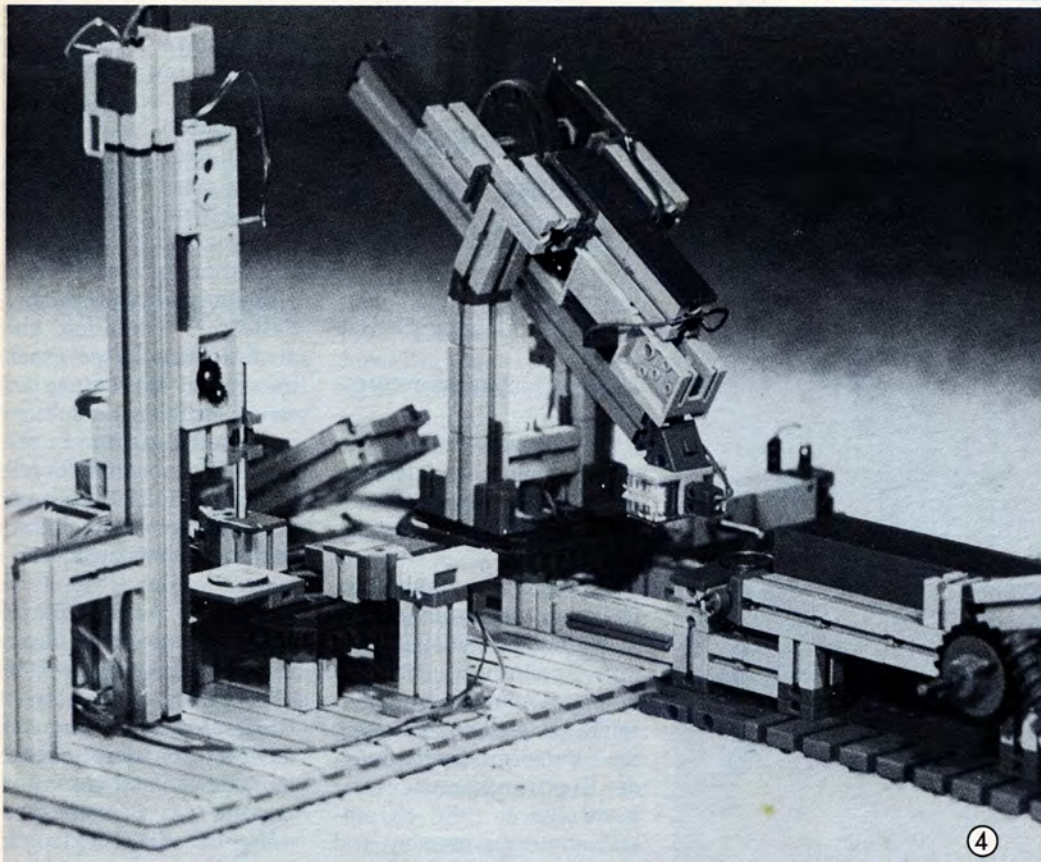


Tabelle 1:
Benutzter Steckverbinder für die verschiedenen Computer:

Acorn B: zweireihige Pfostenverbinder mit 2x 10 und 2x 13 Polen, Rastermaß 2,54 mm. Analogeingabe über 13poligen D-Stecker.

Apple II (alle Modelle außer IIc): 16poliger IC-Verbinder.

Commodore (alle Modelle außer C16 und 116, PC und Amiga): Rippenstecker mit 2x12 Polen, Rastermaß 3,96 mm. Bei den Computern 8032-SK und 8096-SK wird abweichend ein Stecker mit 2x12 Polen nach der IEEE-488-Bus-Norm verwendet. Eingabe über 9poligen D-Stecker (Joystickport, nicht bei den PETs und CBMs).

IBM-PC und Kompatible: Zum Anschluß der Eingabeplatine wird die Game-card benötigt; sie verwendet einen 13poligen D-Stecker. Zur Ausgabe wird ein 25poliger D-Stecker benutzt.

Maccorn 2000: einreihige 10polige Steckverbinder im Rastermaß 2,54 mm je nach Zahl der verwendeten Ein- und Ausgänge.

NDR-Klein-Computer: Verwendung der IOE-Karte. Die Ein- und Ausgabeschaltungen können auf dem Experimentierfeld aufgebaut werden. Andernfalls ist ein zweireihiger Steckverbinder 2x20 Pole, Rastermaß 2,54 mm, vorzusehen.

Schneider-Computer: Modell CPC-464 und CPC-664 verwenden zum Anschluß der Ausgabeplatine einen 2x17poligen Rippenstecker im Rastermaß 2,54 mm. Das Modell CPC-6128 verwendet einen Centronics-Normstecker mit 36 Polen. Zum Anschluß der Eingabeplatine wird einheitlich ein 9poliger D-Stecker eingesetzt.

Bild 4:
Modell eines flexiblen Fertigungssystems mit Förderband, Roboter und Bearbeitungsmaschine.

dern. Dies bedeutet letztlich für jeden, daß er sich ein Leben lang in einem Lernprozeß befindet, um sich mit der wandelnden Arbeitswelt auseinanderzusetzen zu können. Warum also nicht gleich die Gelegenheit beim Schopfe fassen und bei der

ELO-Serie „Einführung in die Robotik“ mitmachen? Sie brauchen dazu neben Ihrem Heimcomputer nicht viel (wir werden es weiter unten genau aufzählen) und es ist, nebenbei, mit recht viel Spaß und Freude verbunden. Wir werden in einfachen

Schritten vorangehen. Wir werden uns ein kleines Experimentierlabor aufbauen, das wir auch außerhalb dieser Serie sinnvoll einsetzen können. Wir werden jedes Stück einzeln besprechen und testen. Auch die Software wird in kleinen Schritten aufgebaut, die alle erläutert werden. Der Autor hält nicht viel von Mammutprogrammen, die einen wahren Zauber auf dem Bildschirm entfesseln, aber nicht verstanden werden können. Wichtiger ist das Herausarbeiten der elementaren Programmierschritte ohne großen Ballast.

Auch die Maschinen werden wir an unseren Computer anschließen. Nun sollen Sie sich aber keinen zigtausend Mark teuren Roboter ins Wohnzimmer stellen. Vielmehr werden wir Modelle benutzen, die in sinnvoller Vereinfachung die wesentlichen Prinzipien der Robotik deutlich herausstellen (Bild 2). Niemand wird uns schelten, daß wir „Spielzeug“ verwenden und nicht „richtig“ damit arbeiten können. Wir wollen ja schließlich die Elemente der Robotik kennenlernen und da spielt es keine Rolle, ob der Roboter nun zwanzig Zentimeter oder zwei Meter hoch ist, ob er einige Gramm oder eine halbe Tonne heben kann. Wir werden uns auch nicht nur mit Robotern befassen. Nicht umsonst haben wir die Beschreibung der flexiblen Fertigungsanlage vorangestellt. Einfache Modelle wie z. B. Werkzeugmaschinen oder Fördereinrichtungen stehen bei uns neben den Robotern und werden am Schluß wie in der Realität miteinander verkettet.

Ein Blick in die Werkzeugkiste

Was brauchen wir nun, um an der „Einführung in die Robotik“ teilzunehmen? Kern- und Angelpunkt unseres Experimentierlabors

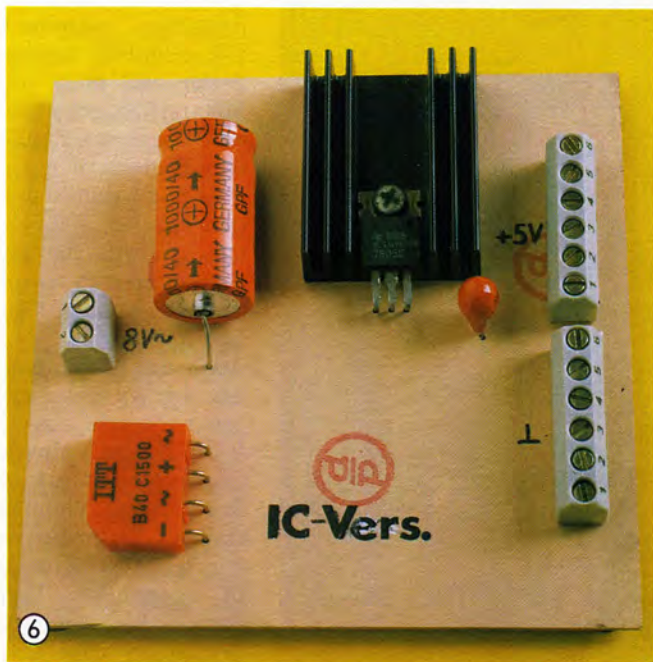
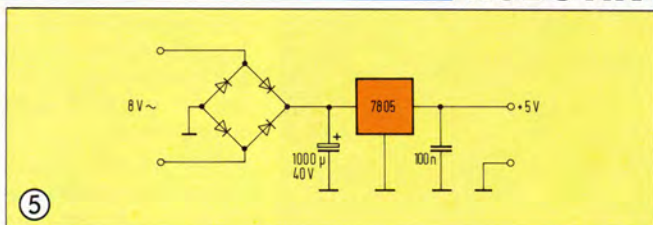


Bild 5:
Schaltbild und Foto der
IC-Versorgung.

Bild 6:
Musteraufbau der
IC-Versorgung.

wird wohl ein Heimcomputer sein. Die Computer, die derzeit auf dem Markt vertreten sind, dürften sich wohl ausnahmslos für die Experimente eignen. Allerdings können wir Schaltungen und Programme nicht für alle diese Computer angeben. Hier werden wir eine Auswahl treffen müssen, die keineswegs etwas über die Qualitäten eines bestimmten Computers aussagt, sondern lediglich über dessen Verbreitungsgrad. Wir werden uns namentlich auf die Computer Commodore 64 und 128 sowie Schneider CPC 464, 664 und 6128 beziehen. Andere Computer werden manchmal gestreift, insbesondere dann, wenn es um das Anschließen der Robotersteuerungen geht. Die Programme werden Sie gegebenenfalls anpassen müssen. Jedoch sollte auch dies kein unüberwindliches Hindernis sein, da sie

nahezu vollständig in Basic geschrieben sind.

Des weiteren werden wir einige Platinen erstellen, die zum Anschluß der Roboter dienen. Für diese Platinen werden die Druckvorlagen und Bestückungspläne weiter hinten im Heft veröffentlicht. Die Platinen sind recht einfach zu erstellen, wir haben sie mit nur zwei Sternchen (s. o.) gekennzeichnet. Die Platinen bilden zusammen ein Experimentiersystem, das sich in vielfältiger Weise zusammenstellen und ausbauen läßt.

Wenn Sie die fertig geätzten und gebohrten Platinen beziehen, brauchen Sie nur wenig Werkzeug, um sie in Funktion zu setzen:

– Einen Lötkolben, der maximal 30 W haben sollte, wenn er unreguliert ist. Besser ist natürlich ein temperaturgeregelter Elektroniklötkolben.

Auch eine Fabrik der „Zukunft“ werden Sie bauen

- Einen kleinen Seitenschneider.
- Einen feinen Schraubendreher.
- Eine Abisolierzange; wer viel Geschick besitzt, kann das Abisolieren auch mit einem Bastelmesser versuchen.

– Eine kleine Flachzange oder eine stabile Pinzette.

– Vielfachmeßinstrument, mit Spannungs-, Strom- und Widerstandsmeßbereich. Es wird dazu dienen, die Versorgungsspannung der Elektronikbausteine zu kontrollieren, die Stromaufnahme zu messen und die Leitungen und Platinen auf Durchgang zu testen.

Die Schaltungen, die jeweils auf einer Platine Platz finden, sind nicht allzu umfangreich. Geübtere Hobbyelektroniker werden sie auch genausogut auf Lochrasterplatinen aufbauen können.

Bei dem benötigten Material handelt es sich um handelsübliche integrierte Schaltungen (ICs), meist aus der sogenannten TTL-Reihe. TTL steht für Transistor-Transistor-Logik und kennzeichnet den inneren Aufbau dieser Schaltungen. Dies sind die Arbeitspferde der Elektronik, und die von uns benutzte Unterfamilie fängt durchweg mit dem Familiennamen 74LS... an.

Was sonst noch an Material benötigt wird, Transistoren, Dioden und Kondensatoren, ist auch handelsüblich. Außerdem werden Sie noch einen oder zwei Stecker benötigen, um den Anschluß zu Ihrem Computer herzustellen. Schauen Sie in **Tabelle 1** und auf **Bild 3**.

Daneben sollten Sie sich für die Verkabelung Litze

zulegen, am besten in möglichst vielen verschiedenen Farben, um nachher die Leitungen besser auseinanderhalten zu können. Der Querschnitt 0,14 mm² ist für alle unsere Zwecke ausreichend. Die Litze können Sie außer im Elektronikhandel auch bei dem Modelleisenbahnbedarf erhalten. Sie können aber auch farbcodiertes Flachbandkabel einkaufen, dessen Einzeladern sich gut auftrennen lassen. In diesem Fall haben Sie bereits eine große Farbvielfalt und können auch Kabelpaare, -drillings usw. zusammenlassen.

Außerdem werden Sie einen oder mehrere Trafos mit einer Ausgangsspannung von ca. 8 V benötigen. Die Trafos sollen einerseits die Motoren der Robotermodelle antreiben, andererseits die Betriebsspannung für die Elektronikbausteine zur Verfügung stellen. Wir benötigen einen Strom bis zu 1 A. Wenn ein Trafo diesen Strom nicht liefern kann, so können Sie die Aufgaben auch aufteilen und einen Trafo für die Schaltungsvorsorgung und einen anderen Trafo für den Antrieb der Modelle einsetzen.

Aussichten

Damit Sie die Katze nicht im Sack kaufen müssen, wollen wir nun noch ein paar Details angeben. Wir werden in den kommenden Folgen abschaltbare Pufferplatinen für Datenein- und -ausgabe aufbauen. Zwei verschiedene Verstärkerplatinen dienen zur Ansteuerung der Aktuatoren: Kleinmotoren, Lampen, Relais, Magnetventile usw. Mit diesen Hilfsmitteln werden wir einen Dualzahlentrainer und eine Verkehrsampel aufbauen. Danach geht es los mit dem Maschinenpark der Fabrik der „Zukunft“, die eingangs geschildert wurde: Bearbeitungsmaschinen, Roboter, Förderbänder. Dabei werden wir auch

studieren, wie mehrere Aufgaben miteinander zu einem flexiblen Fertigungssystem verkettet werden (**Bild 4**). Und ganz am Schluß kommen noch ausfallsichere Steuerungssysteme, die auf Mehrrechnersystemen basieren.

Die Stromversorgung

Neugierig geworden? Dann können wir gleich noch mit einer kleinen Übung zum Aufwärmen beginnen. Am Anfang eines jeden Projekts steht die Stromversorgung. Was nützte die schönste Schaltung, wenn wir keine Stromversorgung besitzen? Das Schaltbild der Stromversorgung ist in **Bild 5** wiedergegeben. Der Eingang muß mindestens mit ca. 7 V Wechselspannung gespeist werden. Mehr als 12 V sollten Sie aber auch nicht anlegen. Wenn Ihr Trafo nur einen Gleichspannungsausgang besitzt, ist es auch nicht weiter schlimm. Auf jeden Fall wird der Strom erst einmal über einen Brückengleichrichter geführt und ggf. gleichgerichtet. Die Gleichspannung wird dann zur Glättung an einen Elektrolytkondensator gelegt. Von dort aus geht es weiter zu einem integrierten Spannungsregler, der eine recht genau eingehaltene und brummfreie 5-V-Gleichspannung erzeugt. Abschließend erfolgt noch einmal eine Siebung am Ausgang, um auch Regelschwingungen zu verhindern. Die 5 V werden als Stromversorgung für die integrierten Schaltungen

(IC) der TTL-Familie eingesetzt. Wir werden sie im folgenden als IC-Versorgung (im Gegensatz zur Motor-Versorgung) bezeichnen.

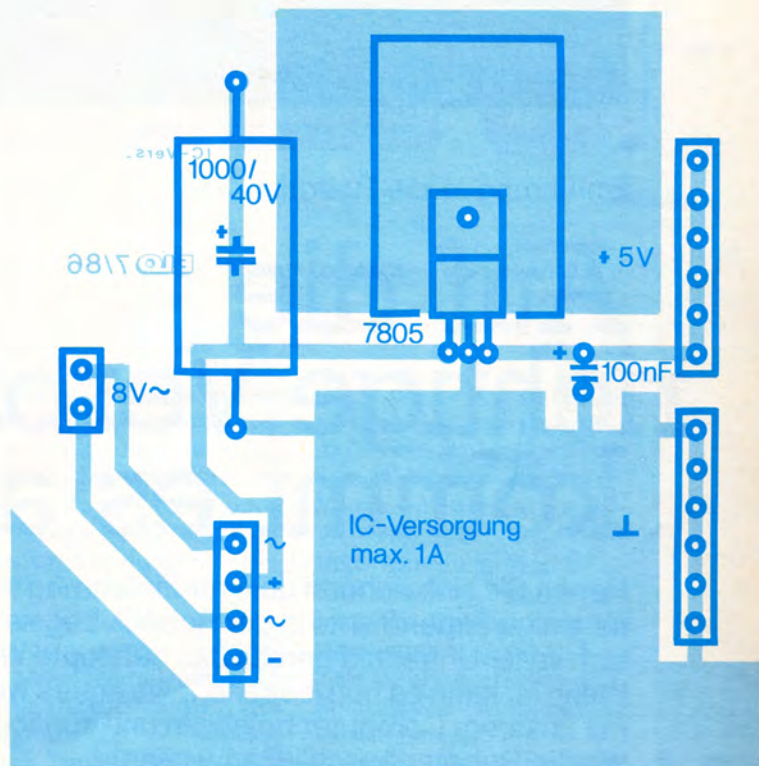
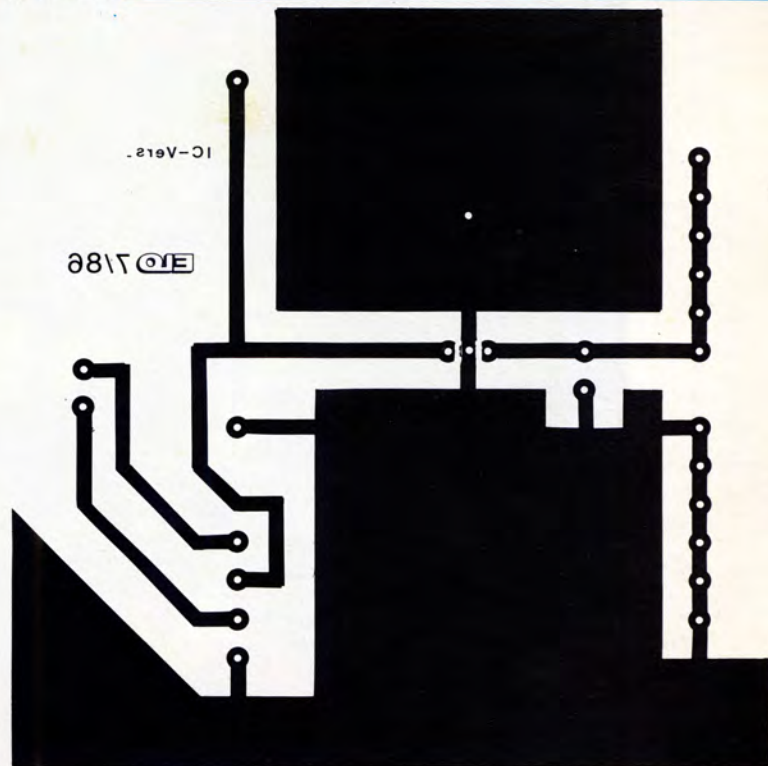
Der Aufbau (**Bild 6**) stellt weiter keine großen Anforderungen. Achten Sie auf die richtige Polung des Brückengleichrichters und des Elektrolytkondensators. Der Spannungsregler sieht wie ein Leistungstransistor aus. Winkeln Sie die Anschlußbeine mit der Flachzange ab, so daß Sie das Gehäuse des IC mit der Platine verschrauben können. Bevor Sie den Spannungsregler allerdings festschrauben, wird er noch auf den dazu passenden Kühlkörper gesetzt. Zum Anschluß der Versorgungsspannung und als Ausgang der IC-Versorgung bestücken wir die Platine mit Lüsterklemmen. Diese ergeben zuverlässige Verbindungen, die nach Bedarf leicht wieder gelöst werden können.

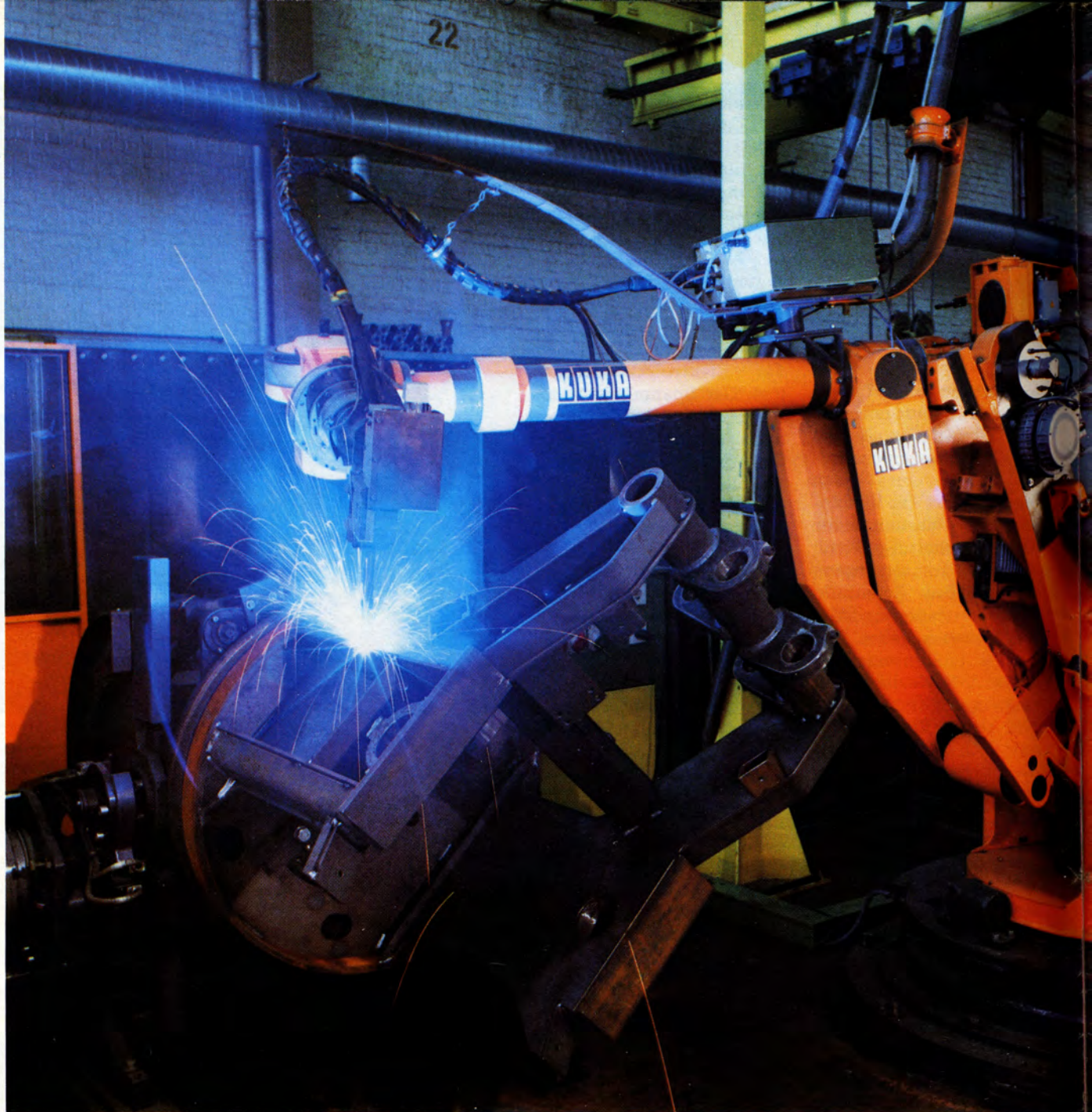
Wenn Sie nachher die Experimente aufbauen, verwenden Sie die zuvor erwähnte Litze, schneiden sich ein passendes Stück ab und entfernen die Isolation an beiden Enden. Verdrillen Sie die feinen Drähtchen der Litze und verzinnen Sie sie, damit sie sich nachher nicht wieder in ein struppiges Büschel verwandeln. So präpariert können Sie die Verbindungsleitungen auch mehrmals verwenden. Bei Aufbauten, die längere Zeit verwendet werden sollen, in Schulen oder Clubs z. B. oder bei der Ausarbeitung eines größeren Projekts, empfiehlt es sich, die Leitungen ordentlich zu verlegen und mit Kabelbindern zu fixieren. Dieser Rat erfolgt nicht deshalb, weil der Autor ein Ordnungsfanatiker wäre, sondern weil wir ja schließlich Objekte aufbauen, die sich nachher bewegen. Wir sollten also vermeiden, daß sich jene in ihren Zuleitungen verstricken.

Dr. Roland Schulé

Stückliste IC-Versorgung

- 1 Brückengleichrichter
B40C1500/1000, 1,5 A/40 V
- 1 Elektrolytkondensator,
1000 µF/40 V
- 1 Festspannungsregler 7805,
+5 V/1 A
- 1 Keramikkondensator,
100 nF
- 1 Kühlkörper,
ca. 29 mm × 37 mm
- 2 Anschlußklemmen, 6polig
- 1 Anschlußklemme, 2polig
- 1 Platine, IC-Versorgung





Aufnahme: Kuka

Einführung in die Robotik:

Auf die richtige Verbindung kommt es an

Haben Sie sich anhand unserer Einführung in der letzten ELO schon für einige Experimente in der Robotik begeistern können? Haben Sie sich voller Ungeduld bereits das benötigte Werkzeug bereitgelegt? Wenn ja, kann es nun losgehen. Als erstes wollen wir uns ein bißchen mit unserem Computer befassen und zunächst einmal überlegen, wo wir die Roboter anschließen wollen.

Die Funktionsgruppen innerhalb des Computers werden durch einen sogenannten Bus verbunden. Dabei handelt es sich um eine Daten-schiene, zu der alle angeschlossenen Baugruppen Zugriff haben. Der Bus des Computers ist meist auch an einen Stecker geführt, so daß an dieser Stelle weitere Ausbaustufen angeschlossen werden können, z. B. eine Speichererweiterung, eine zusätzliche Ein-/ Ausgabe-Schnittstelle oder ein zusätzlicher Mikroprozessor. Die dort angeschlossenen Baugruppen nehmen praktisch den



Rang von Systemkomponenten ein. Auf diese Weise lassen sich sicherlich auch Ein- und Ausgabeschnittstellen für Roboter aufbauen. Allerdings wird an eine solche Schnittstelle auch eine erhöhte Anforderung gestellt. Sie muß anhand der Signale auf dem Adreßteil des Bus erkennen, wann der Mikroprozessor mit ihr Daten austauschen will und diese Daten innerhalb eines Arbeitstaktes des Mikroprozessors (typisch 1 µs) übernehmen oder zur Verfügung stellen. Daher ist es vielleicht nicht verkehrt, wenn wir uns zur

Einführung auf Schnittstellen verlassen, die der Computer schon eingebaut hat. Dort abgelegte Daten können wesentlich gemächlicher behandelt werden. Z. B. bleiben bei der Druckerschnittstelle die Daten anstehen, bis sie durch nachfolgende Daten anlässlich des nächsten Ausgabekommandos in Basic überschrieben werden. Wenn wir wollen, können sie also tagelang oder auch nur den Bruchteil einer Sekunde stehenbleiben. Die Eingabe an der Steuerknüppelschnittstelle kann auch leicht verwendet werden und läßt sich jederzeit per Basic-Kommando einlesen. Sollten wir später dann unser Roboterlabor weiter ausbauen wollen, können wir immer noch auf einen Ein-/Ausgabebaustein an dem Bus des Computers zurückgreifen. Die Platinen, die wir im folgenden vorstellen werden, lassen sich dann allesamt noch weiter verwenden. Der Vorteil einer Ein-/Ausgabeschnittstelle wird eben darin bestehen, daß wir sogar mehrere Platinen zur

TEIL 2

chen Sie sich die Schnittstelle Ihres Computers heraus. Schauen sie Ihrem Computer auf die Rückseite oder auf die Unterseite, damit Sie mit den folgenden Beschreibungen vergleichen können.

Acht Datenleitungen werden wir benötigen, die im folgenden als Ausgang 0 bis Ausgang 7 bezeichnet sind (einige Computer haben weniger). Diese Ausgänge müssen wir mit den entsprechenden Eingängen 0 bis 7 der Ausgabeplatine verbinden. Außerdem wird noch eine Masseleitung benötigt. Sie wird mit dem Masseanschluß verbunden. Ganz ähnlich wie bei der hohen Politik muß zuerst eine gemeinsame Basis vorhanden sein, bevor detaillierte Verhandlungen aufgenommen werden können. Die Masseleitung sorgt dafür, daß der Computer und die Ausgabeplatine sich auf gleichem Potential befinden und die Signale austauschen können.

Acorn B

Der Acorn B dürfte wohl der für Steuerungszwecke

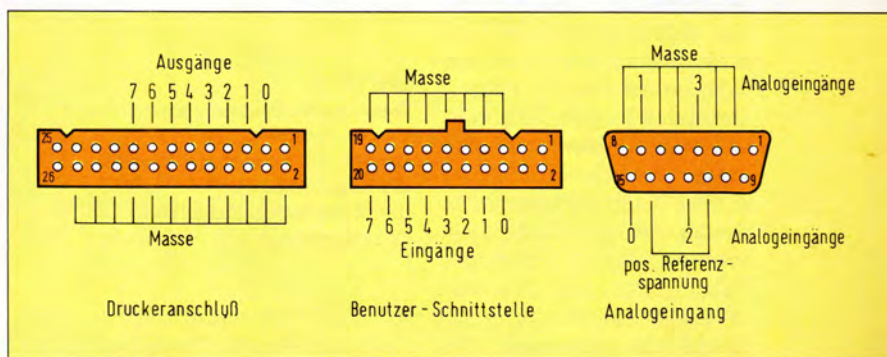
ein Analogwertanschluß (wir kommen später darauf zurück, was das ist) mit vier Eingabekanälen. Der Druckeranschluß ist auf einen zweireihigen Postenverbinder im Rastermaß 2,54 mm mit 2×13 Anschlußkontakten geführt (Bild 1a). Der Druckeranschluß wird unter der Adresse 65121 mit dem ?-Operator angesprochen. Die Benutzerschnittstelle besitzt einen ähnlichen Stecker, jedoch mit 2×10 Kontakten. Sie wird unter Adresse 65120 angesprochen. Wir empfehlen jedoch die Verwendung der Druckerschnittstelle, um uns die Benutzerschnittstelle für die Eingabe aufzusparen.

Apple II (alle Modelle außer IIC)

Der Apple II hat keine eingebaute Benutzer- oder Druckerschnittstelle. Insofern wird dieser Computer sicherlich einer der ersten sein, für den wir eine Ein-/Ausgabeschnittstelle aufbauen oder fertig erwerben müßten. Für erste Experimente können wir aller-

Bild links:
Roboter schweißen Roboter.

Bild 1a (rechts):
Anschlußbelegung des Acorn B (Drucker, Benutzerschnittstelle und Analogeingang).



gleichen Zeit anschließen können. Doch dies benötigen wir, wie gesagt, für unsere ersten Experimente noch nicht.

Ausgabeschnittstellen

Befassen wir uns zunächst mit der Ausgabe. Im folgenden werden wir für einige Computer angeben, wie die Ausgabeschnittstelle beschaffen ist. Su-

am reichhaltigsten ausgestattete Homecomputer sein. Als Ausgabekanal läßt sich sowohl die Druckerschnittstelle als auch die eigens für Steuerungszwecke vorgesehene Benutzerschnittstelle verwenden. Letztere können wir auch genausogut für Eingaben verwenden, da ihre Datenrichtung frei programmiert werden kann. Hinzu kommt noch

dings die reichlich ausgestattete Joystickschnittstelle benutzen, die über vier Ausgangs- und drei Eingangsleitungen sowie vier Analogeingänge verfügt. Allerdings sind diese Ein- und Ausgänge im Verlaufe der Entwicklung des Computers noch für andere Zwecke eingesetzt worden. So werden Ausgangsleitungen benutzt, um die 80-Zeichen-Darstellung anzu-

steuern, und zwei Eingangsleitungen sind beim Apple IIe auf die Apfeltasten geführt. Wenn Sie die Joystickschnittstelle benutzen wollen, so entfernen Sie gegebenenfalls vorher eine 80-Zeichenkarte und benutzen Sie nicht die Apfeltasten. Die Ein- und Ausgangsleitungen sind alle mit POKE- bzw. PEEK-Kommandos unter getrennten Adressen ansprechbar (s. Bild 1b). Zum Anschluß dient ein 16poliger IC-Sockel in der rechten hinteren Ecke der Hauptplatine. Wenn Sie einen passenden IC-Sockelverbinder einstecken wollen, gehen Sie mit Bedacht vor; der Sockel ist schwer zugänglich und die Steckstifte verbiegen sich leicht.

VC-20, C-64 und C-128, PET 2000, CBM 3000, 4000 und 8000

Diese Computer haben keinen Paralleldruckeranschluß, stattdessen aber eine eigens für Steuerungszwecke vorgesehene Benutzerschnittstelle. Sie ist (bis auf einige Modelle der Serie 8000) als Leiterplattenverbinder ausgeführt. Die Kontakte auf der Unterseite sind bei all diesen Computern einheitlich – zumindest in dem Umfang, wie wir sie benutzen. Die Masse wird dabei Kontakt A oder N entnommen, acht Steuerleitungen den Kontakten C bis L, außerdem werden wir noch eine zusätzliche Steuerleitung an Kontakt M entnehmen, die wir auf den Eingang des freien Inverters führen (E8). Diese Leitung werden wir später zur Steuerung der Eingabeplatine benutzen.

Als Stecker wird ein 2x12-poliger Direktverbinder im Rastermaß 3,96 benötigt (Bild 1c). Die Adresse dieser Schnittstelle variiert von Computer zu Computer:

VC-20 : 37136
C-64 und C-128 : 56577
PETs und CBMs: 59457

Bild 1b:
Anschlußbelegung des Apple II game controller port.
Adressen:
Eingang 0 abfragen: 49249
Eingang 1 abfragen: 49250
Eingang 2 abfragen: 49251
Ausgang 0 setzen: 49240
Ausgang 0 löschen: 49241
Ausgang 1 setzen: 49242
Ausgang 1 löschen: 49243
Ausgang 2 setzen: 49244
Ausgang 2 löschen: 49245
Ausgang 3 setzen: 49246
Ausgang 3 löschen: 49247
Die Analogeingänge werden mit Hilfe der PDL-Funktion abgefragt.

Bild 1c:
Anschlußbelegung der Commodore Benutzerschnittstelle.

Bild 1d:
Anschlußbelegung der Druckerschnittstelle des IBM-PC und kompatibler Computer.

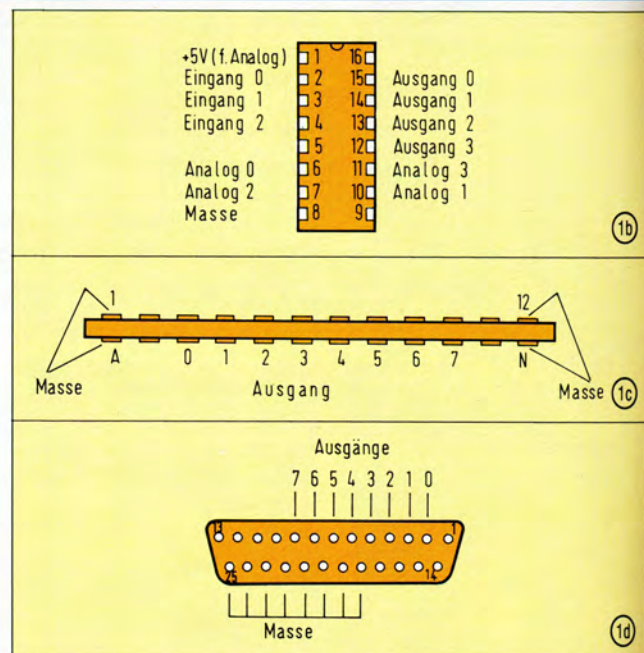
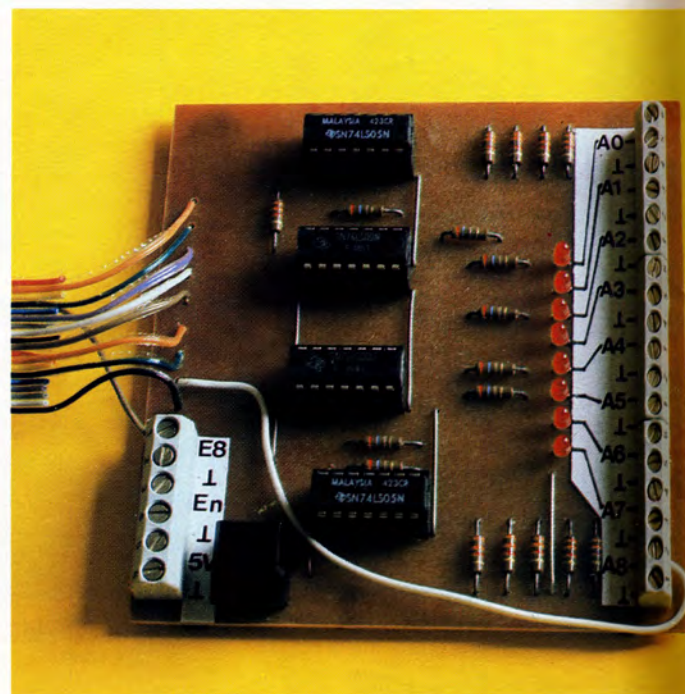


Bild 2a:
Die Ausgabeplatine schützt das Peripherie-IC im Computer. Außerdem zeigen die Leuchtdioden den Schaltungs-Status der Datenleitungen an.



Vor der Benutzung dieser Schnittstelle als Ausgabe-schnittstelle muß dies allerdings erst dem Computer mitgeteilt werden. Hierzu wird an der jeweils um zwei höheren Adresse per POKE-Kommando der Zahlenwert 255 abgelegt. Danach kann die Schnittstelle per POKE-Kommando benutzt werden.

IBM-PC und Kompatible

Bei diesen Computern wird die Druckerschnittstelle

zur Ausgabe eingesetzt. Abweichend von der Centronics-Norm wird ein 25-poliger D-Stecker verwendet, dessen Belegung, soweit für unsere Zwecke von Interesse, in Bild 1d aufgeführt ist. Außergewöhnlich ist die Adreßlage bei diesem Computer. Da die verschiedensten Druckerkarten eingesetzt sein können, ermittelt das Betriebssystem beim Einschalten die Adresse und legt sie in den Speicherzellen 1032 bis 1039 ab. Dort müssen wir sie zunächst

Computer sind vielfältig. Jeder wird anders angeschlossen

TEIL 2

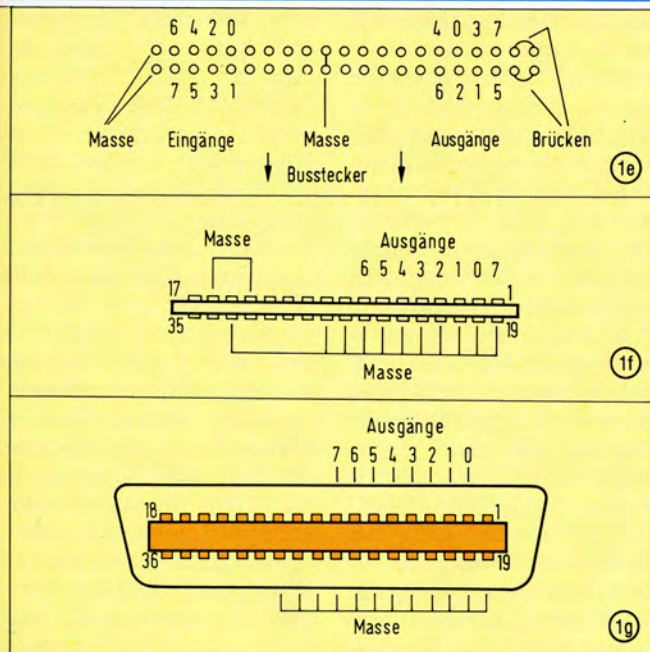


Bild 1e:
Anschlußbelegung der Ein- und Ausgabeschnittstelle der IOE-Platine des NDR-Klein-Computersystems.

Bild 1f:
Anschlußbelegung der Druckerschnittstelle der Schneider-Computer CPC 464 und 664.

Bild 1g:
Anschlußbelegung der Druckerschnittstelle des Schneider-Computers CPC 6128

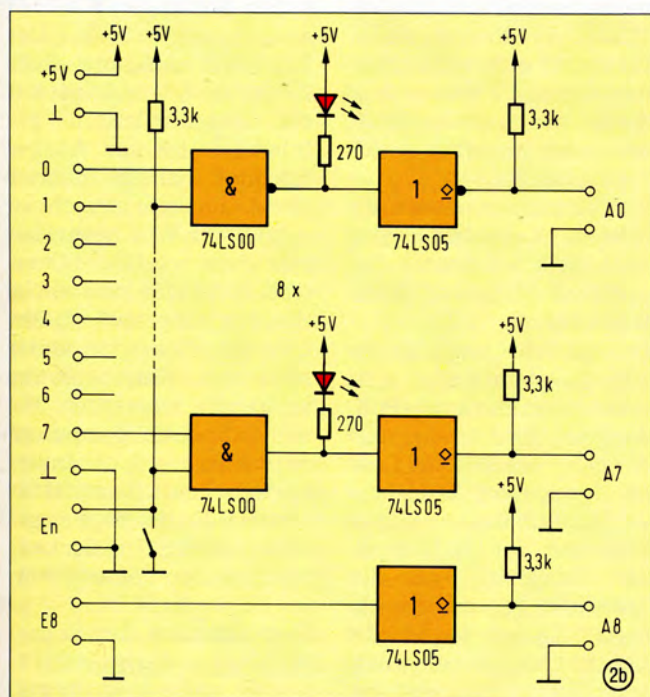


Bild 2b:
Nach dieser Schaltung ist die Ausgabeplatine leicht zu bauen.

abfragen. Die erste Druckeradresse ist in den Bytes 1032 und 1033 (low und high Byte) zu finden, die des zweiten Druckers in den Bytes 1034 und 1035 usw. Wenn der Inhalt dieser Speicherzellen 0 ist, so ist kein Drucker angeschlossen.

Maccorn 2000

Dieser Computer wurde speziell für Meß- und Steuerungszwecke entwickelt. Seine Vielzahl von Ein- und Ausgabean-

schlüssen wurden in der ELO ausführlich beschrieben, so daß auf die dortige Dokumentation verwiesen wird.

NDR-Klein-Computer

Bei dem NDR-Klein-Computer wird die IOE-(Input-, Output- und Experimentier-)Platine eingesetzt. Wir können z. B. als Ausgabeschnittstelle die Adresse 48 festlegen. Der Anschluß erfolgt über den zweireihigen Postenverbinder (siehe **Bild 1e**). Zur

Programmierung wird (unter Basic) das OUT-Kommando verwendet.

Schneider CPC 464 und CPC 664

Der Druckeranschluß ist als Leiterplattenstück ausgeführt, das auf beiden Seiten mit Kontaktstreifen versehen ist (**Bild 1f**). Die meisten Kontakte auf der Unterseite liegen auf Massepotential. Auf der Oberseite werden wir die Kontakte 1 bis 8 als acht unabhängige Steuerleitungen

verwenden. Wir müssen allerdings beachten, daß Ausgang 7 an Pin 1 wegen eines eingebauten Inverters gerade mit High und Low vertauscht erscheint. Zweckmäßigerweise führen wir daher diese Leitung zunächst auf den freien Inverter (E8 - A8), um ihn anschließend erst dem Eingang der Platine zuzuführen. Später haben wir mit dieser Leitung noch eine andere Verwendung im Sinn.

Als Gegenstück des Druckeranschlusses müssen Sie einen Direktverbinder 2x17-polig, Rastermaß 2,54 mm, besorgen. Der Druckeranschluß wird mit dem OUT-Kommando angesprochen. Die Adresse ist 61439.

Schneider CPC 6128

Der Druckeranschluß ist diesmal als Centronics-Normstecker (**Bild 1g**) ausgeführt. An der Benutzung der Anschlüsse ändert sich nichts gegenüber den kleineren Brüdern CPC 464 und CPC 664. Auch die Adresse und das Basic-Kommando ist das gleiche.

Die Ausgabeplatine

An die zuvor beschriebenen Anschlüsse schließen wir nun eine Ausgabeplatine an (**Bild 2a**). Sie hat einen mehrfachen Nutzen.

Zunächst dient sie der Schonung des im Computer eingebauten Ausgangs. Die im Computer eingebauten Peripherie-ICs sind oft nicht einmal so belastbar, um die nachfolgend geschilderten Verstärkerplatten antreiben zu können. Und im rauen Experimentierbereich kann es auch leicht mal vorkommen, daß ungewollt Kurzschlüsse auftreten.

In solch einem Fall kann das Peripherie-IC möglicherweise in die ewigen Jagdgründe der Elektronik eingehen. Dann sind guter Rat und Hilfe teuer. Denn wer traut sich die Reparatur seines Computers zu, wo heutzutage die ICs meist

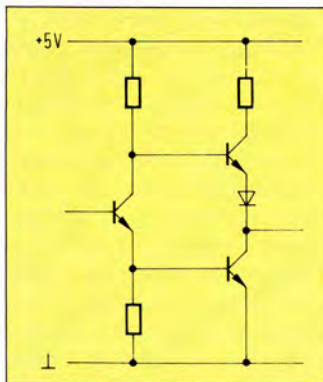


Bild 3:
Die übliche TTL-Ausgangs-
stufe, ein sogenannter
Push-Pull-Ausgang.

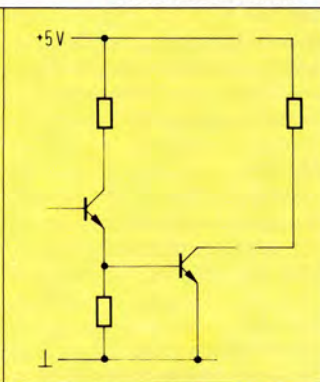


Bild 4:
Ausgangsstufe eines ICs
mit offenem Kollektor-
Ausgang.

nicht mehr gesockelt sind? Bei Verwendung der Ausgabeplatine hält sich der Schaden in Grenzen, denn wir haben eine Schaltung vorgesehen, die wesentlich robuster ist. Im extremsten Fall werden nur die ICs der Ausgabeplatine beschädigt und schützen so die Peripherie-ICs des Computers. Die von uns benutzten ICs können wir für Pfennigbeträge wiederbeschaffen und selbst in die Platine einsetzen.

Der nächste Nutzen der Ausgabeplatine: Sie zeigt den Schaltzustand der Datenleitung mit Hilfe der Leuchtdioden an. Dies ist bei dem Austesten der Roboterprogramme eine große Hilfe.

Bei Programmfehlern kann es auch zu unkontrollierten Bewegungen des Roboters kommen. Für diesen Fall besitzt die Ausgabe-Platine eine Not-Aus-Taste.

Zu guter Letzt haben wir auf der Platine eine einfache und für Experimente geeignete Steckerlösung vorgesehen. Die Ausgabeleitungen sind auf Schraubbefestigungen geführt. Verbindungskabel zwischen den Platinen und dem Roboter können so auf einfache Weise selbst hergestellt werden.

Schauen wir uns nun den Schaltplan der Ausgabeplatine in **Bild 2b** an. Ein Platinenlayout zusammen mit dem dazugehörigen Bestückungsplan finden Sie wie üblich auf den

blauen Seiten in diesem Heft. Die Signale von den acht Datenausgängen führen zunächst zu einem NAND-Baustein. Die jeweils anderen Eingänge der NAND-Bausteine sind alle in einer Freigabeleitung zusammengeschlossen. Im Ruhezustand liegt sie auf High, denn sie ist mit einem 3,3-k Ω -Widerstand gegen +5 V geschaltet. Das Signal der Datenleitung erscheint dann invertiert hinter dem NAND-Gatter.

Wird jedoch die Taste gedrückt oder eine externe Verbindung zwischen En (Enable = freigeben) und Masse hergestellt, so liegt die Freigabeleitung auf Low. In diesem Fall sind alle Ausgänge der NAND-Gatter auf High-Pegel, unabhängig von dem Schaltzustand der Datenleitungen. In der nächsten Stufe werden die Signale durch Leuchtdioden angezeigt. Die Leuchtdioden sind gegen +5 V geschaltet, d. h. sie leuchten, wenn der Ausgang des NAND-Gatters Low ist. Der Grund für diese Anordnung: die Ausgänge der verwendeten TTL-ICs schalten einen wesentlich größeren Strom im Low-Zustand gegen Masse als sie im High-Zustand abgeben können. Der Stromfluß durch die Leuchtdioden wird durch einen 270- Ω -Widerstand begrenzt.

Wenn wir uns an das oben Gesagte erinnern, sehen

wir auch, daß nach der Invertierung der Signale in den NAND-Bausteinen die Leuchtdioden gerade richtig herum anzeigen: Ein Datenausgang mit High-Pegel wird in Low umgewandelt und bringt die Leuchtdiode zum Leuchten. Umgekehrt erlischt die Leuchtdiode, wenn ein Low-Signal am Computer ausgegeben wird.

In der letzten Stufe der Ausgabeplatine werden die Signale nochmals invertiert. Damit besitzen sie wieder die gleiche Polarität wie am Ausgang des Computers. Doch nicht allein die Richtigstellung der Pegel gehört zu den Aufgaben der Inverter-Bausteine. Es handelt sich bei dem TTL-IC 7405 um einen Inverter mit offenem Kollektorausgang. Dies wird durch den auf der Linie stehenden Rhombus im IC-Symbol angezeigt.

Für diejenigen, denen ein offener Kollektorausgang noch kein vertrauter Begriff ist, folgende kurze Darstellung:

Der üblich Ausgang der TTL-Bausteinfamilie ist ein sogenannter Push-Pull-Ausgang (**Bild 3**). Liegt der Ausgang des ICs auf Low, ist der untere Transistor durchgeschaltet. Strom kann aus der nachfolgenden Schaltung über den Transistor gegen Masse abfließen (wie z. B. bei der Leuchtdiode oben). Liegt jedoch der Ausgang auf High, ist der obere Transistor durchgeschaltet. Strom kann aus der IC-Versorgung durch den Transistor in die nachfolgende Schaltung fließen. Sollte sich aber an dem Ausgang ein Kurzschluß gegen Masse ergeben, so wird das IC beschädigt.

Der offene-Kollektor-Ausgang enthält eine Ausgangsstufe nach **Bild 4**, die ihm offensichtlich den Namen gegeben hat. Wird dieser Ausgang auf Low geschaltet, so wirkt der Transistor wie beim TTL-Ausgang als Stromsenke. Im High-Zustand ist dagegen

der Transistor gesperrt. Der Ausgang ist hochohmig und kann durch den Strompfad über einen externen Widerstand auf High gelegt werden. Tritt an einem solchen Ausgang ein Kurzschluß gegen Masse auf, passiert nichts Schlimmes. Der Stromfluß wird durch den Widerstand begrenzt. (Sie werden sich fragen, warum solche kurzschlußfesten Ausgänge nicht durchwegs verwendet werden – die Antwort: mit ihnen lassen sich nicht für alle Zwecke genügend hohe Schaltgeschwindigkeiten erzielen.) Wenn Sie mitzählen, werden Sie feststellen, daß nicht alle zwölf Inverterstufen der beiden ICs 7405 ausgenutzt werden. Der Ein- und Ausgang eines freien Inverters wird auf die Anschlußleisten geführt (als E8 und A8 bezeichnet). Einen Inverter übrig zu haben, ist nicht zu verachten: Wie zuvor beschrieben, wird beim Commodore 64 eine zusätzliche Steuerleitung auf diesen Inverter geführt, um später mit der Eingabeplatine verwendet zu werden. Bei den Schneider Computern wird Ausgang 7, der invertiert erscheint, nochmal invertiert. Damit liegt er gerade wieder in der ursprünglichen Polarität vor.

Der Aufbau der Ausgabeplatine

Der Aufbau der Ausgabeplatine bietet keine größeren Schwierigkeiten. Lediglich bei den Leuchtdioden ist auf die Orientierung zu achten. Zur Kennzeichnung sind die Leuchtdioden meist auf einer Seite abgeflacht oder die Anschlußbeinchen sind verschieden lang. Die Katode der Leuchtdiode (dies ist die Seite, in deren Richtung der Pfeil verläuft) befindet sich auf der abgeflachten Seite bzw. an dem kürzeren Anschlußdraht. Alle Bauelemente sollten ganz in die Platine eingesteckt sein, so daß beim

Offene Ausgänge brauchen unbedingt einen Pull-up- Widerstand

Drücken auf die Bauelemente nicht die Leiterbahn auf der Unterseite der Platine abgehoben wird. Die ICs werden nicht in die Platine eingelötet, sondern IC-Sockel entsprechender Polzahl. Beim Einlöten der IC-Sockel werden diese zunächst provisorisch an zwei diagonal gegenüberliegenden Ecken verlötet. Durch Erhitzen dieser Lötstellen kann der Sitz des IC-Sockels nochmals korrigiert werden, bevor die restlichen Anschlußstifte verlötet werden. Gerade bei den IC-Sockeln muß gewährleistet sein, daß beim nachfolgenden Einrücken der ICs nicht die Leiterbahnen beschädigt werden. Als letztes werden die Leitungen zum Computerausgang eingelötet. Hierzu eignet sich ein Flachbandkabel, das ein kurzes Stück aufgetrennt und abisoliert wird. Die Drahtenden werden verdreht und verzinnt. So lassen sie sich leicht durch die Platine stecken, ohne daß ein struppiges Büschel entsteht. Bei einem Büschel ist immer die Gefahr, daß ein feines Drähtchen einen ungewollten Kurzschluß zu einem Nachbarn herstellt. An anderen Ende wird der zu dem Computerausgang passende Stecker angelötet.

Verpolungen müssen vor dem ersten Anlegen der Betriebsspannung unbedingt entfernt sein. Verbinden Sie nun die Ausgabeplatine mit der IC-Versorgung aus ELO 7/86. Wohl gemerkt, die ICs sind noch nicht bestückt und der Computer ist auch nicht angeschlossen. Messen Sie nun mit dem Vielfachinstrument die Spannung der IC-Versorgung nach. Sie darf nicht unter 4,9 V liegen. Außerdem darf keine der Leuchtdioden aufleuchten.

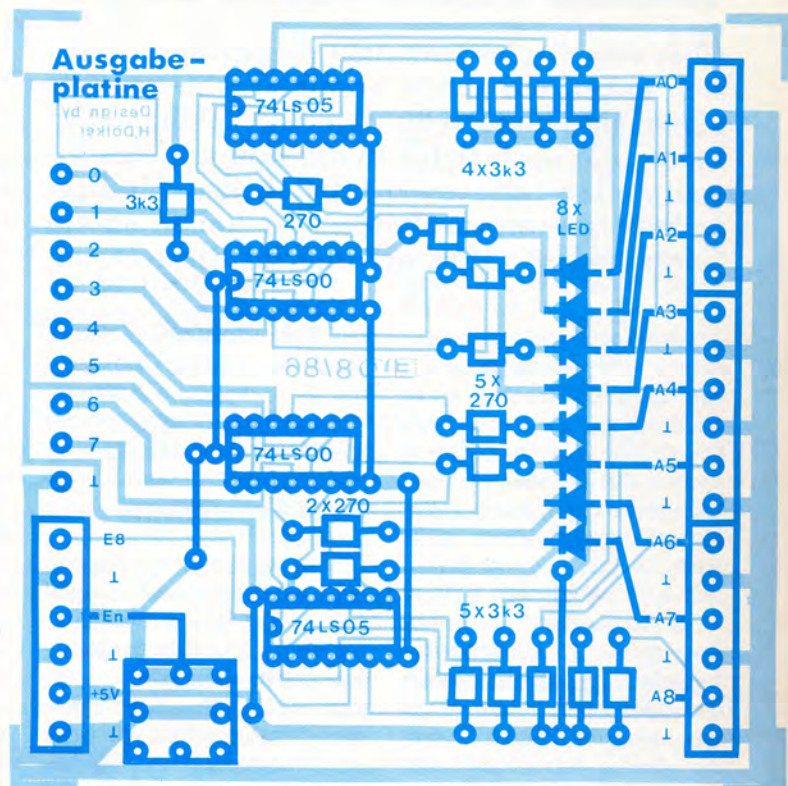
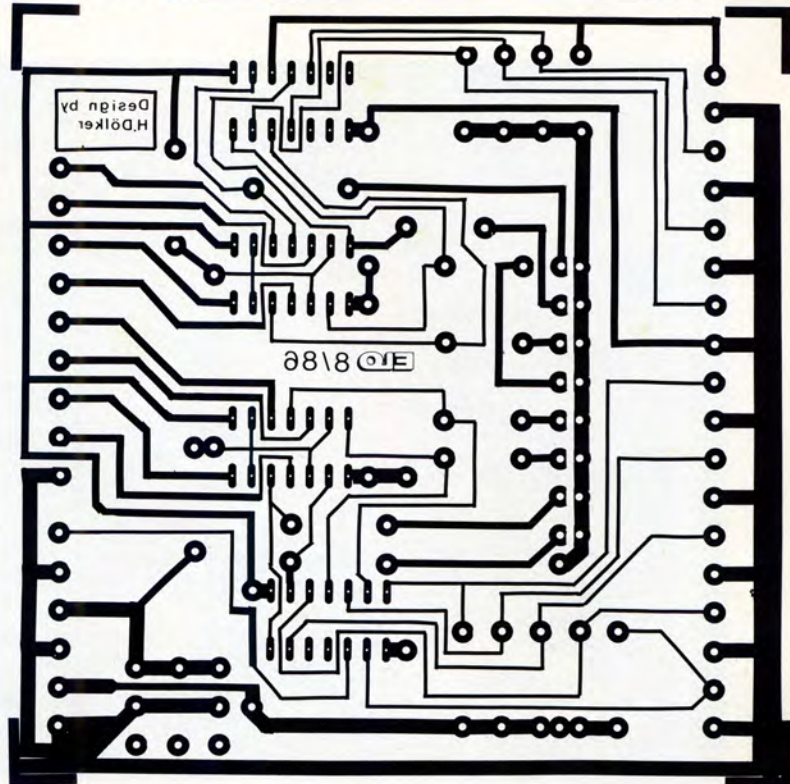
Ist dies dennoch der Fall, so klemmen Sie die Versorgungsspannung wieder ab und untersuchen die Platine erneut auf Kurzschlüsse. Messen Sie zur Sicherheit mit Hilfe des Meßinstruments die Stromaufnahme der Platine. Hierzu wird es in die 5-V-Zuleitung geschaltet. Sie sollten so gut wie keinen Stromfluß messen können.

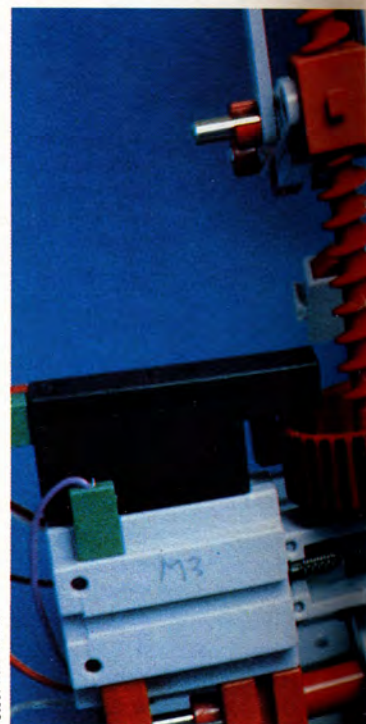
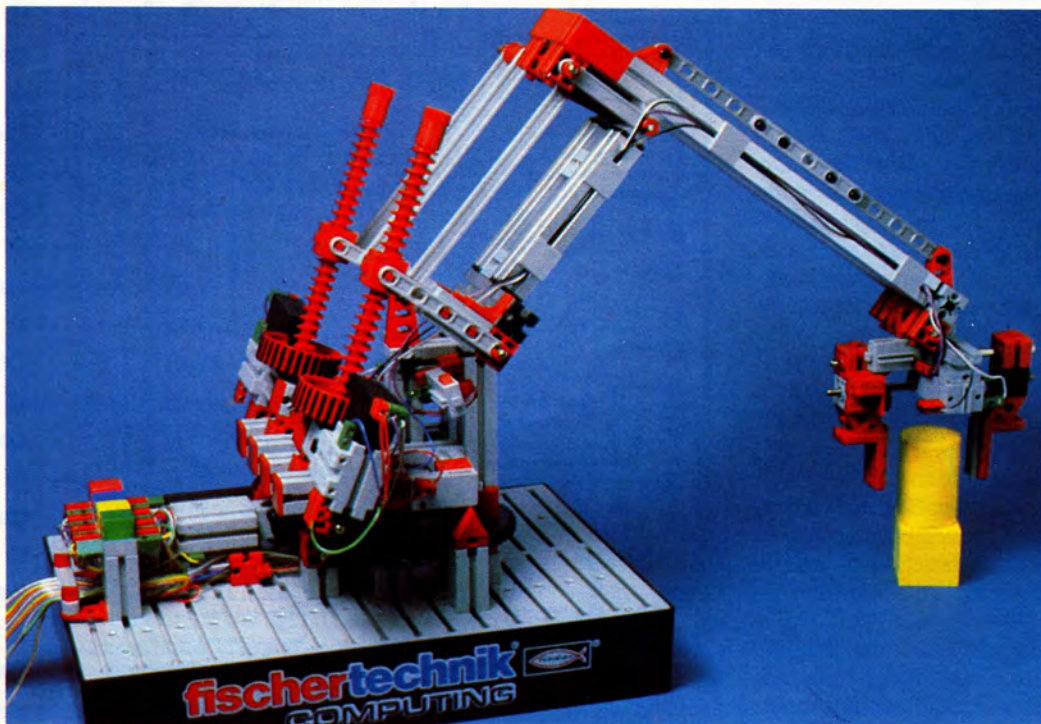
Wir nehmen an, daß Sie irgendwann, bei sorgfältiger Arbeit sogar sofort, das gewünschte Prüfergebnis erhalten. Nun können Sie die ICs in die ausgeschaltete Platine einsetzen. Achten Sie auf die Lage der Kerbe im IC-Gehäuse, die sich zwischen dem ersten und letzten Anschlußstift befindet. Einige Hersteller markieren auch den Anschlußstift 1 mit einem Punkt am Gehäuse. Legen Sie nun erneut die Betriebsspannung an. Eine Mahnung an die ganz Ungeduldigen: Der Computer steht noch immer nichtsahnend an der Ecke.

Dr. Roland Schulé

Inbetriebnahme der Ausgabeplatine

Zunächst werden die ICs noch nicht bestückt und die Leiterplatte einer gründlichen Sichtprüfung unterzogen. Kurzschlüsse, fehlende Kontakte und





Fotos: Ralf Siebler

Test: Fischertechnik-Trainingsroboter

Fischers starker Arm

Über den Fischertechnik-Computing-Baukasten haben wir schon berichtet. Ein überaus lehrreiches Spielzeug, wie wir meinen. Für Leute, die noch tiefer in die Robotik einsteigen wollen, gibt es zusätzlich den Fischertechnik-Trainingsroboter. Wir haben so ein Gerät einmal aufgebaut und für Sie getestet.

Am Anfang stand eine angenehme Überraschung: Auch der Trainingsroboter kann mit dem Fischertechnik Computing Interface, das es bereits für viele bekannte Computertypen gibt, angesteuert werden. Hier paßt wirklich eins zum anderen. Wie heute allgemein üblich, wird auch der Baukasten für den Trainingsroboter komplett geliefert. Neben den Bauteilen sind auch ein Flachbandkabel und die nötigen Anschlußstecker dabei. Auch ein klei-

ner Schraubenzieher wurde nicht vergessen. Und auch die Software zur Ansteuerung bekommt man von Fischertechnik.

Erste Experimente

Bevor es an den Aufbau geht, müssen erst einmal die ebenfalls mitgelieferten Infrarot-Reflexlichtschranken eingestellt werden. Aber keine Angst, wie man das macht, wird im Anleitungsbuch genau beschrieben. Dazu muß ein

kleiner Prüfstand aufgebaut und mit dem Interface verbunden werden. Mit Hilfe eines ebenfalls mitgelieferten Hilfsprogrammes ist dann das Einstellen ein Kinderspiel.

Problemloser Aufbau

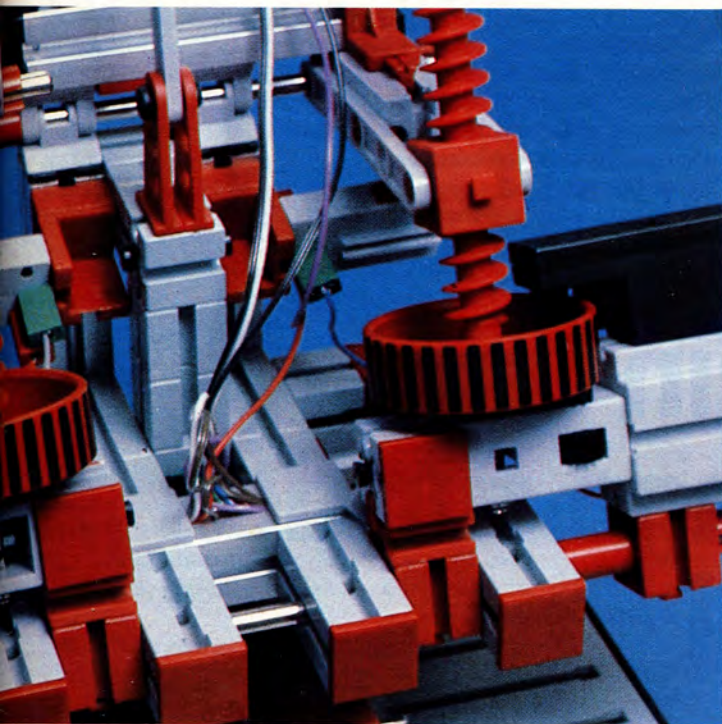
Danach geht's dann richtig los. Beim Aufbau kann eigentlich nicht viel schiefgehen, denn sämtliche Phasen werden im Anleitungsbuch anhand vieler Bilder genau beschrieben. Es empfiehlt sich allerdings, die Anleitung peinlichst genau zu befolgen, denn der Baukasten enthält keinen Stein zuviel. Unser Praktikant, den wir mit dem Aufbau betraut hatten, war schon halb fertig, als er einen Baustein benötigte, den er fälschlicherweise ganz unten in den Sockel eingebaut hatte. Daraufhin mußte er fast alles wieder auseinandernehmen. Aber auch wenn man solche Zwischenfälle mit einberechnet, dauert der Aufbau auch bei Fischertechnik-unerfahrenen Zeitgenossen wohl kaum länger als einen Nachmittag. Ein wenig Kraft ist allerdings manch-

Bild 1:
Mit Hilfe von Reflexlichtschranken und Impulsgeber-Scheiben findet der Roboter einprogrammierte Positionen immer wieder.

mal schon vonnöten, denn die stellenweise verwendeten Metallbausteine halten bombenfest.

Nur Verkabelung mit Hindernissen

Nun fehlt „bloß“ noch die Verdrahtung. Aber gerade dieser letzte Teil des Aufbaus hat es in sich. Zuerst müssen die einzelnen Adern des Flachbandkabels auf die richtige Länge zugeschnitten werden. Dabei sollte man die alte Elektriker-Weisheit im Hinterkopf behalten, daß nämlich zu lange Kabel sehr einfach kürzer, zu kurze Kabel aber nur sehr schwer wieder länger gemacht werden können. Achten Sie also bitte genau darauf, welches Kabel wie lang sein muß, und kontrollieren Sie lieber zweimal, bevor Sie ein Kabel zu kurz abschneiden. Die einzelnen Strippen des Flachbandkabels haben zwar verschiedene Farben, die Verdrahtungsanleitung



ist aber sinnigerweise in dezentem schwarz-weiß gehalten. Hier hätte man dem Anleitungsbuch doch wirklich etwas Farbe spendieren können.

Als letztes werden die einzelnen Kabelstränge mit den Motoren, Endschaltern, und Reflexlichtschranken verbunden (**Bild 1**). Auch dabei könnte das Anleitungsbuch hilfreicher sein. Die (Schwarz-Weiß-)Fotos sind etwas klein geraten und wenn man herausfinden will, welcher Anschluß auf dem Foto nun welchem Anschluß auf dem Modell entspricht, kann man sich oft eines gewissen Schwindelgefühles nicht erwehren. Hier empfiehlt es sich, die einzelnen Teile entsprechend dem Anleitungsbuch zu beschriften, sonst verliert man bald den Überblick. Für die Verbindungen wird ein ganzes Tütchen voller Stecker mitgeliefert. Was allerdings fehlt, ist ein Werkzeug zum Abisolieren der Drahtenden. Dafür sollte man sich schon eine Abisolierzange kaufen (oder ausleihen), Zähne, Fingernägel oder Muttis Küchenmesser sind hierfür nicht ge-

eignet. Und auch eine lange Pinzette wäre sehr hilfreich, denn im Anleitungsbuch steht zwar „Kabel durch den Drehkranz führen“, wieviele Finger man sich dabei abbricht, wurde allerdings nicht vermerkt! Es empfiehlt sich auch, alle verlegten Leitungen auf Durchgang zu prüfen. Dazu braucht man nicht unbedingt ein Meßgerät, ein einfaches Prüflämpchen tut's auch.

Viel Software

Wie schon beim Computing-Baukasten wird auch beim Trainingsroboter ein sogenanntes „Grundprogramm“ mitgeliefert. Es handelt sich dabei um eine Art Basic-Erweiterung. Das Programm stellt eine Reihe von Grundbefehlen für die Robotersteuerung zur Verfügung, mit denen man dann wie mit ganz normalen Basic-Befehlen frei programmieren kann. Daneben befindet sich auf der Diskette noch ein komplettes Teach-in-Programm. Mit Hilfe der Computertasten kann man den Arm in verschiedene Stellen bringen und die Posi-

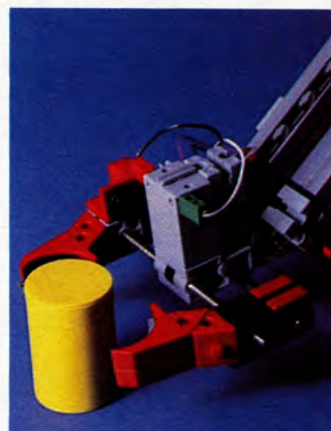
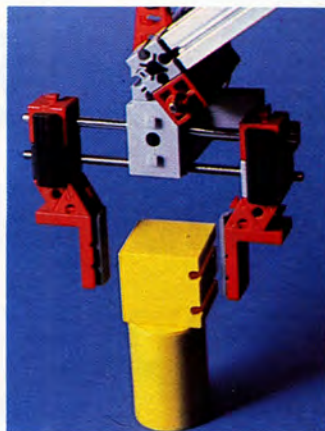
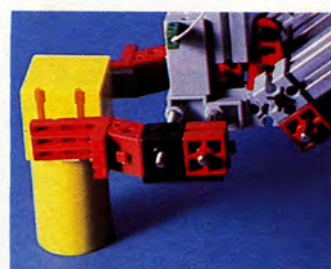


Bild 2...5:
Die Greifzange läßt sich leicht umbauen. Hier zeigen sich die Stärken des Systems.

tionen per Knopfdruck abspeichern. Über einen anderen Menüpunkt kann man den Roboter dann das Erlernte ausführen lassen. Obwohl der Roboter insgesamt einen etwas wackeligen Eindruck macht, trifft er die vorgegebenen Positionen ziemlich genau wieder. Die vom Hersteller versprochene Ansteuergenauigkeit von 1 mm wurde bei unseren Tests erreicht.

Ein brauchbares Experimentiergerät

Es lohnt sich wirklich, längere Zeit mit diesem Roboter herumzuexperimentieren. Das Handbuch gibt dafür viele wertvolle Hinweise. Im Umgang mit dem Roboterarm macht man sozusagen spielerisch wichtige Erfahrungen. Zum Beispiel die, daß der Roboter immer den „kürzesten Weg“ zwischen zwei einprogrammierten Punkten wählt (und dabei alles Dazwischenliegende gnadenlos abräumt), auch wenn man beim Teach-in

noch so schön „außen herum“ gefahren ist. Die Greifzange läßt sich sehr vielseitig umgestalten, so daß man die verschiedensten Gegenstände greifen kann (**Bild 2...5**).

Aber eins müssen wir klarstellen: Dieser Roboter ist und bleibt ein Spielzeug. Bitte verwenden Sie für Ihre Versuche nur sehr leichte Gegenstände, wie zum Beispiel Streichholzschachteln oder Plastik-Bauklötze. Wir haben den Roboter versuchsweise einmal einen Telefonhörer abheben lassen. Das hat er mit Mühe und Not geschafft, sich dabei aber so bedrohlich verbogen, daß wir von diesem Versuch dringend abraten müssen. Was Sie aber unbedingt versuchen sollten, ist, mit Hilfe des Grundprogrammes eine eigene Roboter-Steuerung zu programmieren. So könnte man z. B. dem Roboter beibringen, Schachfiguren richtig auf dem Brett zu bewegen. Und auch das Teach-in-Programm ließe sich noch wesentlich komfortabler gestalten – eine Herausforderung für den Programmierfreak.

Ralf Siebler

Fortsetzung von Seite 13

darüber hinaus A3 und A4 für das Netzwerk und das Microdrive-Laufwerk reserviert. Bleiben also noch A5, A6 und A7, die wir jedoch alle zur Adressierung der vier Register der Daten-Drehscheibe benutzen. Mit A5 = Low wählen wir zunächst die Baugruppe an. A6 und A7 lassen vier Kombinationen zu, die den vier Registern des Bausteins zugeordnet werden.

Zum Anschluß der Daten-Drehscheibe besorgen Sie sich den jeweils passenden Busstecker (Direktverbinder im Rastermaß 2,54 mm) für den ZX-81 bzw. ZX-Spectrum. Die Busbelegung ist dem Bild zu entnehmen. Zwar sind die Busanschlüsse von ZX81 und Spectrum unterschiedlich, nicht jedoch für die wenigen Signale, die wir benötigen. Der Stecker paßt also auf beide Computer. Die Leitung Φ des ZX-81 ist in dem Handbuch des Spectrum mit CK bezeichnet. Gemeint ist jedoch beide mal das Prozessor-Taktsignal. Achten Sie auch darauf, die Adreßanschlüsse Ad0 bis Ad7 (in der Zeichnung links) nicht mit den Portanschlüssen A1 bis A8 (rechts) zu verwechseln. Die Daten-Drehscheibe wird übrigens aus dem Netzteil des ZX mitversorgt. Den verschiedenen Platinen der Robotik-Serie sollten Sie aber ihre eigene Versorgung gönnen.

Wer mehr Informationen über den Aufbau und die Arbeitsweise des Z80-PIO-Bausteins in der Daten-Drehscheibe erhalten will, möge bitte in den Funkschau-Veröffentlichungen nachschlagen. Wir werden hier nur kurze Kochrezepte geben, wie sie in unserer Robotik-Anwendung vonnöten sind. Der Baustein hat zwei, jeweils acht Bit umfassende Datenschnittstellen (Ports), die mit A und B (auch auf der Platine) bezeichnet sind. Entgegen der Konvention in diesem Artikel sind die Bits

auf der Daten-Drehscheibe mit 1 bis 8 und nicht mit 0 bis 7 bezeichnet. Behalten Sie dies beim Experimentaufbau im Auge. Die Ein- und Ausgänge zwischen den Ports, die mit RDY, STB usw. bezeichnet sind, benötigen wir nicht. Port A ver-

wenden wir zur Ausgabe. Die Leitungen A1 bis A8 werden mit den Eingängen 0 bis 7 zusammengeschlossen. Außerdem werden die Masseanschlüsse miteinander verbunden. Die Eingabeplatine wird an Port B angeschlossen. Dabei

wird folgende Zuordnung getroffen:

0	- B1
1	- B2
2	- B3
3	- B4
PY	- B7
PX	- B8

Mehr Ein- und Ausgänge für den Apple II

Die Nachfrage nach mehr Ein- und Ausgabeleitungen an dem Apple II ist verständlich, sind doch drei Eingabe- und vier Ausgabeleitungen nicht gerade viel und nur für die einfachsten Experimente ausreichend. Es empfiehlt sich daher eine Ein-/Ausgabekarte für einen der Apple-Slots. Solche Karten gibt es von mehreren Anbietern; meist basieren diese Karten auf dem IC 6522, das auch den Userport von etlichen anderen Computern (z. B. VC20) bildet. Da der Hardware-Aufwand nicht allzu groß ist, können Sie auch durchaus selbst dieses IC auf einer Apple-Experimentierkarte aufbauen, siehe Bild.

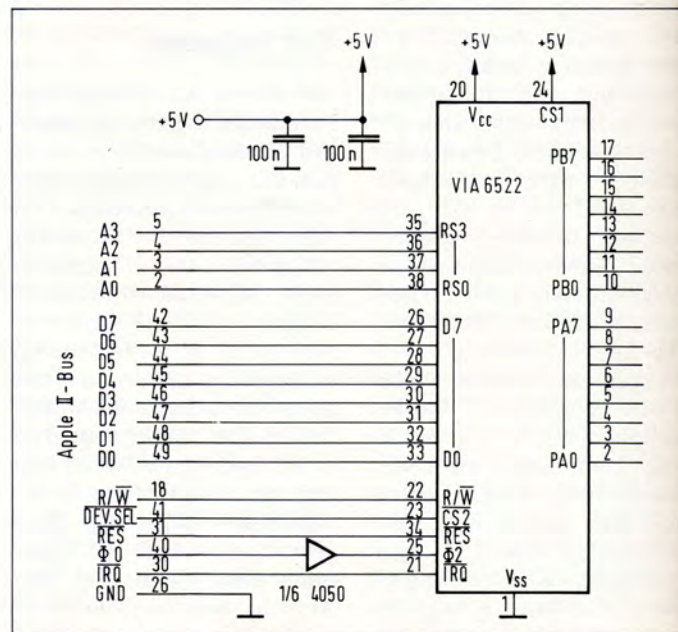
Der Baustein besitzt zwei 8-Bit-Ports, die als PA und PB bezeichnet werden. Verwenden Sie z. B. Port A (Bit 0-3) zum Anschluß der Eingabeplatine und Port B (0-7) zum Anschluß der Ausgabeplatine. Bei dem gezeigten Aufbau und bei Verwendung in Slot 4 ergeben sich folgende Adressen zur Programmierung:

Die beiden Analogeingänge

```
10 A=49344 : REM PORT B ALS AUSGABEPORT
20 POKE A+2,555 : REM DATENRICHTUNG
30 E=49345 : REM PORT A ALS EINGABEPORT
40 POKE E+2,0 : REM DATENRICHTUNG
50 INPUT „AUSGABEWORT ?“;X
60 POKE A,X
70 PRINT „EINGABEWORT =“;PEEK(E)
```

sollten jedoch nach wie vor an dem Game-Controller Port verbleiben, denn es ist nicht nötig, den Aufwand der Analog-Digital-Wandlung nochmals auf der Slot-Karte zu betreiben.

Sie sehen also, daß mit einer solchen Karte die Programmierung recht analog zu anderen 6502-Computern, wie z. B. C64, verläuft und die Programme daher leicht angepaßt werden können.



Die Ein-/Ausgabekarte besitzt zwei 8-Bit-Ports, PA und PB.

Mit der Netzteil-Platine hörten wir letztes auf, noch bevor sie zu Ende getestet war. Bei dem jetzigen Test sollten alle Leuchtdioden aufleuchten. Die Stromaufnahme darf gut und gerne bis zu 120 mA betragen. Sollten Sie nicht zu dem gewünschten Ergebnis kommen, so schalten Sie sofort wieder die Betriebsspannung aus und suchen den Fehler.

Als nächsten Test drücken Sie die Not-Aus-Taste. Alle Leuchtdioden sollten mit einem Schlag ausgehen. Auch eine Drahtbrücke nach Masse an dem externen Freigabeeingang muß den gleichen Effekt zeigen.

Schließen Sie nun an einem Masseanschluß ein längeres Stück Schaltendraht an. Suchen Sie an dem Stecker für den Computeranschluß die Lage der acht Datenleitungen und berühren Sie sie mit dem blanken Drahtende. Die jeweils dazugehörige Leuchtdiode muß erlöschen, solange der Eingang Massekontakt hat. Als letzten Test schließen Sie ein Voltmeter der Reihe nach an die acht Ausgänge der Platine an. Im Ruhezustand müssen Sie ca. 5 V messen, die IC-Versorgungsspannung. Wenn die Not-Aus-Taste gedrückt wird, muß die Spannung auf ca. 0,3 V abfallen. Auch hier gilt wieder, daß jeder entdeckten Unregelmäßigkeit erst nachgegangen und die Ursache behoben werden muß, bevor Sie zu dem nächsten Schritt weitergehen.

Test der Ausgabeplatine

Nun ist es endlich soweit! Schalten Sie die Ausgabeplatine ebenso wie den Computer und möglichst auch noch die angeschlossenen Peripheriegeräte aus. Verbinden Sie die Ausgabeplatinen und den Computer. Schalten Sie nun den Computer wieder ein. Er sollte ganz normal

reagieren und sich mit seinem gewohnten Titel zur Stelle melden. Nun können Sie auch die IC-Versorgung an die Ausgabeplatine anschalten. Auch dies darf den Computer zu keiner besonderen Aktion veranlassen. Bei den meisten Computern werden die Leuchtdioden der Ausgabeplatine aufleuchten. Geben Sie nun folgendes ein:

Commodore 64 und 128:

A=56577

Commodore VC20:

A=37136

Commodore PETs und CBMs:

A=59457

CPC 464, 664 und 6128:

A=61439

Bei den Commodore Computern geben Sie zusätzlich ein:

POKE A+2,255

Nun wollen wir unsere erste Steuerung vornehmen:

Commodore Computer:

POKE A,0

Schneider Computer:

OUT A,0

Die unterschiedliche Schreibweise ist leider notwendig. Schneider-Computer, wie alle anderen Computer, die auch auf dem Mikroprozessor Z80 aufbauen, sprechen die Peripheriegeräte mit einem eigenen Kommando an, eben dem OUT-Kommando. Commodore Computer teilen mit anderen Computersystemen auf der Basis des 6502-Mikroprozessors die Eigenschaft, Peripheriegeräte wie Speicheradressen mit dem POKE-Kommando anzusprechen. Beides hat seine Vor- und Nachteile, die jedoch in diesem Zusammenhang nicht von Bedeutung sind. Zurück zu unserem Experiment. Das obige Kommando muß bewirkt haben, daß spätestens jetzt alle Leuchtdioden erloschen sind. Schalten wir sie mit dem folgenden Kommando wieder ein:

Commodore Computer:

POKE A,255

Schneider Computer:

OUT A,255

Nachdem wir die Ausgabeplatine soweit erfolgreich



Einführung in die Robotik:

Ampelsteuerung

Mit den Angaben aus den bis dato absolvierten zwei Folgen der Robotik-Serie konnten Sie immerhin schon die Netzteil- und die Ausgabe-Platine aufbauen. Bevor es weitergeht, werden wir die beiden gründlich testen und uns vor einer weiteren Anwendung an eine neue Platine heranwagen.

in Betrieb genommen haben, wollen wir sie nun detaillierteren Tests unterziehen. Insbesondere müssen wir testen, ob die Reihenfolge der Datenleitungen – z. B. beim Herstellen des Computeranschlußkabels – gewahrt wurde. Dazu müssen wir die Datenleitungen einzeln ansteuern. Auch die Druckerschnittstelle besitzt acht Bits und sie entsprechen gerade den erwähnten acht Datenleitungen. Obige OUT- bzw. POKE-Befehle wandeln

den nach der Adresse A angegebenen (dezimalen) Zahlenwert in eine Binärzahl um und lassen sie als ein Muster von High- und Low-Signalen auf der Ausgabeplatine erscheinen. Da die meisten Basic-Interpreter nur mit Dezimalzahlen arbeiten, sollten wir uns in den obigen Zahlenkonstanten etwas einüben; es sind gerade jene, bei denen eine einzige Datenleitung gesetzt wird. Geben wir der Reihenfolge nach die Kommandos:

POKE A,1 bzw. OUT A,1
POKE A,2 OUT A,2
POKE A,4 ...
:
:

POKE A,128 OUT A,128
Der Reihenfolge nach müssen die Leuchtdioden von Ausgang 0 bis Ausgang 7 aufgeleuchtet haben. War dies nicht der Fall, überprüfen Sie die Leitungsführung zwischen Computeranschluß und Ausgabeplatine.

Dualzahlentrainer

In der Praxis werden wir jedoch meist mehr als nur ein Bit zur gleichen Zeit

Bildschirm aus. Sie müssen nun die dazugehörige Dezimalzahl an der Tastatur eingeben. Vergleichen Sie das an der Ausgabeplatine entstehende Bitmuster mit jenem auf dem Bildschirm. Das Programm sagt Ihnen auch noch, ob sie recht gehabt haben.

Stark soll er sein

Die Ausgänge der TTL-ICs reichen zwar zum Betrieb von Leuchtdioden, nicht aber von Motoren, Lampen, Relais und Magneten, all jenen Akteuren von Robotern und Automaten. Wir müssen uns daher mit dem Problem befassen, wie wir die Signale der Ausgabeplatine geeignet verstärken. Für viele Zwecke genügt ein einfacher Schaltverstärker, wie er in **Bild 2** gezeigt ist. Der Transistor ist ein Darlingtontistor, der einen sehr hohen Verstärkungsfaktor besitzt, da er aus einem Doppeltransistorsystem besteht. Der Verbraucher wird zwischen die positive Versorgungsspannung und den Kollektor des Transistors geschaltet. Wenn an den Eingang der Schaltung, der übrigens zum Schutze der steuernden Schaltung einen Begrenzungswiderstand enthält, ein High-Signal angelegt wird, so steuert der Transistor durch. Über der Kollektor-Emitter-Strecke bleibt dann selbst bei Stromstärken von ca. 1 A nur eine Spannung von ca. 0,7 V stehen. Dies dürfte für die meisten Modellzwecke vollständig ausreichend sein.

Da wir mit dieser Platine meist Motoren und andere induktive Lasten steuern werden, müssen wir noch einige Besonderheiten beachten. Wenn das Steuersignal auf Low geht, wird der Transistor gesperrt. In dem angeschlossenen (induktiven) Verbraucher hat der Stromfluß entsprechend dem Induktionsgesetz die Tendenz, weiterzufließen. Da der Transistor

jedoch gesperrt ist, entsteht über ihm eine hohe (Induktions-)Spannung, die leicht ein Vielfaches der ursprünglichen Versorgungsspannung betragen kann. Die Induktionsspannung wird vielleicht sogar die zulässige Kollektor-Emitter-Sperrspannung übersteigen, so daß der Transistor beschädigt wird. Zum Schutze des Transistors sieht die Schaltung eine sogenannte Freilaufdiode vor, die die Induktionsspannung kurzschließt. Auf den normalen Betrieb hat sie jedoch keinen Einfluß, da sie dann in Sperrrichtung liegt.

Die Stromversorgung für die Verbraucher unterliegt starken Belastungen. Durch den Anlaufstrom von Motoren wird sie Einbrüche haben. Die trotz Freilaufdiode noch verbleibende Induktionsspannung führt zu Spannungsspitzen. Mit anderen Worten: die Versorgungsspannung ist recht „unsauber“. Daher darf selbst bei Verbrauchern, die ca. 5 V Versorgungsspannung benötigen, die Versorgung nicht aus der IC-Versorgung entnommen werden. Sicherheitshalber ist immer eine getrennte Versorgung vorzusehen. Die Versorgungs-

Stückliste Ausgabeplatine

Halbleiter

- 2 IC 74LS00
- 2 IC 74LS05
- 8 Leuchtdioden, 3 mm Ø

Widerstände

- 8 270 Ω, 0,25 W
- 10 3,3 kΩ, 0,25 W

Sonstiges

- 1 Taster, 2poliger Umschalter (Shadow)
- 4 Anschlußklemmen, 6polig
- 1 Platine, ELO 9/1986
- Kabel und Anschlußstecker entsprechend Computersystem

setzen müssen. In diesem Fall werden die Stellenwerte aller Bits, die gesetzt sein sollen, aufaddiert. Diese Zahl wird nun per POKE- oder OUT-Befehl an die Ausgabeplatine weitergegeben. Ein Beispiel: Ausgang 0, Ausgang 1 und Ausgang 4 sollen gesetzt, alle anderen gelöscht sein. In diesem Fall wird 1 (2⁰), 2 (2¹) und 16 (2⁴) aufaddiert.

Das Kommando:
POKE A,19 bzw.
OUT A,19

liefert das gewünschte Ergebnis. Bitte ausprobieren! Um uns in dieser Materie zu üben, eignet sich das in **Bild 1** abgebildete Programm. Es erzeugt zufällige 8-Bit-Binärrzahlen und gibt das Bitmuster am

```
10 REM DUALZAHLENTAINER I
20 REM
30 REM DAS PROGRAMM ZIEHT ZUFALLSZAHLN
40 REM ZWISCHEN 0 UND 255, DIE IN DUAL-
50 REM ZAHLEN GEWANDelt UND AUF DEM
60 REM SCHIRM DARGESTELLT WERDEN. IN
70 REM DER NACHFOLGENDEN EINGABE MUSS
80 REM DIE ENTSPRECHENDE DEZIMALZAHL
90 REM ANGEZEIGT WERDEN.
100 :
110 A=56577 :REM AUSGABEKANAL
120 POKE A+2,255 :REM NUR COMMODORE
130 GOSUB 500 :REM ZUFALLSZAHL ZIEHEN
140 GOSUB 600 :REM DUALZAHL ERMITTELN
150 GOSUB 700 :REM BILDSCHIRMAUSDRUCK
170 INPUT "DIE ENTSPRECHENDE DEZIMALZAHL";D
180 IF D<0 THEN GOTO 220
190 IF D>255 THEN GOTO 220
200 POKE A,D :REM KONTROLLAUSGABE
210 IF D=Z THEN PRINT "RICHTIG!"
220 IF D<>Z THEN PRINT "LEIDER FALSCH."
230 GOSUB 800 :REM WARTESCHLEIFE
240 GOTO 110
250 :
500 REM UNTERPROGRAMM ZUFALLSZAHL
510 Z=INT(256*RND(1))
520 IF Z>255 THEN GOTO 510 :REM BEREICH
530 RETURN
540 :
600 REM UNTERPROGRAMM DUALZAHLZERLEGUNG
610 Z2=Z : Z$=""
620 FOR BIT=7 TO 0 STEP -1
630 BW=INT(Z2/2^BIT) :REM BINAERWERT
640 Z2=Z2-2^BIT*BW
650 IF BW=1 THEN Z$=Z$+"1"
660 IF BW=0 THEN Z$=Z$+"0"
670 NEXT BIT
680 RETURN
690 :
700 REM UNTERPROGRAMM BILDSCHIRMAUSDRUCK
710 PRINT CHR$(147):REM SCHIRM LOESCHEN
720 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT
730 PRINT "          ";Z$
740 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT
750 RETURN
800 REM UNTERPROGRAMM WARTESCHLEIFE
810 FOR ZEIT = 0 TO 1000
820 NEXT ZEIT
830 RETURN
READY.
```

Bild 1a: Programm Dualzahlentrainer für Commodore 64.

spannung für die Verbraucher kann bei dem gewählten Transistortyp bis zu 30 V betragen. Achten Sie darauf, die Verbraucherversorgung und die IC-Versorgung sauber getrennt zu halten. Eine Verbindung beider führt unvermeidlich zu dem Ende aller angeschlossenen ICs.

Aufbau und Test der Verstärkerplatine

Der Aufbau der Verstärkerplatine bereitet weiter keine Schwierigkeiten. Die Anschlußbeine der Transistoren winkeln Sie mit einer Flachzange ab, so daß die mit dem Kollektor ver-

bundene Metallplatte auf der Leiterplatte zu liegen kommt (Bild 3). Wenn Sie höhere Stromstärken als ca. 1 A voraussehen, sollten Sie den Transistoren Kühlkörper spendieren. Nach dem Aufbau nehmen Sie eine gründliche Sichtprüfung der Platine vor. Achten Sie insbesondere auf Kurzschlüsse und unsaubere Lötstellen. Zum Test verbinden Sie die Ausgabeplatine mit der Verstärkerplatine und der IC-Versorgung (Bild 4). Der Computer ist aus Sicherheitsgründen nicht angeschlossen! Ebenso auch die Verbraucherversorgung.

```
10 REM Dualzahlentrainer CPC
20 REM
30 REM Das Programm zieht Zufallszahlen
40 REM zwischen 0 und 255, die in Dual-
50 REM zahlen gewandelt und auf dem
60 REM Schirm dargestellt werden. In
70 REM der nachfolgenden Eingabe muss
80 REM die entsprechende Dezimalzahl
90 REM angegeben werden.
100 :
110 a=61439 :REM Ausgabekanal
120 GOSUB 500 :REM Zufallszahl ziehen
130 GOSUB 600 :REM Dualzahl ermitteln
140 GOSUB 700 :REM Bildschirmausdruck
150 INPUT "Die entsprechende Dezimalzahl";d
160 IF d<0 THEN GOTO 220
170 IF d>255 THEN GOTO 220
180 OUT a,d :REM Kontrollausgabe
190 IF d=z THEN PRINT "Richtig!"
200 IF d<z THEN PRINT "Leider falsch."
210 GOSUB 800 :REM Warteschleife
220 GOTO 110
230 :
240 REM Unterprogramm Zufallszahl
250 z=INT(256*RND(1))
260 IF z>255 THEN GOTO 250 :REM Bereich
270 RETURN
280 :
290 REM Unterprogramm Dualzahlzerlegung
300 z$=BIN$(z,8) :REM dualzahl
310 RETURN
320 :
330 REM Unterprogramm Bildschirmausdruck
340 CLS :REM Schirm loeschen
350 LOCATE 15,7
360 PRINT z$
370 LOCATE 1,15
380 RETURN
390 :
400 REM Unterprogramm Warteschleife
410 FOR zeit=0 TO 2000
420 NEXT zeit
430 RETURN
```

Bild 1b: Programm Dualzahlentrainer für Schneider CPC.

Dualzahlen

Sicher werden Sie schon zur Genüge gehört haben, daß Computer auf der Basis einer binären Darstellung der Zahlen arbeiten. Wir fassen uns deswegen kurz: In einem binären Zahlensystem werden nur zwei Ziffern verwendet, die Null und die Eins. Eine Null stellen wir in den Schaltungen des Computers und auch an der Parallelschnittstelle durch ein Low-Signal dar, eine Eins durch ein High-Signal. Um zu größeren Zahlen zu kommen, verwendet man ein Stellenwertsystem, wie bei dem gewohnten Zehnersystem auch. Bei jedem Weiterücken nach links in der Zahlendarstellung erhält die Zahlenstelle eine höhere Wertigkeit. Beim Zehnersystem unterscheiden sich die Zahlen 1, 10, 100 usw. jeweils durch den Faktor zehn; beim Binärsystem unterscheiden sich die Zahlen 1, 10, 100 jeweils um den Faktor zwei. Um keine Verwechslungen aufkommen zu lassen, sollte das Zahlensystem gekennzeichnet sein. Entspre-

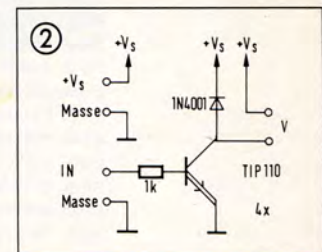
chend der Übereinkunft bei vielen Assemblerversionen stellen wir der Binärschreibweise ein B nach. Wenn es die Klarheit erforderlich macht, wird eine Zahl in dem normalen Zehnersystem durch ein nachgestelltes D (für dezimal) gekennzeichnet. Also:

1B = 1D
10B = 2D
100B = 4D
1000B = 8D
10 000B = 16D
100 000B = 32D
1 000 000B = 64D
10 000 000B = 128D

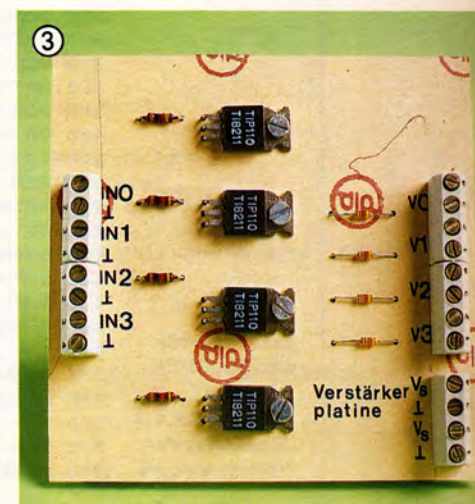
Wir haben gerade bei der achtstelligen Binärzahl aufgehört, weil die meisten Heimcomputer auch in dieser „Verpackung“ Zahlen hantieren. Acht Binärstellen, oder wie man auch dazu sagt, acht Bits werden zu einer Einheit zusammengefaßt und auf einmal, d. h. parallel, zwischen den Computerbausteinen ausgetauscht. Größere Zahlenwerte erfordern ein Nacheinander von Bytes, z. B. zwei Bytes für eine Integer-Zahl und fünf Bytes für eine Real-Zahl der üblichen Basic-Interpreter.

Bild 2: Die Verstärkerplatine liefert die Kraft, die ein Roboter braucht.

Bild 3: Der Testaufbau für die Verstärkerplatine. Der Computer ist noch nicht angeschlossen.



Wenn Sie nun die IC-Versorgung einschalten, springen alle Ausgänge der Ausgabeplatine auf High, wie man leicht an den Leuchtdioden erkennen kann. Dies haben wir schon beim letzten Mal kennengelernt. Die vier angeschlossenen Verstärker sollten nun alle durchgesteuert sein, allerdings sind sie noch nicht mit Spannung versorgt. Wir schließen nun ein Voltmeter an der Verbraucherversorgung an und beobachten es, wenn wir diese Versorgungsspannung an



die Verstärkerplatine (Anschluß Vs) anlegen. Es darf zu keinem merklichen Einbruch der Spannung kommen. Sie sollten dieses Ergebnis durch eine Temperaturkontrolle an den Transistorgehäusen bestätigen (vorsichtiges Berühren mit dem Zeigefinger genügt!).

Klemmen Sie nun das Voltmeter an einen der Ausgänge an; es sollte nahezu die Spannung der Verbraucherversorgung anzeigen. Betätigen Sie nun die Not-Aus-Taste auf der Ausgabeplatine. Da nun alle Aus-

Grünes Licht für die erste Computersteuerung

Nun ist es soweit: Eine Verkehrsampel soll mit Hilfe unserer Erstausrüstung an Platinen gesteuert werden. Für den Aufbau der Verkehrsampel verwenden wir vorteilhafterweise die Teile aus dem Fischertechnik Computing Baukasten (Bild 5). Die dort enthaltenen Bausteine und Baugruppen können uns auch noch später, wenn wir zu anspruchsvolleren Instrumenten kommen, die notwendige Ausrüstung in der Mechanik liefern. Der Aufbau der Modelle ist außerdem in einer bebilderten Anleitung beschrieben. Die Teile werden nur durch Zusam-

menschieben miteinander verbunden. Damit entfallen langwierige Konstruktionsarbeiten und Vermaßungen sowie der anschließende Umgang mit Säge, Bohrmaschine und Kleber (von dem wir allerdings niemanden, der sich dazu berufen fühlt, abhalten wollen!).

Dem Baukasten sind drei Lampensockel beigelegt, die an dem Ampelmast befestigt werden. Wenn die kleinen Glühbirnchen eingesteckt sind, werden die farbigen Lampenabdeckungen aufgesetzt. Oben kommt die rote, in der Mitte die gelbe und unten die grüne Abdeckung hin. Die drei Lampensockel werden mit der Verstärkerplatine verbunden. Dazu kann das in dem Baukasten enthal-

tene Flachbandkabel verwendet werden. In diesem Fall muß aber der zwanzigpolige Stecker abgeschnitten werden. Alternativ kann auch zweiadriges Litze mit einem Querschnitt von 0,14 mm² eingesetzt werden. Solche Kabel sind im Elektronikhandel erhältlich aber auch zum Verkabeln von Modelleisenbahnanlagen üblich. Für die Fischertechnik-Komponenten werden die Kabel mit den im Kasten beigelegten Flachsteckern ausgestattet, wie in der dortigen Anleitung beschrieben. Das andere Ende des Kabels wird, wie schon bei den Verbindungsleitungen, abisoliert, verzinkt und mit den Klemmen der Verstärkerplatine verschraubt. Die



```

100 4. AUFBAU DER VERKEHRSAMPEL
110 R. SCHULE, 1986
120 FUNKTION: ZYKLISCHER DURCHLAUF
130 REM DURCHLAUF DER AMPELPHASEN GRUEN
140 REM GELB, ROT UND ROT-GELB.
150 REM
160 REM EINGANG DER AUSGÄNGE:
170 REM
180 REM GANG 0 - GRUENE LAMPE
190 REM GANG 1 - GELBE LAMPE
200 REM GANG 2 - ROTE LAMPE
210 REM GANG 3 BIS 7 NICHT BENUTZT.
220
230
240 REM TITELMELDUNG
250 POKE 53281,1 :REM HINTERGRUND WEISS
260 PRINT CHR$(147); :REM BILD LOESCHEN
270 PRINT CHR$(31); :REM SCHRIFT BLAU
280 PRINT : PRINT
290 PRINT "VERKEHRSAMPEL";
300 GOSUB 910:REM DRUCKE GRAFIKSYMBOLE
310 DATA 7,3,28,113,31
320 PRINT "-----";
330 GOSUB 910
340 DATA 7,2,158,113
350 GOSUB 910
360 DATA 20,2,30,113
370 GOSUB 910
380 DATA 20,2,152,98
390 GOSUB 910
400 DATA 20,6,98,17,17,17,17,31
410 :
420 REM SCHALTDauern EINGEBEN
430 PRINT "EINGABE DER SCHALTDauern : "
440 PRINT "1 SEKUNDE IST UNGEFAEHR 500"
450 PRINT
460 INPUT "DAUER DER GRUEN-PHASE";GRUEN
470 INPUT "DAUER DER GELB-PHASE ";GELB
480 INPUT "DAUER DER ROT-PHASE ";ROT
490 INPUT "DAUER DER R./G.-PHASE";RG
500 PRINT
510 PRINT "PROGRAMMSTOP MIT STOPTASTE"
520 :
530 REM AUSGABEKANAL BESTIMMEN
540 A=56577 :REM BENUTZERSCHNITTSTELLE

```

```

550 POKE A+2,255 :REM AUSGABESTEUERUNG
560 :
570 REM GRUENPHASE
580 ZEIT=GRUEN
590 MUSTER=1
600 GOSUB 810:REM AUSGABE
610 :
620 REM GELBPHASE
630 ZEIT=GELB
640 MUSTER=2
650 GOSUB 810:REM AUSGABE
660 :
670 REM ROTPHASE
680 ZEIT=ROT
690 MUSTER=4
700 GOSUB 810:REM AUSGABE
710 :
720 REM ROT/GELB-PHASE
730 ZEIT=RG
740 MUSTER=6
750 GOSUB 810:REM AUSGABE
760 :
770 REM RUECKSPRUNG
780 GOTO 560
790 :
800 REM AUSGABEROUTINE
810 POKE A,MUSTER :REM AUSGABE
820 FOR Z=0 TO ZEIT
830 NEXT Z
840 RETURN
850 :
860 REM GRAFIKSYMBOLE AUSDRUCKEN
870 REM AUFBAU DER DATAZEILE:
880 REM LZ=ZAHL FUEHRENDER LEERZEICHEN
890 REM GZ=ZAHL DER GRAFIKSYMBOLE
900 REM GZ GRAFIKSYMBOLE (CHR$(CODE))
910 READ LZ : PRINT SPC(LZ);
920 READ GZ
930 FOR I=1 TO GZ
940 READ GC : PRINT CHR$(GC);
950 NEXT I
960 PRINT
970 RETURN
READY.

```

Bild 7a (C-64) und 7b (CPC): Basic-Programme zur Steuerung der Verkehrsampel.

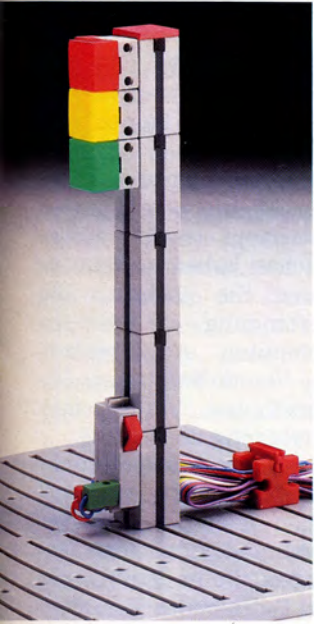


Bild 4:
Modell einer Verkehrsampel
aus dem Fischertechnik-
Computing-Baukasten.

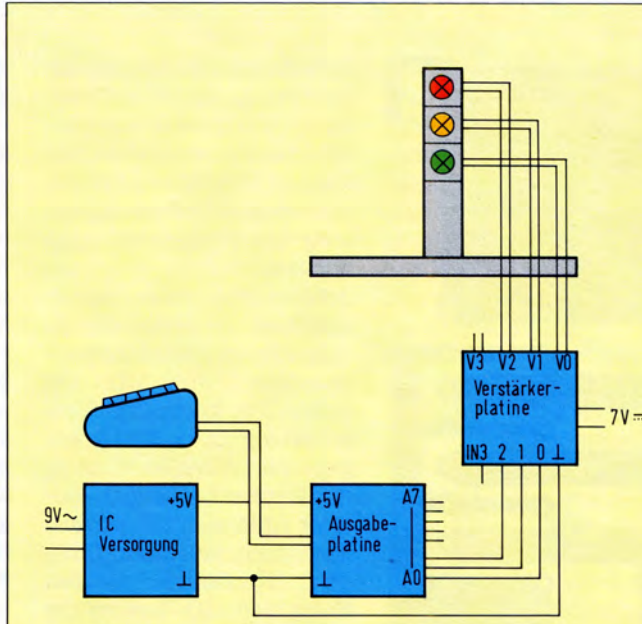


Bild 5:
Schaltungsaufbau zur Verkehrsampel.

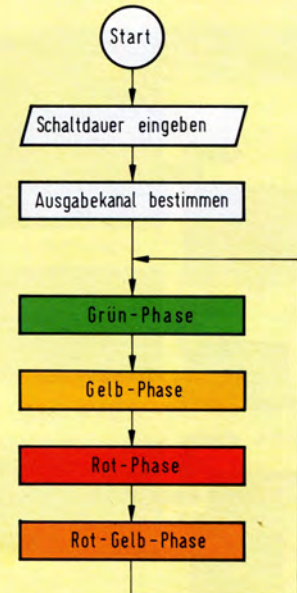


Bild 6:
Programmablaufplan zur
Steuerung der Verkehrsampel.

```

100 REM Steuerung der Verkehrsampel
110 REM
120 REM (C) Dr.R.Schule, 1986
130 REM
140 REM Funktion: Zyklischer Durchlauf
150 REM durch die Ampelphasen gruen,
160 REM gelb, rot und rot-gelb.
170 REM
180 REM Belegung der Ausgaenge:
190 REM Ausgang 0 - grueue Lampe
200 REM Ausgang 1 - gelbe Lampe
210 REM Ausgang 2 - rote Lampe
220 REM Ausgang 3 bis 7 nicht benutzt.
230 :
240 REM Titelmeldung
250 INK 0,0 :REM Hintergrund schwarz
260 INK 1,24 :REM Schriftfarbe gelb,
270 INK 2,6 : INK 3,9 :REM rot u. gruen
280 MODE 1 :REM Bild loeschen
290 LOCATE 1,3
300 PRINT "Verkehrsampel";
310 GOSUB 910 :REM Drucke Grafiksymbbole
320 DATA 7,3,2,231,1
330 PRINT "-----";
340 GOSUB 910
350 DATA 7,1,231
360 GOSUB 910
370 DATA 20,2,3,231
380 GOSUB 910
390 DATA 20,2,1,149
400 GOSUB 910
410 DATA 20,1,149
420 LOCATE 1,10
430 :
440 REM Schaltdauern eingeben
450 PRINT "Eingabe der Schaltdauern :";
460 PRINT "1 Sekunde ist ungefaehr 1000"
470 PRINT
480 INPUT "Dauer der Gruen-Phase";gruen
490 INPUT "Dauer der Gelb-Phase ";gelb
500 INPUT "Dauer der Rot-Phase ";rot
510 INPUT "Dauer der R./G.-Phase";rg
520 PRINT
530 PRINT "Programmstop mit 2xESC"
540 :

```

```

550 REM Ausgabekanal bestimmen
560 a=61439 :REM Druckerschnittstelle
570 :
580 REM Gruen-Phase
590 zeit=gruen
600 muster=1
610 GOSUB 810 :REM Ausgabe
620 :
630 REM Gelb-Phase
640 zeit=gelb
650 muster=2
660 GOSUB 810 :REM Ausgabe
670 :
680 REM Rot-Phase
690 zeit=rot
700 muster=4
710 GOSUB 810 :REM Ausgabe
720 :
730 REM Rot-Gelb-Phase
740 zeit=rg
750 muster=6
760 GOSUB 810 :REM Ausgabe
770 :
780 GOTO 590 :REM Ruecksprung
790 :
800 REM Ausgaberroutine
810 OUT a,muster :REM Ausgabe
820 FOR z=0 TO zeit
830 NEXT z
840 RETURN
850 :
860 REM Grafiksymbbole ausdrucken
870 REM Aufbau der DATA-Zeile:
880 REM lz=Zahl fuehrender Leerzeichen
890 REM gz=Zahl der Grafikzeichen
900 REM gz Grafiksymbbole (CHR$-Code) bzw. PEN
910 READ lz : PRINT SPC(lz);
920 READ gz
930 FOR i=1 TO gz
940 READ gc
950 IF gc<128 THEN PEN gc ELSE PRINT CHR$(gc);
960 NEXT i
970 PRINT
980 RETURN
READY.

```




Zuordnung der Ausgänge zu den Lampen ist willkürlich. Um aber in der Beschreibung der anschließenden Programme von einer einheitlichen Belegung ausgehen zu können, legen wir fest:

Ausgang 0 – grüne Lampe
Ausgang 1 – gelbe Lampe
Ausgang 2 – rote Lampe
Ausgang 3 wird nicht benutzt. Die komplette Verschaltung ist in **Bild 6** angegeben. Alle Verbindungen werden bei *ausgeschalteter* Anlage durchgeführt. Erst nach einer sorgfältigen Kontrolle werden Computer, IC-Versorgung und Verbraucherversorgung in dieser Reihenfolge eingeschaltet. Vermutlich werden alle drei Lampen der Ampel aufleuchten. Da ist sofortige Abhilfe notwendig. Wie zuvor schon, können wir die Ampel im Direktmodus steuern. Je nach Computertyp legen wir zunächst den Ausgabe-kanal fest:

Schneider CPC:

OUT A,4

soll die rote Lampe einschalten.

Sollte die Zuordnung vertauscht sein oder Lampen überhaupt nicht aufleuchten haben, so kontrollieren Sie nochmals die Verbindung zwischen Computer, Ausgabeplatine, Verstärkerplatine, IC-Versorgung, Verbraucherversorgung und Ampel. Mit diesem ersten Funktionstest haben wir auch schon fast alle Ausgabe-kommandos zusammengetragen, die in dem normalen Betrieb der Verkehrs-ampel auftauchen. In Deutschland erfolgt, im Gegensatz zu den Gepflogenheiten der meisten anderen Länder, der Übergang von Rot auf Grün mit der Zwischenphase Rot-Gelb. Für diese Phase sind zwei Bits auf einmal zu setzen. Wie in Folge 2 schon erläutert, werden hierzu die Stellenwerte der betreffenden Bits aufaddiert: Gelb erfordert eine 2, Rot eine 4, zusammen eine 6. Probieren wir also aus:

Commodore 64:

POKE A,6

Schneider CPC:

OUT A,6

Nach diesen Tests im Direktmodus können wir das Programm schreiben. Im Prinzip bräuchten wir nur die obigen Kommandos in der richtigen Reihenfolge aneinanderzureihen und am Schluß den Programm-lauf auf den Anfang zurückzuleiten, um einen endlosen Betrieb sicherzustellen. Jedoch fehlt uns an dieser Stelle noch die Erzeugung der Schaltdauer der einzelnen Ampelphasen. Hierzu nutzen wir, daß der Computer eine, wenn auch geringe, Zeitdauer zur Ausführung der Kommandos benötigt. Durch entsprechende Wiederholung mit Hilfe einer FOR...NEXT-Schleife erzielen wir passende Schaltzeiten. Allerdings benötigt die FOR...NEXT-Schleife

Commodore 64:

A=56577:POKE A+2,255

Schneider CPC:

A=61439

Die Ausgabekanäle weiterer Computer wurden in Folge 2 angegeben.

Durch das Direktkommando

Commodore 64:

POKE A,0

Schneider CPC:

OUT A,0

werden zunächst alle Lampen ausgeschaltet. Danach testen wir einzeln die Lampen der Verkehrsampel:

Commodore 64:

POKE A,1

Schneider CPC:

OUT A,1

soll die grüne,

Commodore 64:

POKE A,2

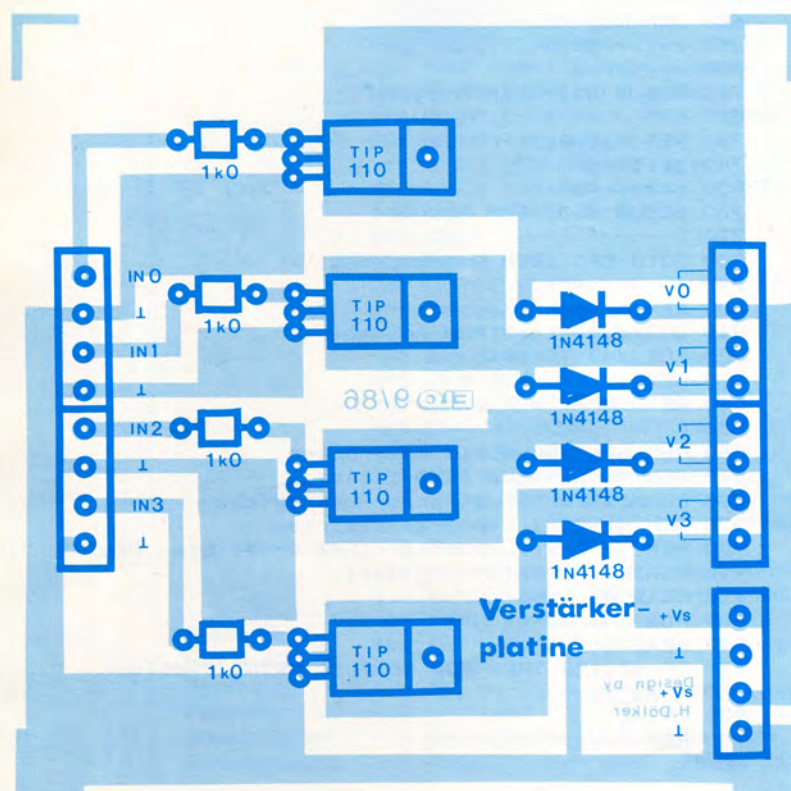
Schneider CPC:

OUT A,2

soll die gelbe und

Commodore 64:

POKE A,4



selbst schon einige Zeit; und wegen der Verwaltung des Sprungziels, der Zählvariablen und der Abfragebedingung sogar noch mehr als eine einfache Zuweisung. Daher genügt eine leere FOR...NEXT-Schleife schon zur Erzeugung von Verweildauern. Die Ausgabe eines bestimmten Bitmusters und die anschließende Erzeugung der Verweildauer tritt bei jeder der vier Ampelphasen auf. Auch künftige Ablaufsteuerungen lassen sich auf dieser Basis konstruieren. Deshalb wird ein gemeinsames Unterprogramm zu diesem Zwecke angelegt. Die vorzunehmende Tätigkeit wird mit Hilfe zweier Übergabeparameter gesteuert. MUSTER ist das auszugebende Bitmuster und ZEIT die geforderte Verweildauer. Eine weitere Zutat zu unserem Programm ist eine Anfangsphase, die den

Ausgabekanal setzt und eine flexible Eingabe der Ampelschaltzeiten ermöglicht. Alles zusammen wird nach dem in **Bild 7** angegebenen Flußdiagramm angeordnet. Das fertige Programm ist in **Bild 8** gezeigt. In den beiden Spalten sind die Versionen für den Commodore 64 und den Schneider CPC angegeben. Die Abwandlung für andere Computer anhand der in Folge 2 angegebenen Hinweise dürfte nicht allzu schwer fallen. Beim Vergleich der beiden Programme sieht man auch, daß die Aufteilung der Aufgaben in Unterprogramme noch einen weiteren Vorteil hat. Die maschinenspezifischen Ausgabekommandos wurden in ein Unterprogramm konzentriert. Die restlichen Teile des Programms sind dann nicht mehr maschinenabhängig und damit besser übertragbar. Wenn alle oben beschriebenen Tests erfolgreich ab-

geschlossen wurden, dürfen jetzt keine Hardwarefehler mehr vorliegen. Jegliche Abweichung vom erwarteten Verhalten müßte sich jetzt in der Software finden lassen. Kontrollieren Sie daher sorgfältig, ob Sie alle Kommandos richtig abgetippt haben. Nach diesem Start überlassen wir Ihnen die Ausarbeitung weiterer Aufgaben. Programmieren Sie z. B. ein Blinklicht mit der gelben Lampe, wie es häufig zu Nachtzeiten bei geringem Verkehrsaufkommen eingesetzt wird. Wenn Sie etwas in Hardware investieren wollen, bauen Sie sich eine zweite Verstärkerplatine auf und schließen diese an die Ausgänge 4 bis 7 der Ausgabeplatine an. Eventuell haben Sie ausreichendes Fischertechnik-Material, um eine ganze Kreuzung mit Ampeln auszustatten (die Lampen gibt es auch einzeln). Nehmen Sie sich eine echte

Stückliste Verstärkerplatine

Halbleiter

4 Transistoren TIP110
4 Dioden 1N4001

Widerstände

4 1,0 k Ω , 0,25 W

Sonstiges

5 Anschlußklemmen, 4polig
1 Platine ELO 9/1986

Kreuzung vor und notieren Sie sich Schaltfolge und Schaltzeiten. Sie werden feststellen, daß die Umschaltungen in den beiden Fahrtrichtungen nicht zusammenfallen und daher das Programm länger wird. Allerdings wird es noch die gleiche Struktur behalten können.

Dr. Roland Schulé

So wird der Atari angeschlossen

Die kleinen Ataris sind immer noch sehr beliebt und weit verbreitet. Sie zur Robotersteuerung einzusetzen erforderte Gedankenarbeit, dafür wenig Hardware:

Beim Atari 600/800XL lassen sich die beiden Joystickbuchsen als ein 8 Bit breiter Ausgabeport nutzen.

Mit POKE 54018,48

POKE 54016,255

POKE 54018,52 wird der Port auf Ausgabe geschaltet.

Anschließend:

Mit POKE 54016,255 werden alle 8 Bits gesetzt.

Mit POKE 54016,0 werden alle 8 Bits gelöscht.

Es läßt sich je nach Zahl zwischen 0 und 255 jedes Bit setzen (binär). Dahinter verbirgt sich der Portbaustein 6520.

An den Pins 5 und 9 jeder Joystickbuchse lassen sich Potentiometer (oder auch Schalter) an-

```
REM * VORBEREITUNG FUER AUSGABE *
POKE 54018,48
POKE 54016,255
POKE 54018,52
```

```
REM * BINAERES ZAEHLEN *
10 A=54016
20 POKE A,0
30 FOR B=0 TO 255
40 POKE A,B
50 FOR I=1 TO 200
60 NEXT I
70 NEXT B
```

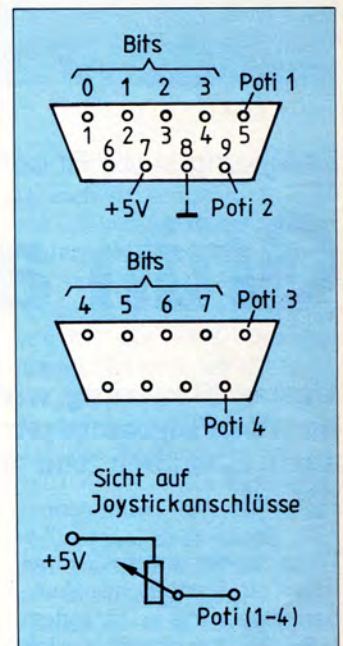
```
REM * POTISTELLUNG AUF AUSGABEPLATINE AUSGEBEN *
REM * POTI 1 *
10 A=54016
20 B=PEEK(624)*1.12
30 POKE A,B
40 GOTO 20
REM * ABRUCH MIT BREAK *
```

schließen. Somit stehen 4 Eingänge bei gleichzeitiger 1 Byte breiter Ausgabe zur Verfügung. In den Speicherzellen 624-627 können die Stellungen der Potis ausgelesen werden. Entweder mit

PEEK () oder dem Befehl PADDLE (0-3).

Die Potis sollten 500 k Ω haben. Die ausgelesenen Werte liegen zwischen 1 und 288.

Nach einem Reset sind die Joy-



stickbuchsen wieder auf Eingabe geschaltet. Wichtig: Wenn der Port auf Ausgabe gesetzt ist, sollten keine Joysticks angeschlossen werden.

Dietmar Gawehn, Seelze



Einführung in die Robotik:

Grün auf Knopfdruck

Unsere Steuerung wird komfortabler. Nicht nur das Programm wird stur verfolgt, sie läßt auch Eingaben von außen zu und reagiert darauf.

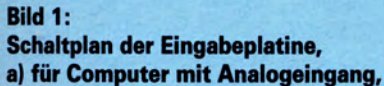
Wenn auch unsere Ampel schon die Vorteile der Computersteuerung zeigt, indem z. B. die Schaltzeiten sich schnell abändern lassen, so sind die Eigenschaften der Computersteuerung noch nicht voll zum Einsatz gekommen. Ein Computer kann auch Signale aus seiner Umwelt aufnehmen und

als Eingaben in Entscheidungsprozessen verarbeiten (mit Hilfe des IF...THEN-Kommandos). Um bei dem bearbeiteten Beispiel zu bleiben: das Eingabesignal könnte z. B. etwas über das Verkehrsaufkommen aussagen. Die Reaktion des Programms darauf könnte sich in einer Verschiebung der

Schaltzeiten äußern, mit dem Ziel, den Verkehrsfluß optimal zu steuern.

Bei unserer Modellampel vereinfachen wir die Aufgabe, ohne das Prinzip zu verlassen. An dem Ampelmast wird ein Tastschalter angebracht. Fußgänger, die die Straße überqueren wollen, betätigen diesen Taster. Die Ampelsteuerung soll nun die Grünphase beenden und einen Zyklus durchlaufen, um dem Fußgänger Gelegenheit zu geben, die Straße zu überqueren.

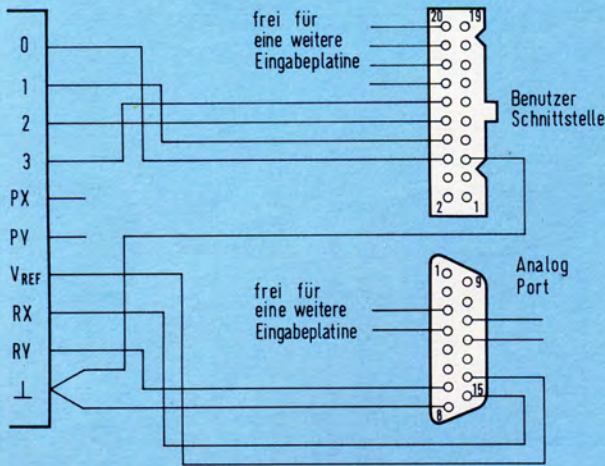
Zu diesem Zweck muß das Signal des Tastschalters verarbeitet werden. Auch hierfür haben wir eine Platine vorgesehen, die diesmal an den Spieleanschluß des Computers angeschlossen wird. Eigentlich hätten wir den bzw. die Taster des Modells direkt an den Spieleanschluß anschließen können. Nichts anderes als Taster befinden sich nämlich in den Joysticks. Dennoch treiben wir etwas mehr Aufwand. Wie schon bei der Ausgabeplatine werden wir die an-



ierlich durchstimmen lassen. Diese Art von Eingängen werden wir später noch vorteilhaft zu nutzen wissen. Je nach zu verwendendem Computer muß für den Paddle-Eingang ein unterschiedlicher Aufwand getrieben werden. **Bild 1** zeigt zwei Schaltpläne, **a)** für die Commodore-Computer VC20, C-64 und C-128 und **b)** für die Schneider-Computer CPC 464, CPC 664 und CPC 6128. Die erste Schaltung ist auch noch für den Apple II und IBM-PC geeignet. Wenn die Kondensatoren C1 und C2 weggelassen werden, kann diese Variante auch für die Analogeingänge des Acorn B und des Maccom eingesetzt werden. Die Schaltung b eignet sich auch für den NDR-Computer.

der Schaltung folgenden Dioden dienen zum Schutz vor Überspannung durch fehlerhafte Verdrahtung. Sie begrenzen die angelegten Spannungen auf den Bereich von 0 V bis +5 V. Genauer gesagt: Es muß jeweils noch die Flußspannung der Diode hinzugeschlagen werden, so daß die Begrenzung auf ca. -0,7 V bis +5,7 V erfolgt. Jedoch auch bei dieser erweiterten Begrenzung ist das nachfolgende IC noch vor Beschädigung geschützt. Dieses IC dient als Inverter. Wieder ist, wie schon bei der Ausgabeplatine, ein Inverter mit offenem Kollektor zum Zuge gekommen. Der Grund wird allerdings aus diesem Schaltungsteil nicht ersichtlich. Das IC wird jedoch auch noch im Analogein-

Bild 2:
Anschluß der Eingabeplatine
an verschiedene Computer-
systeme.



a) Acorn B

Anschluß an die Benutzerschnittstelle und den Analogport.
Einlesen von E0 bis E3 durch 765120, von EX und EY durch die
ADVAL-Funktion. C1 und C2 werden nicht bestückt.

gabeteil benutzt (es enthält ja sechs Inverter!) und dient dort zum Abschalten des Analogeingangs. Der beim Einsatz von ICs mit offenem Kollektor notwendige externe Widerstand zur Erzeugung des High-Signals wird in unserem Fall als Nebeneffekt von den Leuchtdioden zur Anzeige des Eingabestatus geliefert.

Die anschließende Beschaltung der Leuchtdiode kennen wir schon. Ein NAND-Gatter zum Abschluß führt zu einer nochmaligen Signalumkehr, so daß wieder die Originalpolarität hergestellt ist. Der jeweils zweite Eingang des NAND-Gatters ist mit der Freigabeleitung verbunden. Wird diese auf Masse gelegt, so erscheint an den Ausgängen ein High-Signal, unabhängig von den Eingangssignalen. Genauer gesagt: der Ausgang wird hochohmig, da es sich um ein IC mit offenem Kollektor handelt. Die interne Beschaltung des Spieleingangs des Computers wird somit nicht gestört. ICs mit

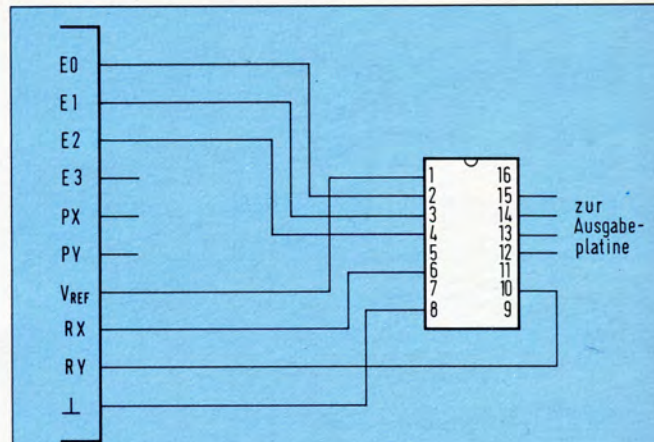
offenem Kollektor sind uns schon von der Ausgabeplatine bekannt.

Die untere Hälfte der Schaltung werden wir besprechen, wenn wir sie zum ersten Male einsetzen.

Aufbau und Test der Eingabeplatine

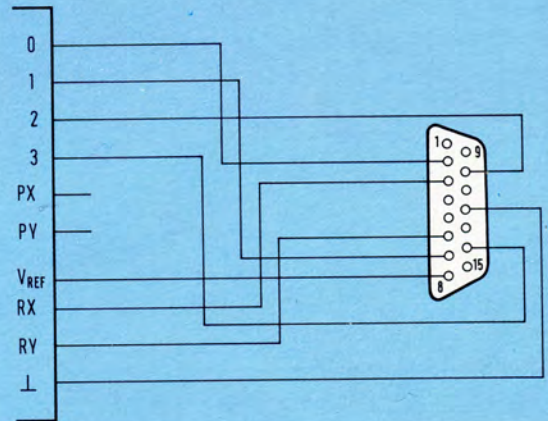
Bei dem Aufbau der Schaltung beachten Sie die Hinweise, die wir schon zum Aufbau der Ausgabeplatine gegeben haben. Für die ICs verwenden wir wieder Sockel. Bei den Leuchtdioden ist auf die Polung zu achten. Für die Bestückung der Kondensatoren sind auf der Leiterplatte mehrere Bohrungen vorgesehen, so daß Bauteile mit verschiedenem Rastermaß verwendet werden können. Der Anschluß der Platine ist in den **Bildern 2a bis g** dargestellt.

Commodore- und Schneider-Computer verwenden als Spieleingang einen neunpoligen Stecker. Daß wir diesen Stecker dennoch nicht immer gleich belegen,



b) Apple II

Anschluß an den Game Controller Port. Einlesen von E0 bis E2 mit PEEK(49249), PEEK(49250), PEEK(49251), EX und EY mit der PDL-Funktion. C1 und C2 mit 0,47 µF dimensionieren.

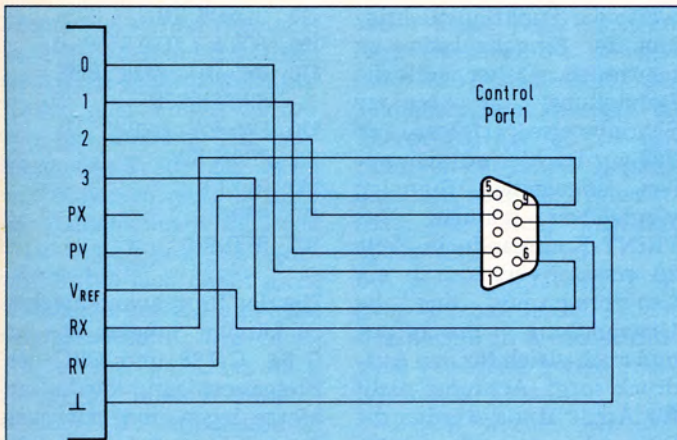


e) IBM-PC

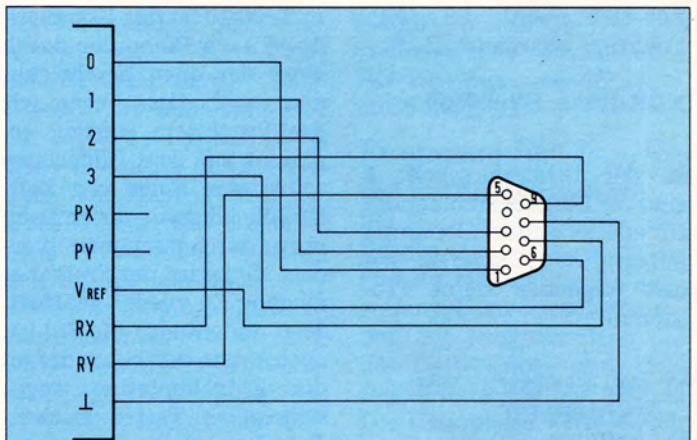
Anschluß an den Spieleadapter. Einlesen von E0 bis E3 durch INP(513). Für die Analogeingänge ist ein Maschinenprogramm notwendig. C1 und C2 müssen mit 0,47 µF dimensioniert sein.

hat Gründe in dem internen Aufbau der Computer. So verbietet es sich z. B. bei dem C-64, Eingang 3 zu benutzen. Würde dieser auf Masse gelegt, so hätte es den gleichen Effekt, wie wenn die CTRL-Taste gedrückt würde, d. h. die Bildschirm- ausgabe und damit das Programm würden verlangsamt. Entsprechendes gilt auch für die Schneider-Computer. Dort kann ein Low-Pegel an Anschluß 3 ein Drücken der ESC-Taste simulieren und den Programmablauf stoppen. Bei den Schneider-

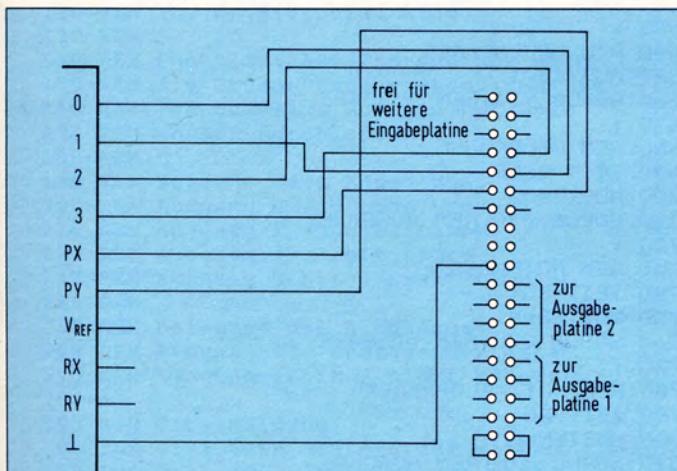
Computern ist zudem kein Massesignal im neunpoligen Stecker vorhanden, sondern lediglich ein Zeilenauswahlsignal der Tastatur. Dies darf natürlich nicht auf Masse gelegt werden. Die Masse muß vielmehr von der Ausgabeplatine herüberver- zweigt werden. Dies ist automatisch der Fall, wenn für Ein- und Ausgabeplatine die gleiche IC-Versorgung verwendet wird. Bei allen diesen Besonderheiten wird es fürs erste am besten sein, Sie halten sich genau an die Anschlußvorschriften aus



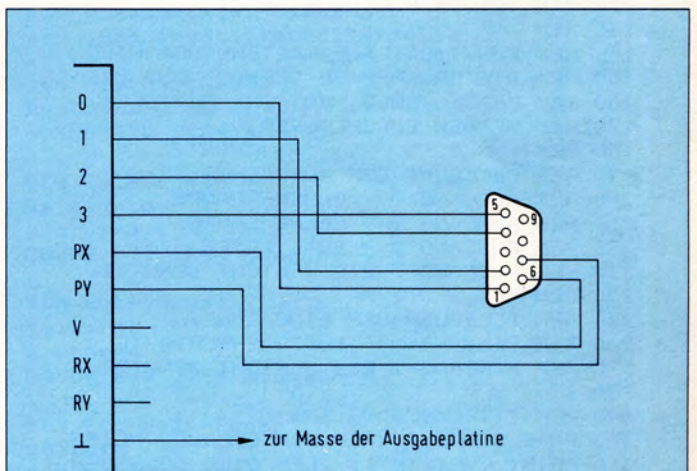
c) Commodore 64
Anschluß an den Control Port 1. Einlesen von E0 bis E3 mit PEEK(56321), EX mit PEEK(54297), EY mit PEEK(54298). C1 und C2 müssen mit 0,068 µF dimensioniert sein.



d) Commodore VC20
Anschluß an den Control Port. Einlesen von E0 bis E3 mit PEEK(37151), EX mit PEEK(36872), EY mit PEEK(36873). C1 und C2 müssen mit 0,068 µF dimensioniert sein.



f) NDR-Klein
Anschluß an die IOE-Platine. Einlesen von E0 bis E3 durch INP(48). Für die Analogeingänge ist ein Maschinenprogramm notwendig. C1 und C2 müssen mit 0,68 µF dimensioniert sein.



g) Schneider CPC
Anschluß an den Joystick-Eingang. Einlesen von E0 bis E3 durch JOY(0). Für die Analogwerte ist ein Maschinenprogramm notwendig. C1 und C2 müssen mit 0,68 µF dimensioniert sein. Der Masseanschluß wird über die Ausgabeplatine hergestellt.

Bild 2. Dort sind auch die Anschlüsse an den Acorn, Apple II, den IBM-PC und den NDR-Klein-Computer beschrieben.

Vor der ersten Inbetriebnahme der Platine erfolgt wieder eine gründliche Sichtprüfung. Die ICs werden nun bestückt. An die vier Eingänge werden Taster angeschlossen. Anschließend wird die Platine mit der IC-Versorgung, nicht jedoch mit dem Computer verbunden.

In der +5-V-Zuleitung ist ein Amperemeter einge-

schaltet, um die Stromaufnahme zu messen. Sie darf nicht mehr als 60 mA betragen. Alle vier Leuchtdioden müssen aufleuchten. Wenn jedoch ein Taster betätigt wird, d. h. der Eingang mit Masse verbunden wird, muß die dazugehörige Leuchtdiode erlöschen.

Wenn dieser erste Test zur Zufriedenheit ausgefallen ist, können wir wieder die Spannung abschalten und die Eingabeplatine auch noch an den Computer anschließen. Der Freigabeeingang En wird mit einer

Drahtbrücke mit dem benachbarten Masseanschluß verbunden.

Einlesen von Daten

Nun nehmen wir wieder alle Gerätschaften in Betrieb. Zum Test der Eingabeplatine müssen wir ein kleines Programm schreiben. Dieses soll den Eingabewert von dem Spieleanschluß einlesen und auf dem Bildschirm darstellen. Je nach Computersystem müssen wir zum Einlesen unterschiedliche Befehle verwenden. Bei

den Commodore-Computern wird die PEEK-Funktion verwendet:

C-64, C-128: EW=PEEK(56321)
VC20: EW=PEEK(37151)
(Control-Port 1)

Bei den Schneider-Computern CPC 464, CPC 664 und CPC 6128 ist ein eigenes Kommando zu diesem Zwecke vorhanden

EW=JOY(0)

Bei anderen Computern mit

Z80-CPU wird die INP-Funktion verwendet. Z. B.

NDR-Klein: EW=INP(48)

In der Bildunterschrift 2 sind die Eingabekommandos für etliche weitere Computer aufgeführt. Probieren wir also folgendes kleine Programm:

```
10 EW=PEEK(56321) : REM
   EINGABEWERT
20 PRINT EW
30 GOTO 10
```

In Zeile 10 ist das Kommando je nach Computer durch eines der oben beschriebenen zu ersetzen. Wenn wir das Programm starten, erscheint auf dem Bildschirm sofort eine Folge von Zahlen, die eilig von unten nach oben durchgeschoben werden. Nun wird der Freigabeeingang En wieder geöffnet. Jetzt ändern sich die Zahlen auch, wenn wir einen der an die Eingabeplatine angeschlossenen Taster drücken. Jedoch wird es uns schwer fallen, anhand der Zahlen-

werte die Funktionstüchtigkeit der Eingabeplatine zu beurteilen. Daher soll die Darstellung in binärer Schreibweise erfolgen, so daß wir leichter mit den außen angelegten Signalen vergleichen können. Das PRINT-Kommando in Zeile 20 ersetzen wir durch ein Unterprogramm, das die Umwandlung durchführt und auch gleich für den Ausdruck sorgt (Achtung: nicht für Apple II, da werden die Bits schon einzeln eingelesen!):

```
20 GOSUB 100
100 FOR I=7 TO 0 STEP -1
110 BW=INT(EW/2^I) : REM
   BINAERWERT
120 EW=EW-2^I*BW
130 PRINT BW;
140 NEXT I;
150 PRINT
160 RETURN
```

Um das Programm eingeben zu können, müssen Sie bei C-64, C-128 und CPC den Freigabeeingang wieder auf Masse legen und erst nach Programmstart wieder öffnen. Bei dem verbesserten

```
100 REM STEUERUNG DER FUSSGAENGERAMPEL
110 REM
120 REM (C) DR.R.SCHULE, 1986
130 REM
140 REM FUNKTION: AUF TASTENDRUCK WIRD
150 REM DIE GRUENPHASE BEENDET UND DIE
160 REM PHASEN GELB, ROT UND ROT-GELB
170 REM WERDEN DURCHLAUFEN.
180 REM
190 REM BELEGUNG DER AUSGAENGE:
200 REM AUSGANG 0 - GRUENE LAMPE
210 REM AUSGANG 1 - GELBE LAMPE
220 REM AUSGANG 2 - ROTE LAMPE
230 REM AUSGANG 3 BIS 7 NICHT BENUTZT.
240 REM
250 REM BELEGUNG DER EINGAENGE:
260 REM EINGANG 0 - FUSSGAENGERTASTE
270 REM EINGANG 1 BIS 3 NICHT BENUTZT.
280 :
290 REM TITELMELDUNG
300 POKE 53281,1 : REM HINTERGRUND WEISS
310 PRINT CHR$(147); : REM BILD LOESCHEN
320 PRINT CHR$(31); : REM SCHRIFT BLAU
330 PRINT : PRINT
340 PRINT "FUSSGAENGERAMPEL";
350 GOSUB 1030: REM DRUCKE GRAFIKSYMBOLE
360 DATA 4,3,28,113,31
370 PRINT "-----";
380 GOSUB 1030
390 DATA 4,2,158,113
400 GOSUB 1030
410 DATA 20,2,30,113
420 GOSUB 1030
430 DATA 20,3,152,98,179
440 GOSUB 1030
450 DATA 20,6,98,17,17,17,17,31
460 :
470 REM SCHALTDauern EINGEBEN
480 PRINT "EINGABE DER SCHALTDauern : "
490 PRINT "1 SEKUNDE IST UNGEFAEHR 500"
500 PRINT
510 INPUT "DAUER DER GELB-PHASE "; GELB
520 INPUT "DAUER DER ROT-PHASE "; ROT
530 INPUT "DAUER DER R./G.-PHASE"; RG
540 PRINT
550 PRINT "PROGRAMMSTOP MIT STOP-TASTE"
560 :
570 REM AUSGABEKANAL BESTIMMEN
580 A=56577 : REM BENUTZERSCHNITTSTELLE
590 POKE A+2,255 : REM AUSGABESTEUERUNG
```

```
600 :
610 REM EINGABEKANAL BESTIMMEN
620 E=56321 : REM JOYSTICKSCHNITTSTELLE
630 :
640 REM GRUENPHASE
650 MUSTER=1
660 GOSUB 930: REM AUSGABE
670 :
680 REM GELBPHASE
690 ZEIT=GELB
700 MUSTER=2
710 GOSUB 870: REM AUSGABE
720 :
730 REM ROTPHASE
740 ZEIT=ROT
750 MUSTER=4
760 GOSUB 870: REM AUSGABE
770 :
780 REM ROT/GELB-PHASE
790 ZEIT=RG
800 MUSTER=6
810 GOSUB 870: REM AUSGABE
820 :
830 REM RUECKSPRUNG
840 GOTO 600
850 :
860 REM AUSGABEROUTINE
870 POKE A,MUSTER : REM AUSGABE
880 FOR Z=0 TO ZEIT
890 NEXT Z
900 RETURN
910 :
920 REM EIN/AUSGABEROUTINE
930 POKE A,MUSTER : REM AUSGABE
940 IF (PEEK(E) AND 1)<>0 THEN GOTO 940
950 REM WARTE AUF TASTENDRUCK AN E0
960 RETURN
970 :
980 REM GRAFIKSYMBOLE AUSDRUCKEN
990 REM AUFBAU DER DATAZEILE:
1000 REM LZ=ZAHL FUEHRENDER LEERZEICHEN
1010 REM GZ=ZAHL DER GRAFIKSYMBOLE
1020 REM GZ GRAFIKSYMBOLE (CHR$-CODE)
1030 READ LZ : PRINT SPC(LZ);
1040 READ GZ
1050 FOR I=1 TO GZ
1060 READ GC : PRINT CHR$(GC);
1070 NEXT I
1080 PRINT
1090 RETURN
READY.
```

Bild 3a:
Die Ansteuerung der Fußgängerampel mit dem C-64.

Programm stehen nun acht Zahlenspalten auf dem Bildschirm, in denen jeweils nur Nullen oder Einsen zu finden sein dürften. Wir probieren den Taster, der an E0 angeschlossen ist, aus. Eine der Zahlenkolonne muß von 1 auf 0 springen. Die anderen dürfen sich dabei nicht ändern. Notieren Sie sich die Spaltennummer. Die Spalten werden von links nach rechts mit 7 bis 0 entsprechend der Stellenposition im Dualsystem durchnummeriert.

Wenn keine Null erscheint (und trotzdem die Leuchtdiode das Funktionieren des Eingabekanals anzeigt), so überprüfen Sie, ob Sie den richtigen Joystick-Eingang benutzen oder die Freigabeleitung En nicht noch mit Masse verbunden ist. Mit einem einfachen Vielfach-Meßinstrument läßt sich nachprüfen, ob der Eingang genügend hohen Pegel bekommt. Vielleicht werden Sie sich wundern, daß wir die Bits des Eingaberegisters nicht

der Reihenfolge nach verwendet haben (bei Commodore-Computern). Beim C-64, C-128 und den CPCs durfte Bit 2 nicht verwendet werden – siehe obige Notiz über den Konflikt mit der CTRL- bzw. ESC-Taste –, beim VC20 sind die Spielhebeleingänge auf zwei verschiedene Eingaberegister in verschiedenen Bits verteilt. Daher sollten Sie sich entsprechende Notizen machen, in welchen Bits Sie die Eingänge E0 bis E3 wiederfinden.

Sollten nicht alle Kanäle funktionieren, kontrollieren Sie die Belegung des Joystick-Steckers.

Programm für die Fußgängerampel

Mit Hilfe der funktionstüchtigen Eingabeplatine ist es nun ein leichtes, die Fußgängerampel zu programmieren. Die Programmteile für die Gelb-, Rot- und Rot-Gelb-Phase bleiben gleich. Die Grünphase wird jedoch nicht mehr über den Zeitverlauf gesteuert, sondern

```

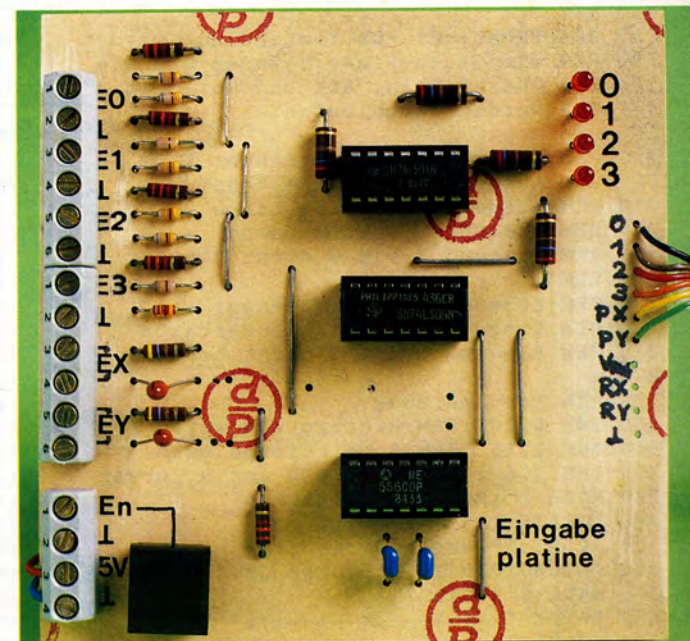
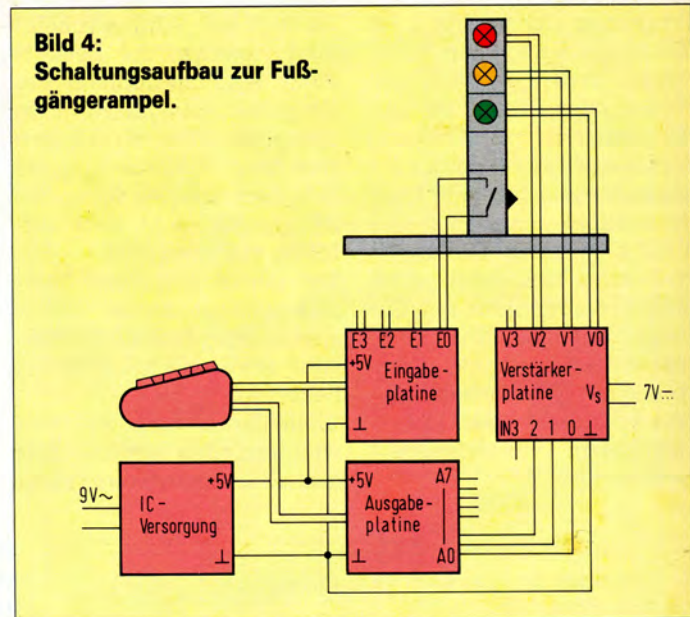
100 REM Steuerung der Fussgaengerampel
110 REM
120 REM (C) Dr.R.Schule, 1986
130 REM
140 REM Funktion: Auf Tastendruck wird
150 REM die Gruen-Phase beendet und die
160 REM Phasen Gelb, Rot und Rot-Gelb
170 REM werden durchlaufen.
180 REM
190 REM Belegung der Ausgaenge:
200 REM Ausgang 0 - gruene Lampe
210 REM Ausgang 1 - gelbe Lampe
220 REM Ausgang 2 - rote Lampe
230 REM Ausgang 3 bis 7 nicht benutzt.
240 REM
250 REM Belegung der Eingange:
260 REM Eingang 0 - Fussgaengertaste
270 REM Eingang 1 bis 3 nicht benutzt.
280 :
290 REM Titelmeldung
300 INK 0,26 :REM Hintergrund weiss
310 INK 1,15 :REM Schriftfarbe gelb,
320 INK 2,6 : INK 3,9 :REM rot u. gruen
330 MODE 1 :REM Bild loeschen
340 LOCATE 1,3 : PEN 3
350 PRINT "Fussgaengerampel";
360 GOSUB 1030 :REM Drucke Grafiksymbole
370 DATA 4,3,2,231,3
380 PRINT "-----";
390 GOSUB 1030
400 DATA 4,2,1,231
410 GOSUB 1030
420 DATA 20,2,3,231
430 GOSUB 1030
440 DATA 20,2,149,157
450 GOSUB 1030
460 DATA 20,1,149
470 LOCATE 1,10
480 :
490 REM Schaltdauern eingeben
500 PRINT "Eingabe der Schaltdauern :";
510 PRINT "1 Sekunde ist ungefaehr 1000"
520 PRINT
530 INPUT "Dauer der Gelb-Phase ";gelb
540 INPUT "Dauer der Rot-Phase ";rot
550 INPUT "Dauer der R./G.-Phase";rg
560 PRINT
570 PRINT "Programmstop mit 2xESC"
580 :
590 REM Ausgabekanal bestimmen
600 a=61439 :REM Druckerschnittstelle
610 :
620 REM Eingabekanal bestimmen
630 e=0 :REM Joystickschnittstelle
640 :
650 REM Gruen-Phase
660 muster=1
670 GOSUB 930 :REM Ausgabe
680 :
690 REM Gelb-Phase
700 zeit=gelb
710 muster=2
720 GOSUB 870 :REM Ausgabe
730 :
740 REM Rot-Phase
750 zeit=rot
760 muster=4
770 GOSUB 870 :REM Ausgabe
780 :
790 REM Rot-Gelb-Phase
800 zeit=rg
810 muster=6
820 GOSUB 870 :REM Ausgabe
830 :
840 GOTO 640 :REM Ruecksprung
850 :
860 REM Ausgaberroutine
870 OUT a,muster :REM Ausgabe
880 FOR z=0 TO zeit
890 NEXT z
900 RETURN
910 :
920 REM Ein/Ausgaberroutine
930 OUT a,muster :REM Ausgabe
940 IF (JOY(e) AND 1)<>0 THEN GOTO 940
950 REM Warte auf Tastendruck an E0
960 RETURN
970 :
980 REM Grafiksymbole ausdrucken
990 REM Aufbau der DATA-Zeile:
1000 REM lz=Zahl fuehrender Leerzeichen
1010 REM gz=Zahl der Grafikzeichen
1020 REM gz Grafiksymbole (CHR$(Code) bzw. PEN
1030 READ lz : PRINT SPC(lz);
1040 READ gz
1050 FOR i=1 TO gz
1060 READ gc
1070 IF gc<128 THEN PEN gc ELSE PRINT CHR$(gc);
1080 NEXT i
1090 PRINT
1100 RETURN

```

Bild 3b:
Die Fußgängerampel wird vom Schneider-CPC bedient.

Wenn das Programm funktioniert, sollten Sie sich stolz zurücklehnen und sich auf

* Diese Teile entfallen bei Computern mit eingebautem Analogeingang (Acorn, Apple II, C-64, IBM)



die Schulter klopfen lassen. Ist es nicht ein erhebendes Gefühl, den Arbeitstakt des Programms durch ein Signal von *außerhalb* des Computers zu steuern und dort auch wieder die Auswirkungen zu vernehmen?

Zum Schluß noch ein wichtiger Hinweis zur Benutzung der Eingabeplatine. Beim C-64, C-128 und bei den Schneider-Computern ist der Joystick-Anschluß parallel zu der Tastatur geschaltet. Während der Zeit, wo ein Low-Signal an der Eingabeplatine ansteht, hat es den Eindruck, daß die Tastatur „verrückt“ spielt. Dies betrifft sowohl Programmeingaben (INPUT- oder GET- bzw. INKEY-Befehle) als auch die Direktkommandos und Editierungsbefehle nach Programmstopp. Indem Sie die Freigabeleitung der Eingabeplatine auf Masse schalten, befreien Sie sich von diesem Spuk. Gleich anschließend werden wir dieses Problem noch eleganter lösen.

Dr. Roland Schulé

Neuaufgabe

Busch

Basic-Lexikon

Das Basis-
Lexikon der
Grundbegriffe
des BASIC
spricht
sich selbst

Basic

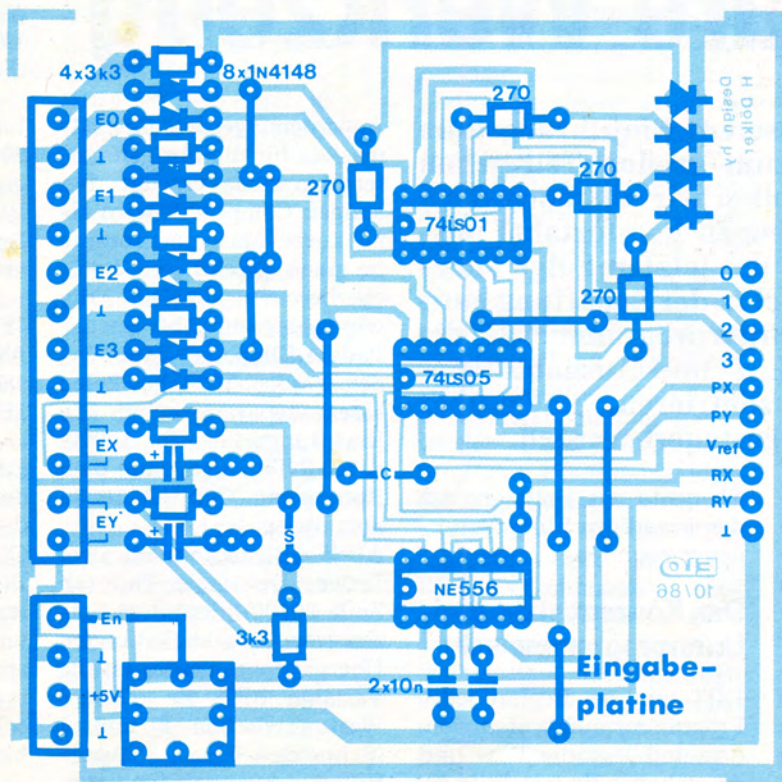
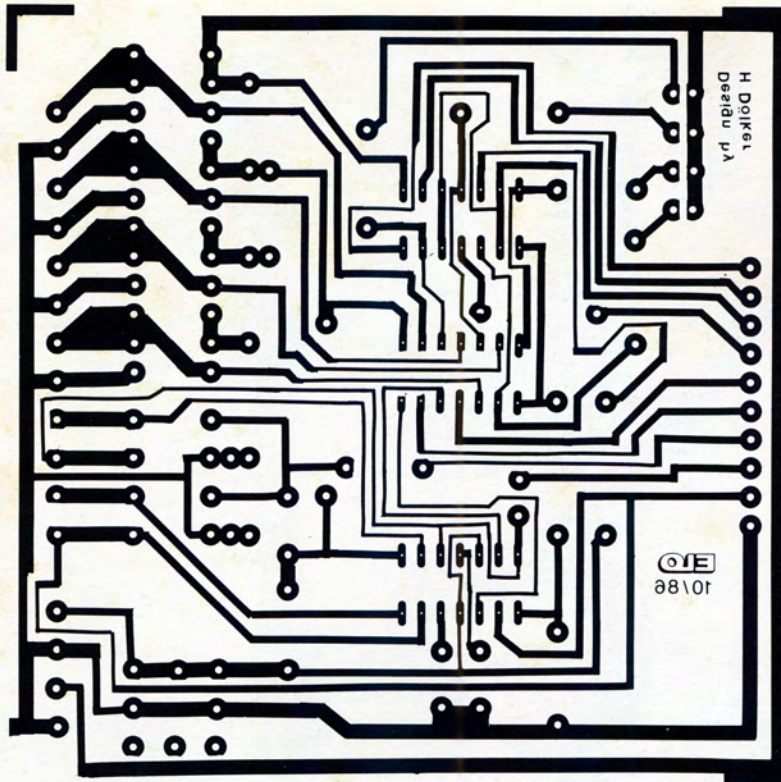
- ABS(X)
- ACOS(X)
- ADOT(X)
- ACS(X)
- AND(X)
- AND A,B
- ANDNOT(X)
- ANDNOT A,B
- ANDNOT A,B,C

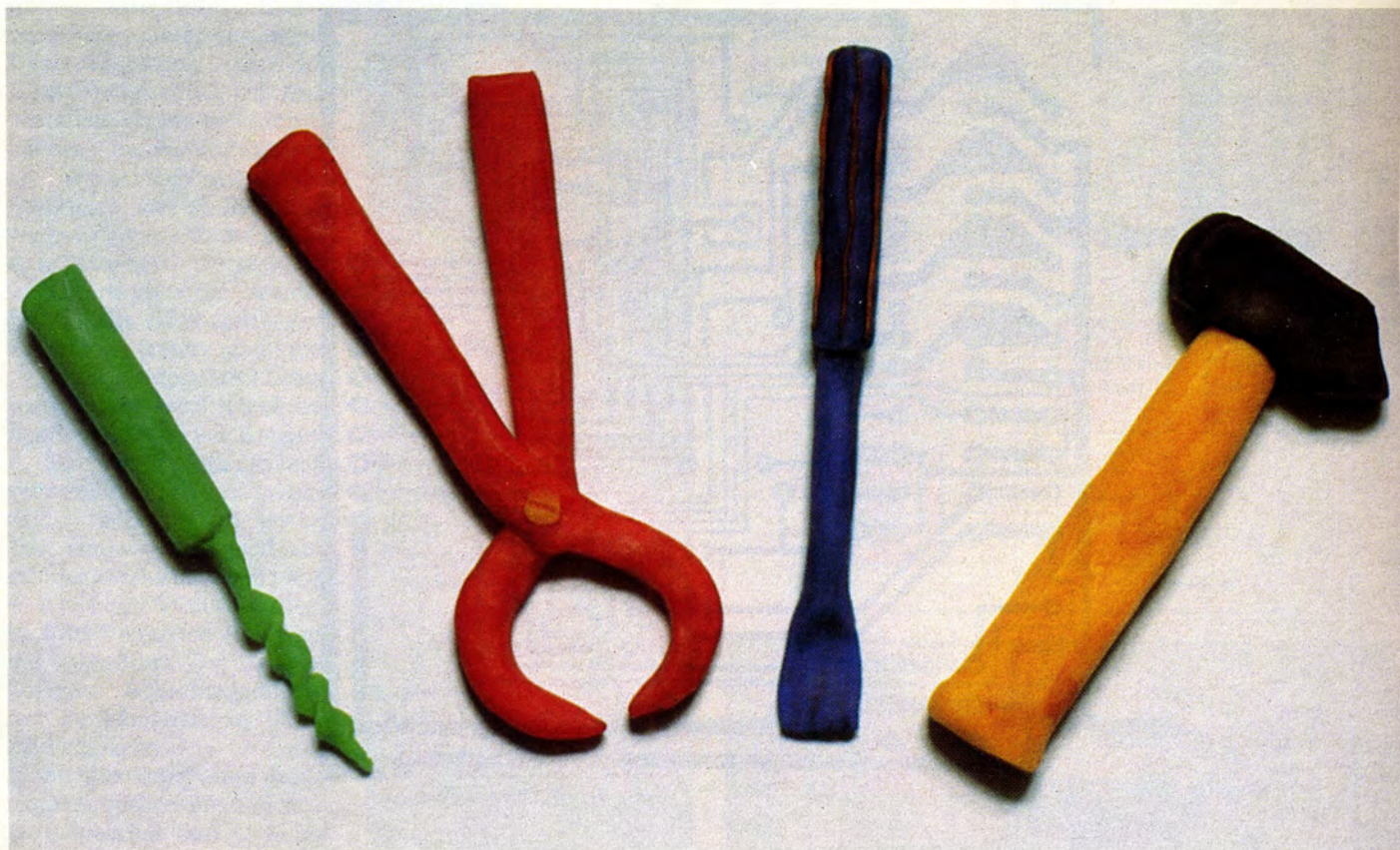
Jedem Anwender wird mit diesem Lexikon das mühsame und aufwendige Suchen und Recherchieren in zahlreichen Handbüchern erspart. Hier hat er die gebräuchlichsten Basic-Dialekte schnell und zuverlässig griffbereit und mit treffenden Beispielen erklärt.

F'

Franzis-Verlag GmbH
Karlstraße 37-41
8000 München 2
Telefon (089) 5117-1

Robo 87-95





Einführung in die Robotik:

Software-Werkzeug

Die Hardware zu unserem Einführungskurs in die Robotik ist zum großen Teil schon fertiggestellt, nun wollen wir uns auch noch mit Software-Werkzeugen komfortabel ausstatten. Wie schon in der letzten Folge angedeutet, besteht das Ziel in der Schaffung einiger leistungsfähiger Unterprogramme, die die computerabhängige Ein- und Ausgabesteuerung in sich konzentrieren und über genormte Parameter gesteuert werden.

Auf diese Weise werden die Roboterprogramme zum großen Teil portierbar, d. h. sie lassen sich von einem zum anderen Computer übertragen.

Ein weiteres Ziel ist die Steuerung und Abfrage von einzelnen Bits. Gerade in komplizierteren Aufbauten sind die unterschiedlichsten

Elemente des Roboters mit den jeweiligen Steuerleitungen verbunden.

Das Konzept sieht drei Unterprogramme vor

INIT muß zu Beginn jedes Programms aufgerufen werden und legt die Ein- und Ausgabekanäle fest (Bild 1).

Außerdem werden die Bitmasken für die übrigen Unterprogramme erzeugt. Bei einigen Computern wird eine kurze Maschinenroutine in einen geschützten Speicherbereich abgelegt, die wir später zur Abfrage der Paddle-Eingänge verwenden werden. INIT hat keine Übergabeparameter. Die Einsprungrzeile ist 10000, d. h. INIT wird mit dem Kommando GOSUB 10000 aufgerufen.

AUSGABE besteht aus zwei Teilen. Der erste Teil (ab Zeile 20100) dient dem Setzen von Ausgabeleitungen. Übergabeparameter ist die Variable BIT, die diesmal Werte zwischen 0 und 7 (Schneider: 0 und 6; Apple II: 0 und 3) annehmen kann.

Die entsprechende Leitung wird auf High geschaltet. Ein Beispiel:

```
1390 BIT=2 :REM LEITUNG
ANWAEHLEN
1400 GOSUB 20100 :REM
LEITUNG AUF HIGH
```

Im zweiten Teil (ab Zeile 20200) geschieht genau das gleiche, nur daß die durch BIT angewählte Leitung auf Low gesetzt wird. Wieder ein Beispiel:

```
1870 BIT=0 :REM LEITUNG
ANWAEHLEN
1880 GOSUB 20200 :REM
LEITUNG AUF LOW
```

Für diese Aufgaben sind in dem Unterprogramm INIT die beiden Felder B(...) und C(...) eingerichtet worden. B(...) enthält gerade die Stellenwerte des Binärsystems und C(...) das jeweils entsprechende 8-Bit-Komplement. Das Setzen und Löschen der Bits des Ausgabekanals geschieht mit Hilfe der OR-Funktion und B(...)


```

READY.

10000 :
10010 REM UNTERPROGRAMM INIT
10020 REM VERSION COMMODORE 64
10030 :
10040 DIM B(7),C(7),F(3) :REM BITMUSTER
10050 REM B(I) ZUM SETZEN VON BITS
10060 REM C(I) ZUM LOESCHEN VON BITS
10070 REM F(I) ZUM MASKIEREN DER EINGAENGE
10080 FOR I=0 TO 7
10090 B(I)=2^I : C(I)=255-B(I)
10100 NEXT I
10110 F(0)=1 : F(1)=2 : REM MASKEN FUER
10120 F(2)=8 : F(3)=16 :REM JOYSTICKEINGANG
10130 REM USERPORTADRESSE ZUR AUSGABE
10140 A=56577 :REM DATENREGISTER
10150 POKE A+2,255 :REM 8 BIT AUSGABE
10160 REM JOYSTICKADRESSE ZUR EINGABE
10170 D=56321 :REM DIGITALEINGAENGE
10180 REM PADDLEADR. ZUR ANALOGEINGABE
10190 EX=54297 :REM EINGANG POT-X
10200 EY=54298 :REM EINGANG POT-Y
10210 REM ALLE AUSGABELEITUNGEN ABSCHALTEN
10220 AUS=0 :REM AUSGABE-BITMUSTER
10230 POKE A,AUS
10240 E=56576 :REM STEUERREGISTER
10250 REM JOYSTICK ABSCHALTEN
10260 POKE E,PEEK(E) OR B(2)
10270 FOR I=0 TO 41 : READ CODE
10280 POKE 49152+I,CODE : NEXT I
10290 SYS 49152
10295 RETURN
10300 DATA 120,169,13,141,40,3,169,192
10310 DATA 141,41,3,88,96,165,145,201
10320 DATA 127,208,22,8,32,204,255,165
10330 DATA 198,72,169,0,141,1,221,173
10340 DATA 0,221,9,4,141,0,221,104

10350 DATA 40,96
20000 :
20010 REM UNTERPROGRAMM AUSGABE
20020 REM VERSION COMMODORE 64
20030 :
20040 REM EINGABEPARAMETER BIT (0...7)
20050 REM 20100 = EINSPRUNG 'SETZEN'
20100 AUS=AUS OR B(BIT)
20110 POKE A,AUS
20120 RETURN
20190 REM 20200 = EINSPRUNG 'LOESCHEN'
20200 AUS=AUS AND C(BIT)
20210 POKE A,AUS
20220 RETURN
30000 :
30010 REM UNTERPROGRAMM EINGABE
30020 REM VERSION COMMODORE 64
30030 :
30040 REM EINGABE DIGITALWERTE
30050 REM EINGABEPARAMETER BIT (0...3)
30060 REM RUECKGABE 'TRUE', WENN HIGH
30070 REM RUECKGABE 'FALSE', WENN LOW
30080 REM 30100 = EINSPRUNG
30100 POKE E,PEEK(E) AND C(2)
30110 EIN=(PEEK(D) AND F(BIT))<>0
30120 POKE E,PEEK(E) OR B(2)
30130 RETURN
30140 :
30170 REM EINGABE ANALOGWERTE
30180 REM RUECKGABE 0...255
30190 REM 30200 = EINSPRUNG X-WERT
30200 X=PEEK(EX)
30210 RETURN
30290 REM 30300 = EINSPRUNG Y-WERT
30300 Y=PEEK(EY)
30310 RETURN

READY.

```

Bild 1: Unterprogramme INIT, EINGABE und AUSGABE für den C-64.

bzw. der AND-Funktion und C(...). Die logischen Verknüpfungen werden zunächst auf eine Variable AUS (Ausgabewert) angewandt, die den jeweils letzten Zustand des Ausgabe-Kanals auch zwischenspeichert. Danach wird der Wert ausgegeben.

EINGABE besteht ebenfalls aus zwei Teilen. In dem ersten Teil (Aufrufadresse 30100) werden die Schaltereinträge der Eingabeplatine eingelesen. Steuerparameter ist wieder die Variable BIT mit den Werten 0, 1, 2 und 3 entsprechend dem zu testenden Eingang (Apple II: 0...2). Das Unterprogramm berücksichtigt die tatsächliche Bitlage im Eingaberegister. Die Rückgabe in der Variablen EIN ist FALSE, falls der Eingang auf LOW liegt, TRUE, wenn er auf High liegt.

Da die Schaltereinträge sowie meist in IF...THEN-Entscheidungen verwendet werden, liegt es nahe, EIN als logische Variable einzu-

richten. Das Beispiel zeigt die Benutzung:

```

1540 BIT=2 :REM LEITUNG
ANWAHLEN
1550 GOSUB 30100 :REM
LEITUNG TESTEN
1560 IF EIN THEN PRINT „HIGH“
1570 IF NOT EIN THEN PRINT
„LOW“

```

Bei einigen Computern (C-64, C-128 und CPC) ist der Spieleingang (Joystick-Port) an die gleichen Ein-Ausgabebebausteine angeschlossen wie die Tastatur. Ein Low-Signal an der Eingabeplatine kann daher die Tastatur blockieren, Eingaben verfälschen oder vortäuschen. Daher haben wir die Steuerung der Freigabeleitung der Eingabeplatine durch ein Ausgabesignal vorgesehen. Beim C-64 ist es die Leitung PA2 auf Pin M des Userports, beim CPC die Strobe-Leitung auf Pin 1 des Druckerports. Beide Signale werden

auf der Ausgabeplatine dem Eingang des Inverters E8 zugeführt. Da die Computersignale nach dem Einschalten auf High liegen, führt Ausgang A8 ein Low-Signal. Dieses wird auf den Freigabeingang En der Eingabeplatine geschaltet (Bild 2), die damit in aller Regel abgeschaltet ist und die Tastatur freigibt. Wird jedoch eine Eingabe vom Roboter benötigt, aktiviert das Programm EINGABE die Eingabeplatine wieder mit einem High an En. Danach können die Eingänge gelesen werden. Anschließend wird die Eingabeplatine gleich wieder abgeschaltet. Beim Stop des Programms wird die Eingabeplatine ebenfalls abgeschaltet. Bei den CPCs wird dies mit dem ON-BREAK-Kommando programmiert. Bei dem C-64 wird dagegen in dem Betriebssystem herumgeflickt. Das INIT-Unterprogramm installiert einen Patch (Flikken), der den Programmfluß bei Betätigen der Stoptaste

zunächst einmal in ein eigenes Maschinenprogramm (Bild 3) umlenkt. Das legt alle Ausgänge auf Low und schaltet die Eingabeplatine ab. Dann wird die Stop-routine des Betriebssystems weiterverfolgt. Beim CPC rücken wir auch von der praktischen JOY-Funktion ab. Der Grund: Die Werte der JOY-Funktion werden unabhängig vom Programmlauf alle 20 Millisekunden durch das Betriebssystem auf den neuesten Stand gebracht. Dies läßt sich aber nicht mit der

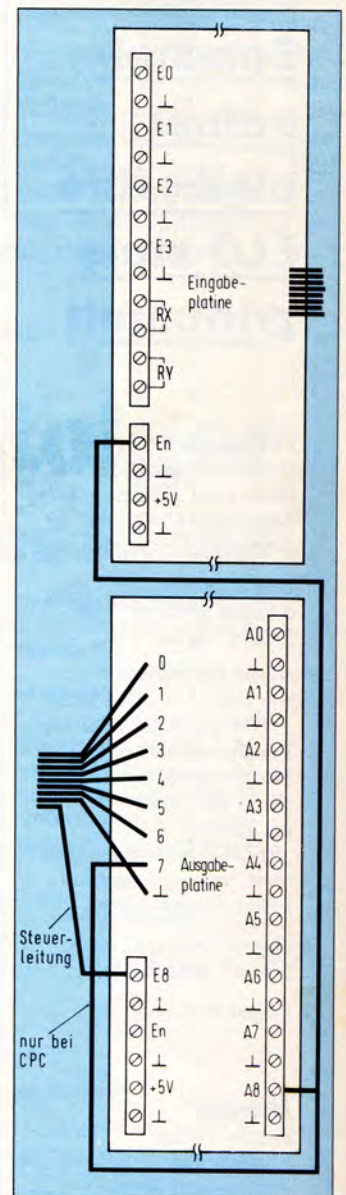


Bild 2: Freigabesteuerung der Eingabeplatine mit Hilfe der Ausgabeplatine.



Im praktischen Sammelordner bleibt Ihre ELO stets griffbereit

ELO-Sammelordner aus blauem Kunststoff für alle Hefte eines Jahrgangs. Das Selbstklebe-Etikett mit der Jahreszahl wird mitgeliefert.

Den Sammelordner erhalten Sie direkt vom Franzis-Verlag.

- Gegen Überweisung von DM 19,50 (DM 16,50 + DM 3,- Porto) auf unser Postgirokonto Nr. 813 75-809 Postgiroamt München (BLZ 700 100 80) mit Hinweis „ELO-Sammelordner 1986“ (bzw. welches Jahreszahl-Etikett Sie wünschen).
- Gegen Zusendung eines Schecks (DM 19,50).
- Gegen Rechnung.

Franzis'
Franzis-Verlag
Kunden- und
Abonnement-Service
Karlstraße 37-41
8000 München 2
Tel. 0 89/51 17-2 39/-3 80

:ASM

```

1 *****
2 *
3 * POWER-DOWN AUTOMATIK C64 *
4 *
5 *****
6 *
7 * SCHALTET DEN USER-PORT DES *
8 * C64 AUF 0, WENN DIE STOPPASTE *
9 * BETÄTIGT WIRD. AUSSERDEM WIRD *
10 * AUSGANG PA2 AUF HIGH GESCHAL- *
11 * TET, UM DIE TASTATUR WIEDER *
12 * FREIZUGEBEN. *
13 *
14 * MAERZ 1986 , DR.R.SCHULE *
15 *
16 *****
17 *
18 * FILE POWERDOWN.64
19 *
20 *****
21 *
22 * SYSTEMADRESSEN:
23 *
24 PORT EQU $DD00 ;CIA 2
25 PATCH EQU $0328 ;PATCH FUER STOPPASTE
26 STOP EQU $F6ED ;ROUTINE FUER STOPPASTE
27 ORG $C000 ;FREISPEICHER
28 SETUP SEI ;INTERRUPT SPERREN
29 LDA #<NEWSTOP ;TRAGE NEUE ADRESSE EIN
30 STA PATCH
31 LDA #>NEWSTOP
32 STA PATCH+1
33 CLI ;INTERRUPT FREIGEBEN
34 RTS
35 *
36 * NEWSTOP
37 *
38 * ERWEITERTE STOPROUTINE
39 *
40 NEWSTOP LDA $91 ;FLAGGE
41 CMP #$7F ;AUF STOP-CODE TESTEN
42 BNE WEITER
43 PHP
44 JSR $FFCC ;CLRCH
45 STA $C6
46 PHA
47 LDA #0 ;USER PORT AUSSCHALTEN
48 STA PORT+1
49 LDA PORT ;PA2 AUF HIGH
50 ORA #4
51 STA PORT
52 PLA
53 PLP
54 WEITER RTS

--END ASSEMBLY--

ERRORS: 0

42 BYTES

SYMBOL TABLE - ALPHABETICAL ORDER:

NEWSTOP = $C00D PATCH = $0328 PORT = $DD00 ? SETUP = $C000
? STOP = $F6ED WEITER = $C029

SYMBOL TABLE - NUMERICAL ORDER:

PATCH = $0328 ? SETUP = $C000 NEWSTOP = $C00D WEITER = $C029
PORT = $DD00 ? STOP = $F6ED

```

Bild 3: Modifizierte Stoproutine des C-64.

gewählten Freigabesteuerung vereinbaren. Die JOY-Funktion wird daher durch ein Stück Maschinenprogramm ersetzt, das im Prinzip dasselbe wie das Betriebssystem macht, jedoch zum vom Programmablauf gegebenen Zeitpunkt.

Da der CPC neben der Strobeleitung zur Steuerung der Eingabeplatine nur sieben Ausgabeleitungen besitzt, empfiehlt es sich, den freien achten Kanal der Ausgabeplatine zur Anzeige des Zustands der Strobeleitung zu verwenden. Hierzu wird der

Inverterausgang A8 mit einem kurzen Kabelstück auf den Eingang 7 zurückgeführt. Mit einem Blick auf die Leuchtdiode von A7 können Sie nun feststellen, ob die Tastatur freigegeben ist (Leuchtdiode aus) oder nicht.

Der zweite Teil des Unterprogramms Eingabe dient dem Einlesen von Paddle-Werten. Wir werden in einer der nächsten ELO noch auf diese Programme zurückkommen. Im Moment haben wir sie in unsere Sammlung mit aufgenommen, so daß die Unterprogramme vollständig sind.

Wenn Sie Programme schreiben, die die genannten Unterprogramme benutzen sollen, so achten Sie bitte darauf, daß Sie nicht die dort eingerichteten Variablen in anderem Zusammenhang benutzen. Die Unterprogramme geben wir in Bild 1 nur für den C-64 wieder. Listings für andere Computer können Sie bei uns bekommen (s. S. 32). Die Programme können Sie dann z. B. mit Hilfe des Diagnoseprogramms aus **Bild 4** testen. Dieses Programm sollte, wenn unsere Verspre-

chungen sich bewahrheiten, unabhängig von dem benutzten Computer sein. Leider trifft auch dies nicht ganz zu, da Unterschiede in der Behandlung der Tastatur und des Bildschirms verbleiben.

Es zeigt außerdem, wie wir künftige Programme dokumentieren werden: Die Ein- und Ausgabeunterprogramme sind in dem Abdruck nicht enthalten; sie müssen jeweils für den entsprechenden Computer ergänzt werden. Sie sollten sich die Unterprogramme auf Kassette oder Diskette abspeichern, um sie bei jedem neuen Programm hinzuladen zu können. Wenn Ihr Computer kein MERGE-Kommando oder etwas Vergleichbares besitzt, laden Sie die Unterprogramme als erstes, um anschließend den eigentlichen Programmtext hinzuzufügen.

Außerdem beschränken sich die Grafiken auf die Benutzung der Blockgrafiksymbole. Diese sind der schlechten Erkennbarkeit im Listing wegen mit ihrem CHR\$-Code (je nach Computer unterschiedlich!) aufgeführt. Eine kleine Routine sorgt für die Darstellung der Codes auf dem Bildschirm. Nach der ersten Erprobung steht es natürlich jedermann frei, die Schreibweise abzukürzen, indem die Grafiksymbole direkt in einen String übernommen werden oder aber die Darstellungen mittels hochauflösender Grafik zu verfeinern.

Da auf dem C-64 keine Kommandos PRINT AT und PRINT USING zur Verfügung stehen, werden sie durch String-Manipulationen ersetzt. Beim CPC und IBM wird die Druckposition mit dem Kommando LOCATE festgelegt, während beim

Apple II die getrennten Kommandos VTAB und HTAB einzusetzen sind.

Das Löschen des Bildschirms wird durch die Kommandos

PRINT CHR\$(147); (C-64),
MODE 1 bzw. CLS (CPC),
HOME (Apple II) usw.

besorgt. Auch die Steuerung der Bildschirmfarben ist, sofern überhaupt möglich, auf den verschiedenen Computern recht unterschiedlich.

Das Einlesen einzelner Zeichen von der Tastatur erfolgt mal mit der INKEY\$-Funktion, mal mit dem GET-Kommando, wobei zu beachten ist, daß beim Apple II das GET-Kommando nochmals vom Üblichen abweicht. Es erlaubt keine Eingabe im „Fluge“, sondern wartet immer auf den nächsten Tastendruck. Soll also das GET-Kommando korrekt in Applesoft BASIC übersetzt werden, so muß

NEU

Elektronik für Einsteiger

Das neue Sonderheft der ELO liefert ganz bewußt Grundwissen, das jeder braucht, der in die Elektronik einsteigen will. Die Beiträge reichen von allgemeinen elektronischen Grundlagen, z. B. die passiven Bauelemente, über Anregungen für das praktische Arbeiten, Bauanleitungen, Schaltungsauflösung, Löten, Platinenätzen bis hin zur Computertechnik. Das Sonderheft erklärt, wie Computer, Speichereinheiten, Mikroprozessoren funktionieren und beschreibt das duale Rechensystem eines Computers. Darüber hinaus erscheint ein Basic-Befehlsvergleich der wichtigsten Homecomputer. Um dem Einsteiger das theoretisch Gelernte auch praktisch deutlich zu machen, finden sich drei einfache Bauanleitungen im Heft. „Elektronik für Einsteiger“ präsentiert einen Wassermelder, der z. B. Kellerüberschwemmungen lautstark anzeigt, eine sogenannte Tastersperre, die bei Ratespielen nützlich ist.

Außerdem berichten wir über die technischen Grundlagen der Videographie, der HiFi-Technik und der Produktion von Videogeräten und Fernsehern. Wesentliches Element der praktischen Elektronik ist das Messen der aufgebauten und zu entwickelnden Schaltung. Dementsprechend werden in diesem Sonderheft die Grundlagen der Elektro-



Ein Sonderheft der **ELO**

nik-Meßtechnik erläutert und beschreibt, welche Geräte dazu notwendig sind ...

Die Voraufgabe wurde komplett überarbeitet und dem neuesten Stand der Technik angepaßt.

Bezugsmöglichkeiten:

Bei allen Bahnhofsbuchhandlungen, beim Elektronik-Fachhandel, in Mikrocomputershops, bei größeren Zeitschriftenverkaufsstellen, in Buchhandlungen oder direkt beim Franzis-Verlag mit Hinweis: Sonderheft „Elektronik für Einsteiger“

- per Postkarte
(Rechnung wird zugesandt)
- per Scheck
(DM 10,- + DM 2,- Porto)

Franzis'

Franzis-Verlag, Karlstraße 37,
8000 München 2,
Tel. 089/5117-240/-279

In der Schweiz: Thali AG, Fachliteratur
und Software, Industriestrasse 6,
CH-6285 Hitzkirch

In Österreich: Erb-Verlag,
Ges.m.b.H. & Co. KG, Mariahilfer Straße 71,
A-1061 Wien

Bild 4:

Mit dem Diagnoseprogramm für den C-64 kann die Hardware ausgetestet werden.

Bit 7 des Tastaturregisters mit Hilfe der Funktion PEEK(-16384) getestet werden.

Das Diagnoseprogramm

Hier wollen wir noch kurz das zuvor schon angesprochene Diagnoseprogramm beschreiben. Das Programm stellt symbolisch die Ein- und Ausgabeplatine auf dem Bildschirm dar. Die Leuchtdiodendarstellung sollte mit dem tatsächlichen Erscheinungsbild übereinstimmen. Ist dies nicht der Fall, sind die Anschlüsse zu überprüfen: Eingabestecker, Ausgabestecker, Stromversorgung. Wenn Sie an die Eingabeplatine vier Tastschalter angeschlossen haben, können Sie die Eingabeleitungen durchtesten. Auch die Paddle-Eingänge werden auf dem Schirm dargestellt; solange nichts angeschlossen ist, zeigen sie den Maximalwert 255 an.

Entsprechend der Belegung der Funktionstasten können Sie die Leitungen der Ausgabeplatine steuern. Wählen Sie den gewünschten Ausgang an. Danach haben Sie die Wahl, ob Sie den Ausgang umschalten oder nur einen kurzen Impuls (aus oder ein) absetzen. Ersteres dient der Dauerfunktion, letzteres insbesondere dem Austesten von motorischen Bewegungen.

Das Programm wird Ihnen immer wieder nützliche Dienste beim Austesten neuer Roboter und Anlagen erweisen. Sie sollten sich zur Gewohnheit machen, bei jedem Aufbau konsequent die Zuordnung der Bits, die Funktionsfähigkeit der Verstärker und die Drehrichtung der Motoren (s. u.) auszutesten. *Roland Schulé*

```

110 REM DIAGNOSEPROGRAMM
130 REM (C) DR.R.SCHULE, 1986
150 REM FUNKTION: DER BETRIEBSZUSTAND
160 REM DER AUSGABE- UND EINGABEPLATINE
170 REM WIRD ANGEZEIGT. DIE AUSGABEPLA-
180 REM TINE KANN MITHILFE DER TASTATUR
190 REM GESTEUERT WERDEN:
200 REM F1 - CURSOR NACH OBEN
210 REM F7 - CURSOR NACH UNTEN
220 REM F3 - AUSGANG UMSCHALTEN
230 REM F5 - AUSGANG UMTASTEN
250 REM TITELGRAPHIK
260 POKE 53281,1 :REM HINTERGRUND WEISS
270 PRINT CHR$(147); :REM BILD LOESCHEN
280 PRINT CHR$(31); :REM SCHRIFT BLAU
290 PRINT : PRINT
300 PRINT "      DIAGNOSEPROGRAMM" : PRINT
310 GOSUB 2250
320 DATA 5,30,111,183,183,183,183,183,183,183,183,112,32,32,32
330 DATA 32,32,32,111,183,183,183,183,183,183,183,183,183,112
340 GOSUB 2250
350 DATA 5,30,165,32,32,32,32,32,32,32,32,32,167,32,32,32
360 DATA 32,32,32,165,32,32,32,32,32,32,32,32,32,32,167
370 GOSUB 2250
380 DATA 5,30,165,65,32,32,32,32,32,32,48,32,119,167,32,32,32
390 DATA 32,32,32,165,69,32,32,32,32,32,32,48,32,119,167
400 GOSUB 2250
410 DATA 5,30,165,85,32,32,32,32,32,32,49,32,119,167,32,32,32
420 DATA 32,32,32,165,73,32,32,32,32,32,32,49,32,119,167
430 GOSUB 2250
440 DATA 5,30,165,83,32,32,32,32,32,32,50,32,119,167,32,32,32
450 DATA 32,32,32,165,78,32,32,32,32,32,32,50,32,119,167
460 GOSUB 2250
470 DATA 5,30,165,71,32,32,32,32,32,32,51,32,119,167,32,32,32
480 DATA 32,32,32,165,71,32,32,32,32,32,32,51,32,119,167
490 GOSUB 2250
500 DATA 5,30,165,65,32,32,32,32,32,32,52,32,119,167,32,32,32
510 DATA 32,32,32,165,65,32,32,32,32,32,176,96,96,96,174,167
520 GOSUB 2250
530 DATA 5,30,165,66,32,32,32,32,32,32,53,32,119,167,32,32,32
540 DATA 32,32,32,165,66,32,32,32,88,125,32,32,32,125,167
550 GOSUB 2250
560 DATA 5,30,165,69,32,32,32,32,32,32,54,32,119,167,32,32,32
570 DATA 32,32,32,165,69,32,32,32,32,171,96,96,96,179,167
580 GOSUB 2250
590 DATA 5,30,165,32,32,32,32,32,32,32,55,32,119,167,32,32,32
600 DATA 32,32,32,165,32,32,32,32,89,125,32,32,32,125,167
610 GOSUB 2250
620 DATA 5,30,165,32,32,32,32,32,32,32,32,32,167,32,32,32
630 DATA 32,32,32,165,32,32,32,32,32,173,96,96,96,189,167
640 GOSUB 2250
650 DATA 5,30,108,175,175,175,175,175,175,175,175,175,186,32,32,32
660 DATA 32,32,32,108,175,175,175,175,175,175,175,175,175,175,186
1000 REM HAUPTPROGRAMM DIAGNOSE
1010 GOSUB 1830:REM INITIALISIEREN
1020 GOSUB 1110:REM AUSGABESTEUERUNG
1030 GOSUB 1620:REM EINGABEDARSTELLUNG
1040 GOTO 1020:REM RUECKSPRUNG
1100 REM AUSGABESTEUERUNG
1110 AA=AL(ABIT)
1120 GET K$
1130 IF K$="" THEN GOTO 1310
1140 K=ASC(K$)
1150 IF K<>133 THEN GOTO 1210
1160 IF ABIT=0 THEN RETURN
1170 GOSUB 1430
1180 ABIT=ABIT-1
1190 GOSUB 1470
1200 RETURN
1210 IF K<>136 THEN GOTO 1270
1220 IF ABIT=7 THEN RETURN
1230 GOSUB 1430
1240 ABIT=ABIT+1
1250 GOSUB 1470
1260 RETURN
1270 IF K<>134 THEN GOTO 1300
1280 AL(ABIT)=1-AL(ABIT) :REM UMSCHALTEN
1290 AA=AL(ABIT)
1300 IF K=135 THEN AA=1-AL(ABIT)
1310 BIT=ABIT
1320 REM AA AN POSITION ABIT AUSGEBEN
1330 ON AA+1 GOTO 1340,1380
1340 GOSUB 20200 :REM LOW AUSGEBEN

```



```

1350 PRINT LEFT$(VT$,7+ABIT);TAB(15);
1360 PRINT AUS$; :REM GRAFIKSYMBOL
1370 RETURN
1380 GOSUB 20100 :REM HIGH AUSGEBEN
1390 PRINT LEFT$(VT$,7+ABIT);TAB(15);
1400 PRINT EIN$; :REM GRAFIKSYMBOL
1410 RETURN
1420 REM PFEIL LOESCHEN
1430 PRINT LEFT$(VT$,7+ABIT);TAB(11);
1440 PRINT CHR$(32);
1450 RETURN
1460 REM PFEIL DRUCKEN
1470 PRINT LEFT$(VT$,7+ABIT);TAB(11);
1480 PRINT CHR$(62);
1490 SOUND=54272 : POKE SOUND+24,15
1500 POKE SOUND+4,0 : POKE SOUND+5,0
1510 POKE SOUND+5,6 : POKE SOUND+6,0
1520 POKE SOUND,135 : POKE SOUND+1,33
1530 POKE SOUND+4,17
1540 RETURN
1600 REM EINGABEDARSTELLUNG
1610 REM CURSOR AUF ERSTE DRUCKPOSITION
1620 PRINT LEFT$(VT$,7);TAB(33);
1630 FOR BIT=0 TO 3
1640 GOSUB 30100 :REM DIGITALE EINGABE
1650 LED$=AUS$ : IF EIN THEN LED$=EIN$
1660 PRINT LED$;
1670 NEXT BIT
1680 GOSUB 30200 :REM X-WERT
1690 X$=RIGHT$(" "+STR$(X),3)
1700 GOSUB 30300 :REM Y-WERT
1710 Y$=RIGHT$(" "+STR$(Y),3)
1720 PRINT LEFT$(VT$,12);TAB(30);
1730 PRINT X$;
1740 PRINT LEFT$(VT$,14);TAB(30);
1750 PRINT Y$;
1760 RETURN
1800 REM ANFANGSWERTE SETZEN

```

```

10000 REM Unterprogramm INIT
10010 REM Version Schneider CPC
10020 :
10030 DIM b(7),c(7),f(3) :REM Bitmuster
10040 REM b(i) zum Setzen von Bits
10050 REM c(i) zum Loeschen von Bits
10060 REM f(i) zum Maskieren der Eingänge
10070 FOR i=0 TO 7
10080 b(i)=2^i : c(i)=255-b(i)
10090 NEXT i
10100 f(0)=1 : f(1)=2 : REM Masken fuer
10110 f(2)=8 : f(3)=64 :REM Joystickeingang
10120 REM Druckeradresse zur Ausgabe
10130 a=61439
10140 REM Einsprungsadressen der
10150 REM Maschinenprogramme zur
10160 REM Eingabe der Digital- und
10170 REM Analogwerte
10180 e1234=&A001 :REM Eingang e1 bis e4
10190 ex=&A005 :REM Eingang Pot-X
10200 ey=&A009 :REM Eingang Pot-Y
10210 REM Maschinenprogramm ablegen
10220 FOR i=0 TO 101
10230 READ code : POKE e1234+i,code
10240 NEXT i
10250 REM Ausgabeleitungen abschalten
10260 aus=0 :REM Ausgänge abschalten
10270 OUT a,aus
10280 ON BREAK GOSUB 10450
10290 RETURN
10300 DATA &1e,&01,&18,&06,&1e,&10,&18,&02
10310 DATA &1e,&20,&f3,&01,&0e,&f4,&ed,&49
10320 DATA &06,&f6,&ed,&78,&e6,&30,&4f,&f6
10330 DATA &c0,&ed,&79,&ed,&49,&04,&3e,&92
10340 DATA &ed,&79,&c5,&cb,&f1,&06,&f6,&ed
10350 DATA &49,&06,&f4,&cb,&43,&20,&27,&16
10360 DATA &00,&ed,&78,&a3,&28,&06,&14,&20
10370 DATA &f8,&15,&18,&1c,&16,&00,&ed,&78
10380 DATA &a3,&20,&06,&14,&20,&f8,&15,&18
10390 DATA &0f,&16,&00,&ed,&78,&a3,&28,&08
10400 DATA &14,&20,&f8,&15,&18,&02,&ed,&50
10410 DATA &c1,&3e,&82,&ed,&79,&05,&ed,&49
10420 DATA &7a,&32,&00,&a0,&fb,&c9
10430 :
10440 REM Ein- und Ausgaben abschalten
10450 OUT a;0

```

```

1810 REM
1820 REM LED-SYMBOL
1830 EIN$=CHR$(150)+CHR$(113)+CHR$(31)
+CHR$(17)+CHR$(157)
1840 AUS$=CHR$(119)+CHR$(17)+CHR$(157)
1850 REM
1860 REM CURSOR-BEWEGUNGEN
1870 VT$=CHR$(19)
1880 FOR I=1 TO 25
1890 VT$=VT$+CHR$(17)
1900 NEXT
1910 REM AUSGABEN VORSETZEN
1920 DIM AL(7)
1930 FOR BIT=0 TO 7
1940 AL(BIT)=0
1950 NEXT BIT
1960 ABIT=0
1970 GOSUB 10000 :REM EIN/AUSGABE VORBEREITEN
1980 PRINT
1990 PRINT " PROGRAMMSTOP MIT STOPPASTE!"
2000 PRINT
2010 PRINT " F1 NACH OBEN"
2020 PRINT " F3 UMSCHALTEN"
2030 PRINT " F5 IMPULS"
2040 PRINT " F7 NACH UNTEN"
2050 RETURN
2200 REM UNTERPROGRAMM GRAFIKSYMBOL
2210 REM PARAMETER DER DATAZEILE:
2220 REM LZ=ZAHL FUEHRENDER LEERZEICHEN
2230 REM GZ=ZAHL DER GRAFIKSYMBOL
2240 REM GZ GRAFIKSYMBOL (CHR$-CODE)
2250 READ LZ : PRINT SPC(LZ);
2260 READ GZ
2270 FOR I=1 TO GZ
2280 READ GC : PRINT CHR$(GC);
2290 NEXT I
2300 PRINT
2310 RETURN

```

```

10460 END
20000 REM Unterprogramm AUSGABE
20010 REM Version Schneider CPC
20020 :
20030 REM Eingabeparameter bit (0...6)
20040 REM 20100 = Einsprung 'SETZEN'
20100 aus=aus OR b(bit)
20110 OUT a,aus
20120 RETURN
20130 REM 20200 = Einsprung 'LOESCHEN'
20200 aus=aus AND c(bit)
20210 OUT a,aus
20220 RETURN
30000 REM Unterprogramm EINGABE
30010 REM Version Schneider CPC
30020 :
30030 REM Eingabe Digitalwerte
30040 REM Eingabeparameter bit (0...3)
30050 REM Rueckgabe 'TRUE', wenn HIGH
30060 REM Rueckgabe 'FALSE', wenn LOW
30070 REM 30100 = Einsprung
30100 OUT a,aus OR 128
30110 CALL e1234
30120 OUT a,aus AND 127
30130 ein=(PEEK(e1234-1) AND f(bit))<>0
30140 RETURN
30170 REM Eingabe Analogwerte
30180 REM Rueckgabe 0...255
30190 REM 30200 = Einsprung X-Wert
30200 OUT a,aus OR 128
30210 REM Verzoeigerung zum Anschwingen
30220 FOR x=0 TO 3 :NEXT
30230 CALL ex
30240 OUT a,aus AND 127
30250 x=PEEK(e1234-1)
30260 CLEAR INPUT
30270 RETURN
30290 REM 30300 = Einsprung Y-Wert
30300 OUT a,aus OR 128
30310 REM Verzoeigerung zum Anschwingen
30320 FOR y=0 TO 3 :NEXT
30330 CALL ey
30340 OUT a,aus AND 127
30350 y=PEEK(e1234-1)
30360 CLEAR INPUT
30370 RETURN

```


Diagnoseprogramm für den CPC.

```

100 MEMORY &9FFF
110 REM Diagnoseprogramm
120 REM
130 REM (C) Dr.R.Schule, 1986
140 REM
150 REM Funktion: Der Betriebszustand
160 REM der Ausgabe- und Eingabeplatine
170 REM wird angezeigt. Die Ausgabepla-
180 REM tine kann mithilfe der Tastatur
190 REM gesteuert werden:
200 REM f7 - Cursor nach oben
210 REM f0 - Cursor nach unten
220 REM f4 - Ausgang umschalten
230 REM f1 - Impulsausgabe
240 :
250 REM Titelgraphik
260 INK 0,26 :REM Hintergrund weiss
270 MODE 1 :REM Bild loeschen
280 INK 1,1 :REM Schrift blau, rot und
290 INK 2,6 : INK 3,1,6 :REM blinkend
300 LOCATE 1,3
310 PRINT"      Diagnoseprogramm"
320 LOCATE 1,5
330 GOSUB 2250
340 DATA 5,30,135,131,131,131,131,131,131,131,131,131,131,139,32,32,32
350 DATA 32,32,32,135,131,131,131,131,131,131,131,131,131,131,131,139
360 GOSUB 2250
370 DATA 5,30,133,32,32,32,32,32,32,32,32,32,32,138,32,32,32
380 DATA 32,32,32,133,32,32,32,32,32,32,32,32,32,32,138
390 GOSUB 2250
400 DATA 5,30,133,65,32,32,32,32,243,32,48,32,230,138,32,32,32
410 DATA 32,32,32,133,69,32,32,32,32,32,32,48,32,230,138
420 GOSUB 2250
430 DATA 5,30,133,85,32,32,32,32,32,32,49,32,230,138,32,32,32
440 DATA 32,32,32,133,73,32,32,32,32,32,32,49,32,230,138
450 GOSUB 2250
460 DATA 5,30,133,83,32,32,32,32,32,32,50,32,230,138,32,32,32
470 DATA 32,32,32,133,78,32,32,32,32,32,32,50,32,230,138
480 GOSUB 2250
490 DATA 5,30,133,71,32,32,32,32,32,32,51,32,230,138,32,32,32
500 DATA 32,32,32,133,71,32,32,32,32,32,32,51,32,230,138
510 GOSUB 2250
520 DATA 5,30,133,65,32,32,32,32,32,32,52,32,230,138,32,32,32
530 DATA 32,32,32,133,65,32,32,32,32,32,150,154,154,154,156,138
540 GOSUB 2250
550 DATA 5,30,133,66,32,32,32,32,32,32,53,32,230,138,32,32,32
560 DATA 32,32,32,133,66,32,32,32,88,149,32,32,32,149,138
570 GOSUB 2250
580 DATA 5,30,133,69,32,32,32,32,32,32,54,32,230,138,32,32,32
590 DATA 32,32,32,133,69,32,32,32,32,151,154,154,154,157,138
600 GOSUB 2250
610 DATA 5,32,133,32,32,32,32,32,32,32,55,32,3,230,1,138,32,32,32
620 DATA 32,32,32,133,32,32,32,32,89,149,32,32,32,149,138
630 GOSUB 2250
640 DATA 5,30,133,32,32,32,32,32,32,32,32,32,138,32,32,32
650 DATA 32,32,32,133,32,32,32,32,147,154,154,154,153,138
660 GOSUB 2250
670 DATA 5,30,141,140,140,140,140,140,140,140,140,140,140,142,32,32,32
680 DATA 32,32,32,141,140,140,140,140,140,140,140,140,140,140,142
690 :
1000 REM Hauptprogramm Diagnose
1010 GOSUB 1830 :REM initialisieren
1020 GOSUB 1110 :REM Ausgabesteuerung
1030 GOSUB 1620 :REM Eingabedarstellung
1040 GOTO 1020 :REM Ruecksprung
1090 :
1100 REM Ausgabesteuerung
1110 aa=a1(abit)
1150 IF INKEY(10) THEN GOTO 1210
1160 IF abit=0 THEN RETURN
1170 GOSUB 1430
1180 abit=abit-1
1190 GOSUB 1470
1200 RETURN
1210 IF INKEY(15) THEN GOTO 1270
1220 IF abit=6 THEN RETURN
1230 GOSUB 1430
1240 abit=abit+1
1250 GOSUB 1470
1260 RETURN
1270 IF INKEY(20) THEN GOTO 1300
1280 a1(abit)=1-a1(abit) :REM umschalten
1290 aa=a1(abit)
1300 IF NOT INKEY(13) THEN aa=1-a1(abit)
1310 bit=abit
1320 REM aa an Position abit ausgeben
1330 ON aa+1 GOTO 1340,1380
1340 GOSUB 20200 :REM Low ausgeben

```



```

1350 LOCATE 16,7+abit
1360 PRINT aus$; :REM Graphiksymbol
1370 RETURN
1380 GOSUB 20100 :REM High ausgeben
1390 LOCATE 16,7+abit
1400 PEN 2 :PRINT ein$; :PEN 1
1410 RETURN
1420 REM Pfeil loeschen
1430 LOCATE 12,7+abit
1440 PRINT CHR$(32);
1450 RETURN
1460 REM Pfeil drucken
1470 LOCATE 12,7+abit
1480 PRINT CHR$(243);
1490 ENV 1,1,18,0,11,0,10
1500 ENT 1,10,2,2
1510 SOUND 129,78,6,7,1,1
1520 RETURN
1590 :
1600 REM Eingabedarstellung
1610 REM Cursor auf erste Druckposition
1620 LOCATE 34,7
1630 FOR bit=0 TO 3
1640 GOSUB 30100 :REM digitale Eingabe
1650 IF ein THEN PEN 2 :PRINT ein$; :PEN 1 ELSE
PRINT aus$;
1660 NEXT bit
1670 GOSUB 30200 :REM X-Wert
1680 GOSUB 30300 :REM Y-Wert
1690 LOCATE 31,12
1700 PRINT USING"####";x;
1710 LOCATE 31,14
1720 PRINT USING"####";y;
1730 RETURN
1790 :
1800 REM Anfangswerte setzen
1810 REM
1820 REM LED-Symbole
1830 ein$=CHR$(231)+CHR$(10)+CHR$(8)
1840 aus$=CHR$(230)+CHR$(10)+CHR$(8)
1850 REM Ausgaben vorsetzen
1860 DIM al(6)
1870 FOR bit=0 TO 6
1880 al(bit)=0
1890 NEXT bit
1900 abit=0
1910 GOSUB 10000 :REM Ein/Ausgabe vorbereiten
1920 PRINT
1930 PRINT "      Programmstop mit 2xESC"
1940 PRINT
1950 PRINT "      +/ nach oben"
1960 PRINT "      f4 umschalten"
1970 PRINT "      f1 Impuls"
1980 PRINT "      f0 nach unten"
1990 SPEED KEY 8,8
2000 RETURN
2190 :
2200 REM Unterprogramm GRAFIKSYMBOLS
2210 REM Parameter der DATA-Zeile:
2220 REM lz=Zahl fuehrender Leerstellen
2230 REM gz=Zahl der Grafiksymbole
2240 REM gz Grafiksymbole (CHR$-Code) bzw. PEN
2250 READ lz : PRINT SPC(lz);
2260 READ gz
2270 FOR i=1 TO gz
2280 READ gc
2290 IF gc<16 THEN PEN gc ELSE PRINT CHR$(gc);
2300 NEXT i
2310 PRINT
2320 RETURN
2330 :
10000 REM Unterprogramm INIT

```

„Init“ für den NOR-Computer.

```

5 CLEAR 1000 :REM FREISPEICHER SCHAFFEN
10000 REM
10010 REM UNTERPROGRAMM INIT
10020 REM VERSION NDR-KLEIN
10030 REM
10040 DIM B(7),C(7)
10050 REM B(I) ZUM SETZEN VON BITS
10060 REM C(I) ZUM LOESCHEN VON BITS
10070 FOR I=0 TO 7
10080 B(I)=2^I : C(I)=255-B(I)
10090 NEXT I
10100 REM ADRESSE DER IOE-KARTE
10110 A=48 :REM IO-PORT 0
10120 OUT A,0 :REM ALLE BITS LOESCHEN
10130 REM DIGITALEINGANGE AUF GLEICHER ADRESSE
10140 D=A
10150 REM MASCHINENPROGRAMMADRESSE FUER ANALOGEINGABE
10160 EX=HEX("FF01") :REM EINGANG POT-X
10170 EY=HEX("FF05") :REM EINGANG POT-Y
10180 REM DIE OBIGEN ADRESSEN HAENGEN VON DEM
10190 REM AKTUELLEN SPEICHERAUSBAU DES NDR-KLEIN
10200 REM COMPUTERS AB; SIE SIND GGF. ZU AENDERN.
10210 REM DER CODE IST BIS AUF DEN ZUGRIFF AUF DAS
10220 REM UEBERGABEBYTE (CODE 32,00,FF IN DER LETZTEN
10230 REM DATAZEILE) VERSCHIEBBAR.
10240 FOR I=0 TO 47
10250 READ CODE$
10260 POKE EX+I,HEX(CODE$)
10270 NEXT I
10280 AUS=0 :REM AUSGABEBITMUSTER
10290 RETURN
10300 DATA 0E,80,18,02,0E,40,06,00
10310 DATA DB,31,A1,28,06,04,20,F8
10320 DATA 05,18,18,06,00,DB,31,A1
10330 DATA 20,06,04,20,F8,05,18,0B
10340 DATA 06,00,DB,31,A1,28,04,04
10350 DATA 20,F8,05,78,32,00,FF,C9
20000 REM
20010 REM UNTERPROGRAMM AUSGABE
20020 REM VERSION NDR-KLEIN
20030 REM
20040 REM EINGABEPARAMETER BIT (0...7)
20050 REM 20100 = EINSPRUNG 'SETZEN'
20100 AUS=AUS OR B(BIT)
20110 OUT A,AUS
20120 RETURN
20190 REM 20200 = EINSPRUNG 'LOESCHEN'
20200 AUS=AUS AND C(BIT)
20210 OUT A,AUS
20220 RETURN
30000 REM
30010 REM UNTERPROGRAMM EINGABE
30020 REM VERSION NDR-KLEIN
30030 REM
30040 REM EINGABE DIGITALWERTE
30050 REM EINGABEPARAMETER BIT (0-7)
30060 REM RUECKGABE '1', WENN HIGH
30070 REM RUECKGABE '0', WENN LOW
30080 REM 30100 = EINSPRUNG
30100 EIN=INP(D) AND B(BIT)
30110 IF EIN<>0 THEN EIN=1
30120 RETURN
30170 REM EINGABE ANALOGWERTE
30180 REM RUECKGABE 0...255
30190 REM 30200 = EINSPRUNG X-WERT
30200 CALL EX :REM MASCHINENPROGRAMM
30210 X=PEEK(EX-1) :REM UEBERGABEBYTE
30220 RETURN
30290 REM 30300 = EINSPRUNG Y-WERT
30300 CALL EY :REM MASCHINENPROGRAMM
30310 Y=PEEK(EY-1) :REM UEBERGABEBYTE
30320 RETURN

```


„Init“ für den Apple.

```

JLIST
10000 :
10010 REM UNTERPROGRAMM INIT
10020 REM VERSION APPLE II
10030 :
10100 REM GAME CONTROLLER PORT ADRESSE ZUR AUSGABE
10110 A = - 16296: REM BASISADRESSE
10120 POKE A,0: POKE A + 2,0: POKE A + 4,0
10130 POKE A + 6,0: REM ALLE BITS LOESCHEN
10140 REM PUSH-BUTTON ADRESSE ZUR EINGABE
10150 D = - 16287
10200 RETURN
20000 :
20010 REM UNTERPROGRAMM AUSGABE
20020 REM VERSION APPLE II
20030 :
20040 REM EINGABEPARAMETER BIT (0...3)
20050 REM 20100 = EINSPRUNG 'SETZEN'
20100 IF BIT < 0 OR BIT > 3 THEN PRINT "UNGUELTIGE BITNUMMER!": RETURN
20110 POKE A + 2 * BIT,1
20120 RETURN
20190 REM 20200 = EINSPRUNG 'LOESCHEN'
20200 IF BIT < 0 OR BIT > 3 THEN PRINT "UNGUELTIGE BITNUMMER!": RETURN
20210 POKE A + 2 * BIT + 1,0
20220 RETURN
30000 :
30010 REM UNTERPROGRAMM EINGABE
30020 REM VERSION APPLE II
30030 :
30040 REM EINGABE DIGITALWERTE
30050 REM EINGABEPARAMETER BIT (0...2)
30060 REM RUECKGABE 'TRUE' WENN LOW
30070 REM RUECKGABE 'FALSE' WENN HIGH
30080 REM 30100 = EINSPRUNG
30100 IF BIT < 0 OR BIT > 2 THEN PRINT "UNGUELTIGE BITNUMMER!": RETURN
30110 EIN = PEEK (D + BIT) > 127
30120 RETURN
30170 REM EINGABE ANALOGWERTE
30180 REM RUECKGABE 0...255
30190 REM 30200 = EINSPRUNG X-WERT
30200 X = PDL (0)
30210 RETURN
30290 REM 30300 = EINSPRUNG Y-WERT
30300 Y = PDL (1)
30310 RETURN

```

J

„Init“ für den IBM-PC.

```

10000 :
10010 REM Unterprogramm INIT
10020 REM Version IBM-PC
10030 :
10040 DIM B(7),C(7) :REM Bitmuster
10050 REM B(I) zum Setzen von Bits
10060 REM C(I) zum Löschen von Bits
10070 FOR I=0 TO 7
10080 B(I)=2^I : C(I)=255-B(I)
10090 NEXT I
10100 REM Printerport-Adresse zur Ausgabe
10110 DEF SEG=64
10120 LPT=1 :REM Druckeranschluß Nr.1 (lpt1:)
10130 A=PEEK(LPT*2+6)+256*PEEK(LPT*2+7)
10140 IF A=0 THEN PRINT "Druckeranschluß nicht
vorhanden!" : BEEP : END
10150 AUS=0
10160 OUT A,AUS
10170 RETURN
20000 :
20010 REM Unterprogramm AUSGABE
20020 REM Version IBM-PC
20030 :
20040 REM Eingabeparameter BIT (0...7)
20050 REM 20100 = Einsprung "Setzen"
20100 AUS=AUS OR B(BIT)
20110 OUT A,AUS

```

```

20120 RETURN
20190 REM 20200 = Einsprung "Löschen"
20200 AUS=AUS AND C(BIT)
20210 OUT A,AUS
20220 RETURN
30000 :
30010 REM Unterprogramm Eingabe
30020 REM Version IBM-PC
30030 :
30040 EINGABE DIGITALWERTE
30050 REM Eingabeparameter BIT (0...3)
30060 REM Rückgabe 'TRUE', wenn high
30070 REM Rückgabe 'FALSE', wenn low
30080 REM 30100 = Einsprung
30100 EIN=STRIG(2*BIT+1)=0
30110 RETURN
30120 :
30170 REM Eingabe Analogwerte
30180 REM Rückgabe 0...255
30190 REM 30200 = Einsprung X-Wert
30200 X=STICK(0)
30210 RETURN
30270 REM 30300 = Einsprung Y-Wert
30280 REM STICK(0) muß aufgerufen werden,
30290 REM um STICK(1) zu triggern.
30300 Y=STICK(0) : Y=STICK(1)
30310 RETURN

```



```

*****
* Maschinenprogramm zum Einlesen von *
* Digital- und Analogwerten *
*****
* Funktion: *
* Eingabe der Digitalwerte E0 bis E3: *
* Der Status der Joystick-Kontakte *
* wird in das Uebergabebyte bei *
* (Startadresse-1) kopiert. Die Rou- *
* tine ist bis auf den Zugriff auf *
* dieses Byte verschiebbar. *
*
* Eingabe der Analogwerte EX und EY: *
* An dem Joystickport wird an Fire 1 *
* und Fire 2 eine von der Potentio- *
* meterstellung abhängige Frequenz *
* angelegt. Das Programm wartet die *
* High-Periode ab und zählt sie aus. *
* Der Zählbereich ist auf 0..255 be- *
* schränkt, um ein 'Hängen' des Pro- *
* gramms bei nicht angeschlossenem *
* Potentiometer zu vermeiden. *
* Die Übergabe des Zahlenwertes er- *
* folgt über das Byte an (Startadres- *
* se-1). Die Routine ist bis auf den *
* Zugriff auf dieses Byte verschieb- *
* bar. *
* Version Z80/Schneider CPC *
*
* Dr.R.Schule 1986
*****
A000 00      EXY: DB 0      ;ÜBERGABEBYTE
A001 1E 01   E1234: LD E,01H ;FLAG FUER DIGITALEINGANG
A003 18 06   EX: LD E,80H   ;MASKE FUER BIT 7
A005 1E 10   JR SETUP
A007 18 02   LD E,40H      ;MASKE FUER BIT 6
A009 1E 20   EY: LD E,40H  ;INTERRUPT SPERREN
A00B F3      SETUP: DI     ;PORT A DES 8255
A00C 01 0E F4 LD BC,0F40EH ;PORT A:=0E
A00F ED 49   OUT (C),C     ;PORT C DES 8255
A011 06 F6   LD B,0F6H    ;EINLESEN
A013 ED 78   IN A,(C)     ;TAPE BITS BELASSEN
A015 E6 30   AND 30H
A017 4F      LD C,A
A018 F6 C0   OR 0C0H      ;BDIR, BC1 =1
A01A ED 79   OUT (C),A    ;STROBE EIN
A01C ED 49   OUT (C),C    ;STROBE AUS
A01E 04      INC B        ;STEUERREGISTER DES 8255
A021 F3 92   LD A,92H    ;CLOW=AUS, B=EIN, MODE 0
A023 ED 79   OUT (C),A    ;CHIGH=AUS, A=EIN, MODE=0
A024 CB F1   SET 6,C      ;BC1 =1
A026 06 F6   LD B,0F6H    ;PORT C DES 8255
A028 ED 49   OUT (C),C    ;PORT C :=01XX0000
A02A 06 F4   LD B,0F4H    ;PORT A DES 8255
A02C CB 43   BIT 1,E      ;DIGITALEINGABE?
A02E 20 25   JR NZ,DIGIN
A030 16 00   WART1: LD D,0 ;ZAEHLER
A032 ED 78   IN A,(C)     ;PORT A EINLESEN
A034 A3      AND A,E      ;MASKIERE BIT
A035 28 06   JR Z,WART2   ;NOCH HIGH?
A037 14      INC D        ;ZAEHLEN
A038 20 F8   JR NZ,WART1+2;UEBERLAUF?
A03A 15      DEC D        ;KORRIGIERE AUF 255
A03B 18 1C   JR AUS      ;AUSGABE
A03D 16 00   WART2: LD D,0 ;ZAEHLER
A03F ED 78   IN A,(C)     ;PORT A EINLESEN
A041 A3      AND A,E      ;MASKIERE BIT
A042 20 06   JR NZ,ZAEHL ;NOCH LOW?
A044 14      INC D        ;ZAEHLEN
A045 20 F8   JR NZ,WART2+2;UEBERLAUF?
A047 15      DEC D        ;KORRIGIERE AUF 255
A048 18 0F   JR AUS      ;AUSGABE
A04A 16 00   ZAEHL: LD D,0 ;ZAEHLER
A04C ED 78   IN A,(C)     ;PORT A EINLESEN
A04E A3      AND A,E      ;MASKIERE BIT
A04F 28 08   JR Z,AUS     ;NOCH HIGH?
A051 14      INC D        ;ZAEHLEN
A052 20 F8   JR NZ,ZAEHL+2;UEBERLAUF?
A054 15      DEC D        ;KORRIGIERE AUF 255
A055 18 02   JR AUS      ;PORT A EINLESEN
A057 ED 50   DIGIN: IN D,(C);STEUERREGISTER DES 8255
A059 C1      AUS: POP BC  ;CLOW=AUS, B=EIN, MODE=0
A05A 3E 82   LD A,82     ;CHIGH=AUS, A=AUS, MODE=0
A05C ED 79   OUT (C),A    ;PORT C DES 8255
A05E 05      DEC B        ;BDIR, BC1 =0
A05F ED 49   OUT (C),C    ;ZAEHLWERT
A061 7A      LDA A,D      ;AN ÜBERGABEBYTE
A062 32 00 A0 LD (0A000H),A;INTERRUPT FREIGEBEN
A065 FB      EI          ;ZURUECK INS BASIC
A066 C9      RET

```

Für die Übergabe von Digitalwerten über den Joystick-Anschluß und der Analogwerte aus der Potentiometerstellung braucht man schnelle, also Maschinenroutinen.

```

*****
* MASCHINENPROGRAMM ZUM EIN- *
* LESEN VON ANALOGWERTEN. *
*****
* FUNKTION: *
* AN EINEM EINGANGSPORT WIRD *
* EINE VON DER POTENTIOME- *
* TERSTELLUNG ABHAENGIGE *
* FREQUENZ ANGELEGT. DAS *
* PROGRAMM WARTET DIE HIGH- *
* PERIODE AB UND ZAEHLT SIE *
* AUS. DER ZAEHLBEREICH IST *
* AUF 0...255 BESCHRAENKT, *
* UM EIN 'HAENGEN' DES PRO- *
* GRAMMS ZU VERMEIDEN. DIE *
* ÜBERGABE DES ZAEHLWERTES *
* ERFOLGT ÜBER DAS BYTE AN *
* (STARTADRESSE-1). DIE ROU- *
* TINE IST BIS AUF DEN ZU- *
* GRIFF AUF DIESES BYTE VER- *
* SCHIEBBAR. *
* VERSION: Z80 / NDR-KLEIN *
*
* DR.R.SCHULE 1986
*****
FF00 00      EXY: DB 0      ;ÜBERGABEBYTE
FF01 0E 80   EX: LD C,80H  ;MASKE FUER BIT 7
FF03 18 02   JR WART1
FF05 0E 40   EY: LD C,40H  ;MASKE FUER BIT 6
FF07 06 00   WART1: LD B,0 ;ZAEHLER
FF09 DB 31   IN A,(31H)   ;LIES PORT
FF0B A1      AND A,C      ;MASKIERE BIT
FF0C 28 06   JR Z,WART2   ;NOCH HIGH?
FF0E 04      INC B        ;ZAEHLER
FF0F 20 F8   JR NZ,WART1+2;UEBERLAUF?
FF11 05      DEC B        ;KORRIGIERE AUF 255
FF12 18 18   JR AUS      ;AUSGABE
FF14 06 00   WART2: LD B,0 ;ZAEHLER
FF16 DB 31   IN A,(31H)   ;LIES PORT
FF18 A1      AND A,C      ;MASKIERE BIT
FF19 20 06   JR NZ,ZAEHL ;NOCH LOW?
FF1B 04      INC B        ;ZAEHLER
FF1C 20 F8   JR NZ,WART2+2;UEBERLAUF?
FF1E 05      DEC B        ;KORRIGIERE AUF 255
FF1F 18 0B   JR AUS      ;AUSGABE
FF21 06 00   ZAEHL: LD B,0 ;ZAEHLER
FF23 DB 31   IN A,(31H)   ;LIES PORT
FF25 A1      AND A,C      ;MASKIERE BIT
FF26 28 04   JR Z,AUS     ;NOCH HIGH?
FF28 04      INC B        ;ZAEHLER
FF29 20 F8   JR NZ,ZAEHL+2;UEBERLAUF?
FF2B 05      DEC B        ;KORRIGIERE AUF 255
FF2C 78      AUS: LD A,B  ;AUSGABE
FF2D 32 00 FF LD (0FF00H),A;AN ÜBERGABEBYTE
FF30 C9      RET

```


Einführung in die Robotik:

Motoren werden gesteuert

In diesem Beitrag geht es nun darum, einen Motor zu steuern – das ist bekanntlich die Grundbedingung, um einen Roboterarm oder eine Werkzeugmaschine zu kontrollieren. Obwohl die Schaltung nicht kompliziert ist, stellt der Aufbau schon einige Anforderungen, denn jeder Fehler kann zum „Wärmetod“ einiger Bauteile führen.



Mit der Verstärkerplatine können wir einen Gleichstrommotor betreiben (und werden es auch tun). Allerdings kann damit der Motor nur ein- und ausgeschaltet werden. Die Drehrichtung läßt sich zwar ggf. durch Vertauschen der Motoranschlüsse wählen. Jedoch ist die Elektronik der Verstärkerplatine nicht in der Lage, danach die Drehrichtung noch zu ändern. Die Verstärkerplatine werden wir also dort einsetzen, wo immer die gleiche Drehrichtung vorliegt, wie bei Förderbändern, Drehtellern usw., oder wo die Stromflußrichtung keine Rolle spielt, wie bei Lampen, Elektromagneten oder Magnetventilen.

Zur wahlweisen Steuerung in beiden Drehrichtungen eines Motors stellen wir hier die Motorplatine vor (Bild 1). Die Motorplatine besteht aus zwei identischen Systemen, von denen jedes wiederum eine Verriegelungslogik und eine Leistungsstufe besitzt. Im Schaltbild (Bild 2) ist nur ein System aufgezeichnet. Beginnen wir mit der Lei-

stungsstufe. Insgesamt spielen hier sechs Transistoren zusammen, zwei Vorverstärker und vier Leistungstransistoren. Werden die Basisanschlüsse der Vorverstärker auf Low gelegt, so sind diese gesperrt. Gleichzeitig sind aber auch die Leistungstransistoren gesperrt. Durch den Motor, der zwischen den Leistungstransistoren angebracht ist, kann kein Strom fließen.

Wird die Basis des linken Vorverstärkers auf High gelegt, steuert er durch; aber auch die beiden zugeordneten Leistungstransistoren werden durchgesteuert. Der pnp-Transistor (links oben) legt den rechten Motorpol an die Versorgungsspannung, der npn-Transistor (links unten) den linken Motorpol an Masse. Der Motor setzt sich in Gang.

Wird dagegen der rechte Vorverstärker auf High gelegt, steuern die beiden rechten Leistungstransistoren durch. Der Stromfluß (im technischen Sinne von + nach -) erfolgt nun vom linken zum rechten Motorpol. Der Gleichstrommotor wird

auf diese Weise umgepolt, d. h. er läuft nun in der anderen Drehrichtung.

Wenn beide Vorverstärker auf High geschaltet werden, steuern alle Transistoren durch und erzeugen damit einen Kurzschluß, der durchaus die Transistoren beschädigen kann. Zwar wird ein sinnvolles Roboterprogramm eine derartige Steuerung, links- und rechts herum zur gleichen Zeit, nicht enthalten. Nur... ein Roboterprogramm muß ja nicht von allem Anfang an richtig sein! Es ist die Aufgabe der Verriegelungslogik, Sie vor einem größeren Transistorverschleiß zu bewahren. Werden an die beiden Steuerleitungen der Verriegelungslogik gleichzeitig High-Pegel gelegt, so sind die beiden Ausgänge auf Low. Der Motor wird somit abgeschaltet.

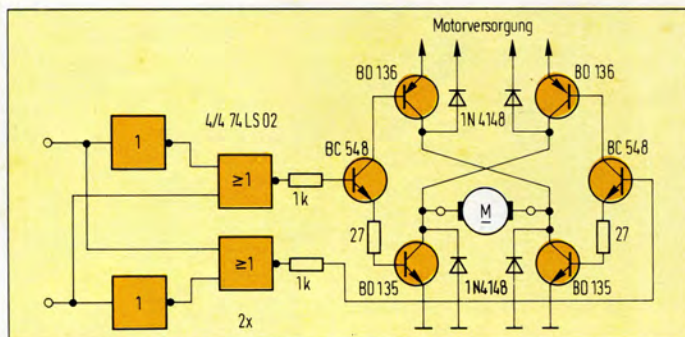
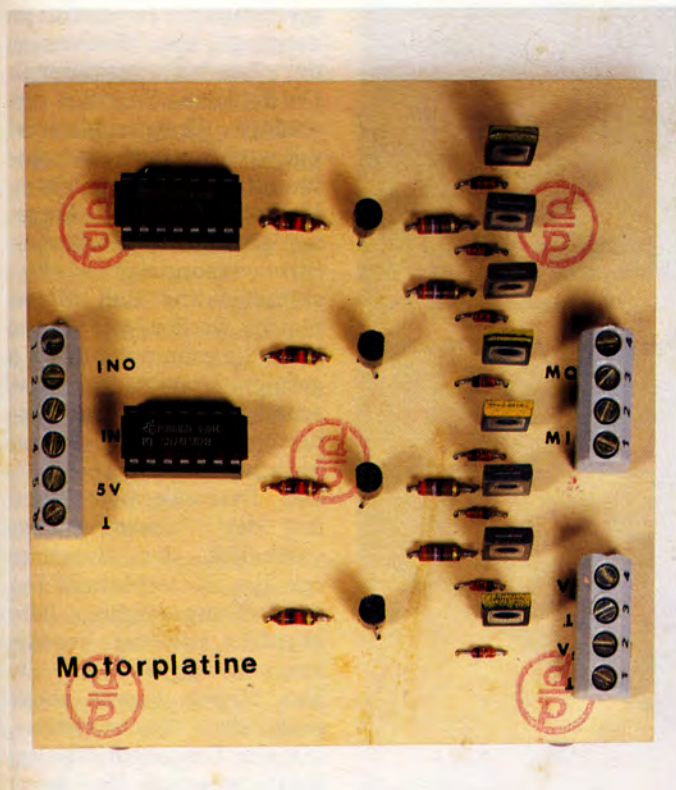
Sie sehen auch, daß zur vollständigen Steuerung eines Gleichstrommotors immer zwei Ausgabeleitungen (und auch zwei Bits) benötigt werden. In unserem Fall steuert die eine Leitung den Rechtslauf, die andere Lei-

tung entsprechend Links-
lauf.

Es sind auch andere Schaltungen denkbar. Sie könnten z. B. die Verstärkerplatine verwenden und um ein Kleinrelais mit zwei Umschaltern ergänzen. Mit einer Ausgabeleitung schalten Sie den Motor ein und aus. Die andere Leitung wird zur Steuerung des Relais benutzt. Dessen Kontaktsatz ist so beschaltet, daß er als Polwender für den Motor dient (Bild 3). In diesem Fall werden also wieder zwei Leitungen benötigt; sie besitzen jedoch eine andere Bedeutung: Ein/Aus und Drehrichtung. Im folgenden werden wir jedoch von der Motorplatine ausgehen. Das spricht nicht gegen die Zuverlässigkeit von Relais, aber elektronisch geht es eleganter.

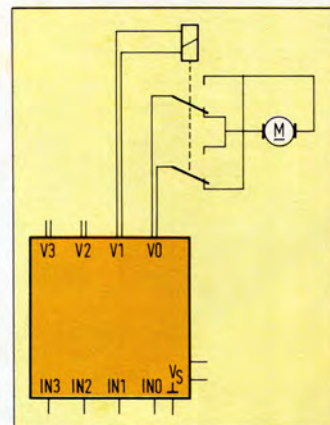
Aufbau der Motorplatine

Obwohl die Schaltung der Motorplatine nicht sonderlich kompliziert ist, stellt der Aufbau schon einige Anforderungen aufgrund der ho-



▲ Bild 2:
Schaltbild der Motorplatine.

◀ Bild 1:
Die Motorplatine steuert
zwei Roboterachsen in beiden
Bewegungsrichtungen.



► Bild 3:
Auch ein Relais kann zur
Steuerung der Drehrichtung
benutzt werden.

hen Zahl der diskreten Bauteile. Achten Sie auf die Orientierung der Transistoren. Schauen Sie auch peinlichst darauf, daß npn- und pnp-Leistungstransistoren jeweils ihre angestammten Plätze erhalten. Gerade beim Vertauschen von npn- und pnp-Transistoren gestaltet sich die Fehlersuche so tückisch, da der Transistor nicht ganz den Geist aufgibt. Er wird in manchen Fällen sogar schalten, aber mehr schlecht als recht. Der Motor kommt nicht auf Touren und der Transistor wird ganz schön heiß.

Auch kommen in der Schaltung reichlich Freilaufdioden vor, die alle ihre Lieblingsrichtung haben. Eine Stufe hat, entsprechend der größeren Zahl von Transistoren, jetzt vier Dioden im Vergleich zu einer in der Verstärkerplatine. Bei einem Verpolen der Diode wird Motorstrom über die Diode fließen. Typisches Kennzeichen für eine verpolte Diode: Ein angeschlossener Motor kann nur in eine Drehrichtung gesteuert werden. Bei Anwahl der entge-

engesetzten Drehrichtung erhitzen sich nur ein Transistor und die verpolte Diode und nehmen vielleicht gar dauerhaften Schaden.

Fehler in der Motorplatine werden sich in aller Regel durch den Wärmetod eines oder mehrerer Bauteile bemerkbar machen, wobei das durchgebrannte Bauteil nochmals die Suche nach der Fehlerursache erschwert. Außerdem sind jene Bauteile nicht wie ICs gesteckt, sondern müssen beim Austausch ausgelötet werden. Daher empfiehlt sich vor dem ersten Anschalten an die Betriebsspannung eine mit doppelter Sorgfalt durchgeführte Sichtkontrolle. Erst wenn Sie hundertprozentig sicher sind, daß alles seine Richtigkeit hat, legen Sie an die Motorplatine die Betriebsspannung an. Die Platine besitzt zwei Eingänge zur Stromversorgung. An den mit +5 V bezeichneten Eingang wird die IC-Versorgung angelegt. Dabei ist ein Amperemeter in die Zuleitung eingeschaltet, um die Stromaufnahme zu messen. Der mit Vs bezeichnete

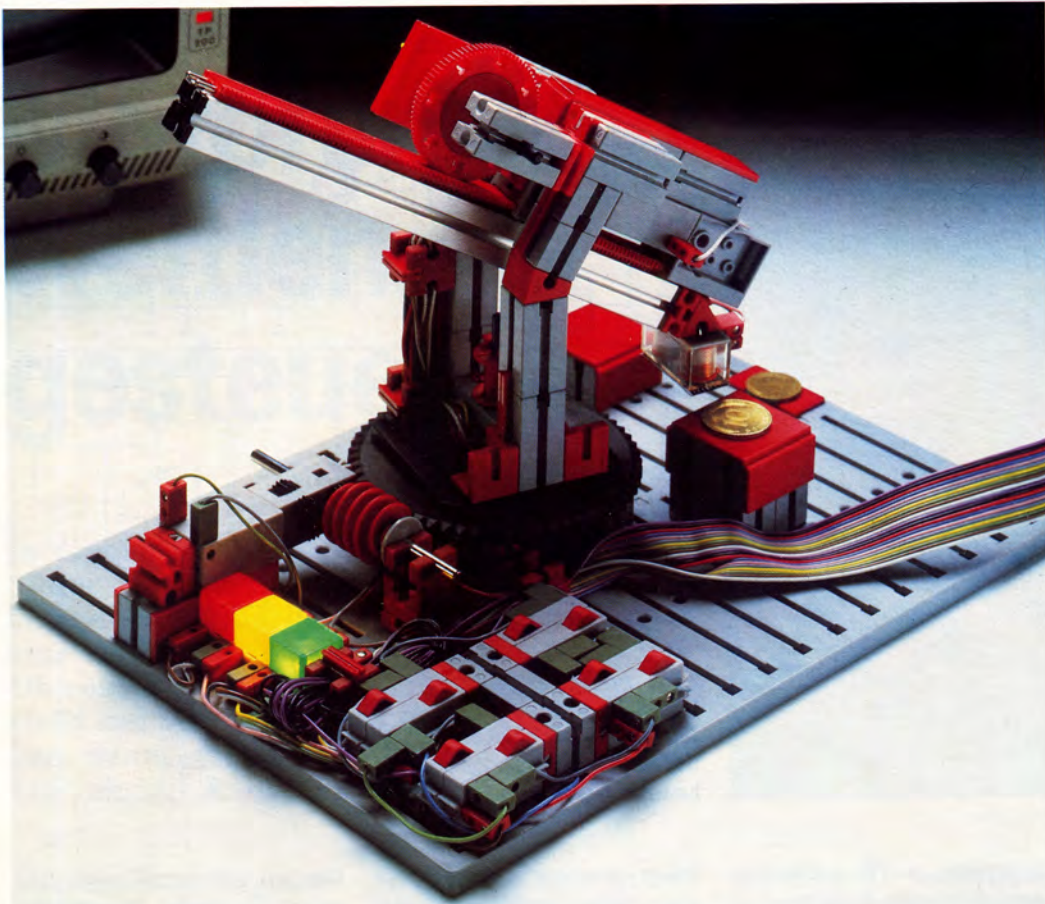
Eingang dient der Einspeisung der Motorspannung und wird zunächst noch nicht angeschlossen. Verbinden Sie die Motorplatine mit den ersten vier Bits der Ausgabeplatine und jene mit dem Computer. Schalten Sie (jetzt erst!) die Geräte ein und messen Sie die Stromaufnahme. Sie darf nicht mehr als 10 mA betragen. Nach dieser Prüfung laden Sie das Diagnoseprogramm. Wählen Sie die Bits der Motorplatine an und schalten Sie sie in beliebiger Kombination – auch in diesem Fall darf die obengenannte Stromaufnahme nicht überschritten werden.

Im nächsten Schritt entfernen Sie das Meßinstrument aus der Zuleitung und verwenden es nun, um am linken Ende der beiden 1-k Ω -Widerstände die Pegel zu messen. Wenn Sie anschließend mit dem Diagnoseprogramm die Ausgänge steuern, sollten die Pegel der Anzeige am Schirm und auf der Ausgabeplatine entsprechen. Es sei denn, Sie schalten beide Eingänge eines Motorkanals auf High. Dann

werden die Ausgänge (dies ist ja Sinn und Zweck der Verriegelungslogik) auf Low liegen.

Nach dieser Prüfung kann im nächsten Schritt auch die Motorversorgung Vs angelegt werden. Es gilt zu der Motorversorgung das, was wir schon bei der Verstärkerplatine gesagt haben. Sie darf bis zu 30 V betragen und ist immer fein säuberlich von der IC-Versorgung getrennt zu halten, auch wenn sie zufällig ebenfalls bei 5 V liegen sollte. Das Meßinstrument wird diesmal als Amperemeter in die Zuleitung Vs eingeschaltet. Die gemessene Stromaufnahme sollte 120 mA bei 7 V nicht übersteigen. Auch wenn Sie mit Hilfe des Diagnoseprogramms die Verstärker aktivieren, darf nicht nennenswert mehr Strom fließen, denn Verbraucher sind ja noch nicht angeschlossen.

Dr. Roland Schulé



Modellroboter aus Fischertechnik computing. Auf ihn beziehen sich die Programmebeispiele.

Einführung in die Robotik:

Werkzeug-Roboter

Ein Roboter ist keineswegs immer menschenähnlich gestaltet. Auch eine vollautomatisierte Drehbank ist ein Roboter, ebenso wie das Krabbeltier, das aus einem Weltraumfahrzeug aussteigt und den Mars untersucht.

Im abschließenden Test betreiben wir einen Verbraucher. Wer die nachstehend geschilderte Werkzeugmaschine baut, kann bei der Stromaufnahme von folgenden Richtwerten ausgehen:
Speisespannung 7 bis 8 V
Stromaufnahme
mit einer Lampe: 80 mA
mit einem Minimotor im Freilauf: 120 mA

mit einem kurzzeitig blockierten Minimotor: 600 mA
Alles in Ordnung? Dann kann es jetzt endlich losgehen!

Die Werkzeugmaschine

Eine Werkzeugmaschine dient der Bearbeitung von Metallteilen und stellt eine

erste Automatisierungsstufe dar. Das zu bearbeitende Rohteil wird einem Drehteller zugeführt. Dies kann von Hand, aber auch von einem Industrieroboter geschehen. Danach wird der Teller um einen bestimmten Drehwinkel gedreht. Die Teile rücken um den gleichen Drehwinkel weiter und kommen unter die nächste Bearbeitungsmaschine. Dann werden alle Werkstücke zur gleichen Zeit bearbeitet, um anschließend wieder einen Takt weiterzurücken. Am Ende des Durchlaufes werden die fertig bearbeiteten Teile dem Drehteller entnommen.

Das Fischertechnik-Modell (Bild 2) ist gegenüber einer realen Werkzeugmaschine stark vereinfacht. Jedoch lassen sich mit ihm die wesentlichen Schritte ebenfalls programmieren: das Umschalten des Drehtellers, das Absenken der Bearbeitungs-

maschine, der Bearbeitungsvorgang, das Hochfahren der Bearbeitungsmaschine. Die in diesen Schritten verwandte Programmieretechnik wird uns auch noch später nützliche Dienste leisten. Modell, Computer, Ausgabeplatine, Eingabeplatine, Stromversorgungen, Verstärkerplatine und Motorplatine werden, wie in dem Blockdiagramm des Bildes 3 gezeigt, miteinander verschaltet. Nochmals zur Erinnerung: Alle Geräte sind dabei ausgeschaltet!

Der Drehtellermotor wird mit der Verstärkerplatine geschaltet, der Hubmotor mit der Motorplatine. Der Bearbeitungsvorgang wird ebenfalls mit der Verstärkerplatine gesteuert. Das Modell besitzt insgesamt drei Positionstaster, die zur Eingabeplatine geführt werden. Zu jedem (komplexen) Robotermodell sollten wir uns eine Belegliste der Ein- und Ausgänge zurechtlegen, die beim Programmieren immer zur Hand ist; etwa in der folgenden Form:

Eingänge:

Eingang 0 – Position Bohrstände oben

Eingang 1 – Position Bohrstände unten

Eingang 2 – Position Drehteller

Eingang 3 – nicht belegt

Ausgänge:

Ausgang 0 – Bohrmaschine nach oben

Ausgang 1 – Bohrmaschine nach unten

Ausgang 2 – nicht belegt

Ausgang 3 – nicht belegt

Ausgang 4 – Drehtellermotor

Ausgang 5 – Bearbeitungsmaschine

Ausgang 6 – nicht belegt

Ausgang 7 – nicht belegt

Befassen wir uns zunächst mit der Aufwärtsbewegung. Das Schema zeigt den Programmablaufplan in Bild 4. Der Motor wird in der Fahrtrichtung nach oben eingeschaltet und anschließend der Eingangskanal 0 überprüft. Solange der Taster nicht betätigt ist, wird

Foto: Fischertechnik

die Abfrage wiederholt. Im anderen Fall wird der Motor abgeschaltet.

Was wir hier kennenlernen, ist ein Grundprinzip der Robotik: Zu jeder Bewegung gehört nicht nur die Steuerung des Antriebs, sondern auch die Überwachung der Position.

Wer gerne die Programmschritte einzeln austesten möchte, kann (zusätzlich zu den Ein- und Ausgabeunterprogrammen) die Zeilen 1100 bis 1200 abtippen. Anschließend werden mit dem Direktkommando GOSUB 10000

die Ein- und Ausgänge eingerichtet. Die Ausgänge werden mit dem Befehl auch gleich ausgeschaltet. Derart vorbereitet kann nun die Versorgungsspannung an die Motor- und Verstärkerplatine angelegt werden. Dabei darf sich kein Motor in Bewegung setzen. Mit GOSUB 1100 : POKE A,0 wird nun die Aufwärtsbewegung aufgerufen. Der erste Teil des Kommandos stellt den eigentlichen Testaufruf dar. Der zweite Teil des Kommandos schaltet sicherheitshalber alle Ausgänge aus. Man weiß ja nie ... Dieser Teil des Kommandos muß bei den CPCs natürlich OUT a,0 lauten.

Halten Sie trotzdem ihren Finger bei der Not-Austaste bereit, wenn Sie das letztgenannte Kommando eingeben. Der Hubmotor könnte in die falsche Richtung laufen. In diesem Fall müssen Sie sofort die Not-Austaste drücken und festhalten, während Sie das Programm stoppen. Die Motoranschlüsse werden in diesem Fall vertauscht, so daß er im nächsten Versuch in die korrekte Richtung läuft. Desgleichen stoppen Sie so auch den Motorlauf, wenn bei Erreichen der oberen Endposition der Motor nicht stehenbleibt. Prüfen Sie in diesem Fall die richtige Verkabelung der Eingangsleitungen. Auf die gleiche Weise wie oben wird die Abwärtsbewegung programmiert. Le-

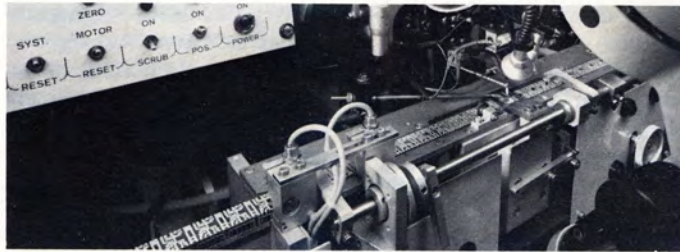


Bild 1: Die zu bearbeitenden Teile laufen auf einem Band von einem Bearbeitungspunkt zum nächsten.

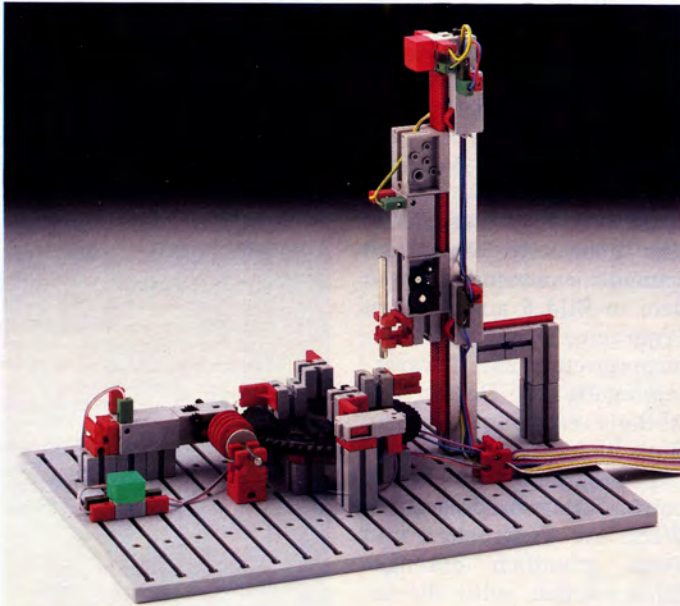


Bild 2: Das Modell der Werkzeugmaschine mit zwei Motoren und drei Positionstastern.

diglich die Ein- und Ausgabelösungen sind ausgetauscht.

Der Drehtellermotor erfordert ein anderes Vorgehen. Würden wir ein vergleichbares Programmstück verwenden, so würde sich der Motor zwar für Bruchteile einer Sekunde in Bewegung setzen, sofort aber wieder stehenbleiben, da der Positionstaster immer noch von der letzten Positionierung betätigt ist. Wer gerne experimentiert, sollte diesen Programmierfehler ruhig einmal bewußt einbauen und sich die Wirkung anschauen. Wenn Sie irgendwann später wieder mal auf einen stotternden Motor stoßen, werden Sie sich an dieses Experiment erinnern und vielleicht so leichter den Fehler finden.

Das in Bild 5 dargestellte Vorgehen verhilft zum Erfolg. Der Motor wird zunächst eingeschaltet. Anschließend wird gewartet, bis der Taster *nicht* mehr betätigt ist. Danach folgt eine weitere Schleife, in der gewartet wird, bis der Taster *wieder* betätigt wird. Dieses

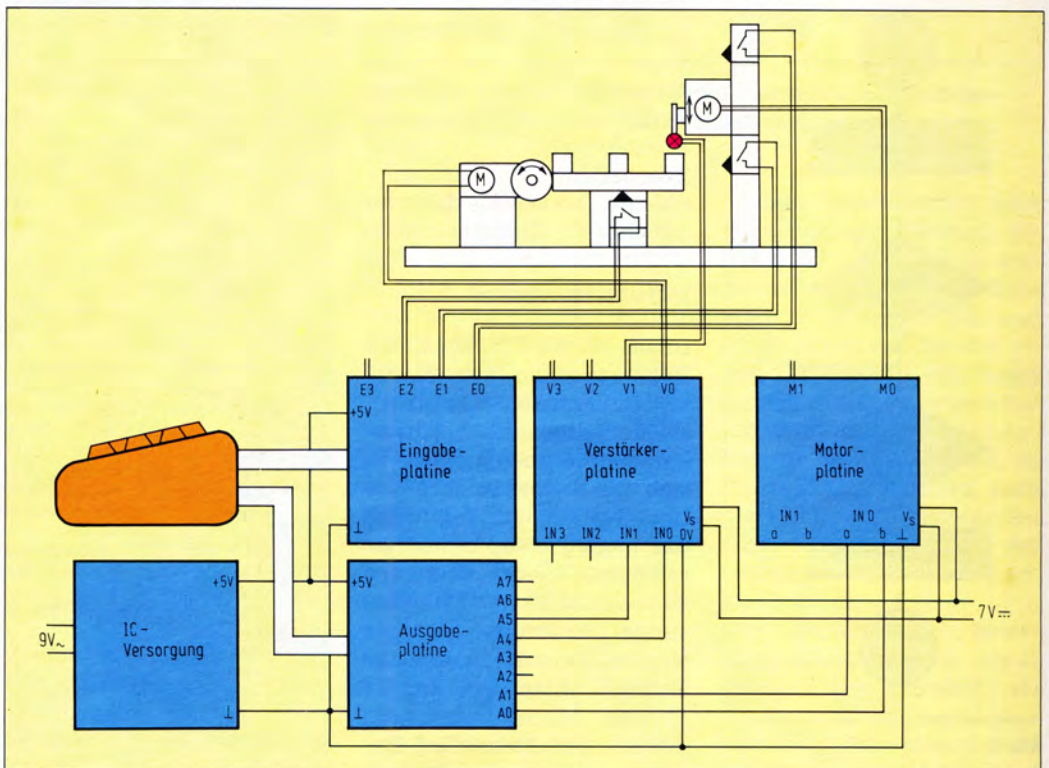


Bild 3: Blockschaltbild des Modells Werkzeugmaschine.

Verfahren wird in der Programmierung regelmäßig zur Erkennung von Impulsflanken eingesetzt. Auch dieses Programmstück (Zeilen 1300 bis 1420) können Sie wieder für sich testen, indem sie im Direktmodus

erst das INIT-Unterprogramm GOSUB 10000 und dann anschließend das Unterprogramm zur Drehtellersteuerung aufrufen: GOSUB 1300 : POKE A,0 Das verbleibende Unterprogramm, das dem eigentlichen Bearbeitungsvorgang gewidmet ist, muß den Bearbeitungsmotor für eine bestimmte Zeit einschalten. Es benutzt, wie schon bei unseren Experimenten mit der Verkehrsampel, eine FOR...NEXT-Schleife zur Erzeugung der Verweildauer.

Mit diesen Ingredienzien ist das komplette Programm schnell zusammengestellt. In dem in Bild 6 aufgelisteten Programm ist noch ein Unterprogramm zur Parametereingabe vor Beginn des Ablaufs eingefügt worden. Eine kleine Grafik ergänzt das Programm wie schon beim Diagnoseprogramm. Wenn die Vorarbeiten und Tests gründlich durchgeführt wurden, sollte die Inbetriebnahme keine Probleme bereiten. Zum Abschluß wollen wir die Frage stellen, warum wohl das Hauptprogramm die einzelnen Unterprogramme in der Reihenfolge

besitzt das hier eingesetzte Modell nur zwei Achsen. Wozu eigentlich fünf, sechs oder noch mehr Achsen eines Roboters? Der uns umgebende Raum hat drei Dimensionen: Höhe, Länge und Breite. Wenn wir uns

anstelle der bekannten Formen des Industrieroboters einen Portalkran vorstellen, sehen wir leicht ein, daß zu jeder Raumdimension ein getrennter Antrieb gehört (Bild 7). Solche Roboter in Form eines Portalkrans gibt

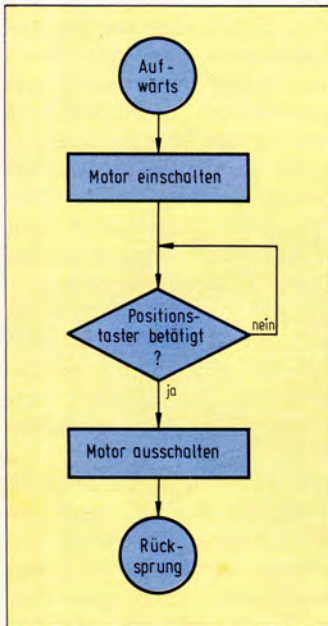


Bild 4:
Programmablaufplan der Aufwärtsbewegung der Werkzeugmaschine.

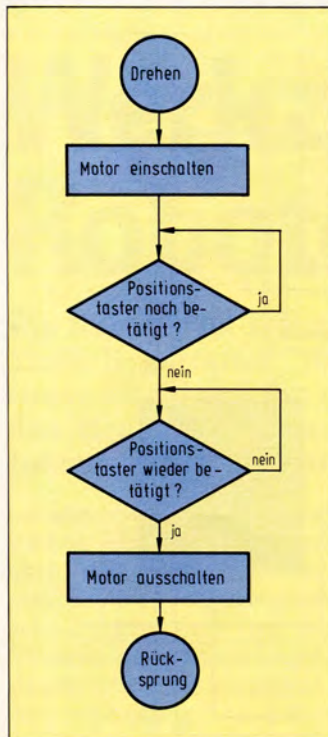


Bild 5:
Programmablaufplan der Drehbewegung des Drehtellers.

Industrieroboter

In unserer Einführung in die Robotik sollen nun Hard- und Software eines richtigen (Modell-)Roboters drankommen. Dabei handelt es sich um keine Science-fiction-Gestalt, sondern um das Modell eines Industrieroboters. Wieder sind, der Übersichtlichkeit zuliebe, einige Abstriche gemacht worden: Versteht man in der Technik unter dem Begriff Roboter einen Automaten, der in mindestens fünf Bewegungsachsen frei steuer- und programmierbar ist, so

```

110 REM STEUERUNG DER WERKZEUGMASCHINE
120 REM
130 REM (C) DR.R.SCHULE, 1986
140 REM
150 REM FUNKTION: DIE WERKZEUGMASCHINE
160 REM WIRD DURCH EINEN WIEDERHOLENDEN
170 REM BEWEGUNGSABLAUF MIT DEN PHASEN:
180 REM ANHEBEN DER BOHRMASCHINE
190 REM DREHEN DES TISCHES
200 REM ABSSENKEN DER BOHRMASCHINE
210 REM BEARBEITUNGSVORGANG
220 REM GESTEUERT.
230 REM
240 REM BELEGUNG DER AUSGÄNGE:
250 REM AUSGANG 0 - ANHEBEN
260 REM AUSGANG 1 - ABSSENKEN
270 REM AUSGANG 2 - NICHT BENUTZT
280 REM AUSGANG 3 - NICHT BENUTZT
290 REM AUSGANG 4 - DREHTERMOTOR
300 REM AUSGANG 5 - BOHRMASCHINE
310 REM AUSGANG 6 - NICHT BENUTZT
320 REM AUSGANG 7 - NICHT BENUTZT
330 REM
340 REM BELEGUNG DER EINGÄNGE:
350 REM EINGANG 0 - POSITION OBEN
360 REM EINGANG 1 - POSITION UNTEN
370 REM EINGANG 2 - POSITION DREHTER
380 REM EINGANG 3 - NICHT BENUTZT
390 :
400 REM TITELMELDUNG
410 POKE 53281,1 :REM HINTERGRUND WEISS
420 PRINT CHR$(147); :REM BILD LOESCHEN
430 PRINT CHR$(31); :REM SCHRIFT BLAU
440 PRINT : PRINT
450 PRINT "WERKZEUGMASCHINE";
460 GOSUB 2150 :REM DRUCKE GRAFISYMBOLS
470 DATA 9,3,152,161,31
480 PRINT "-----";
490 GOSUB 2150
500 DATA 8,3,152,176,161
510 GOSUB 2150
520 DATA 23,3,182,179,161
530 GOSUB 2150
540 DATA 24,2,173,161
550 GOSUB 2150
560 DATA 19,7,162,32,162,32,162,32,161
570 GOSUB 2150
580 DATA 16,17,18,32,32,146,28,113,144,18,99,99,99,99,
99,146,152,32,161
590 GOSUB 2150
600 DATA 16,10,165,167,32,165,165,32,167,167,32,161
610 GOSUB 2150
620 DATA 15,15,163,163,163,163,163,163,163,163,163,
163,163,163,17,17,31
630 :
1000 REM HAUPTPROGRAMM WERKZEUGMASCHINE
1010 GOSUB 1910:REM ANFANGSWERTE SETZEN
1020 GOSUB 1810:REM BILD TEILW. LOESCHEN
1030 GOSUB 1110:REM ANHEBEN
1040 GOSUB 1310:REM DREHEN
1050 GOSUB 1510:REM ABSSENKEN
1060 GOSUB 1710:REM BEARBEITEN
1070 GOTO 1020:REM RUECKSPRUNG
1080 :
1100 REM UNTERPROGRAMM ANHEBEN
1110 PRINT "BOHRKOPF ANHEBEN"
1120 BIT=0
1130 GOSUB 20100 :REM MOTOR AUFWAERTS
1140 BIT=0
1150 GOSUB 30100 :REM ENDOPOSITION ?
1160 IF EIN THEN GOTO 1150
1170 BIT=0
1180 GOSUB 20200 :REM MOTOR AUS
1190 RETURN
    
```

Bild 6: Programm der Werkzeugmaschine C64.

es übrigens auch; sie werden vor allem dort eingesetzt, wo hohe Gewichte zu bewegen sind oder hohe Anpreßdrücke aufzubringen sind. Soll der Roboter kleiner, schneller und flexibler sein, so wird man Vorschubbewe-

gungen mit Dreh- und Kippbewegungen kombinieren oder gänzlich auf Drehbewegungen zurückgreifen (Bild 8). Jedoch gilt auch hier, daß die Positionierung in den drei Raumdimensionen drei Bewegungsachsen

```

1200 :
1300 REM UNTERPROGRAMM DREHEN
1310 PRINT "RUNDTISCH DREHEN"
1320 BIT=4
1330 GOSUB 20100 :REM TISCHMOTOR AN
1340 BIT=2
1350 GOSUB 30100 :REM TASTER FREI?
1360 IF NOT EIN THEN GOTO 1350
1370 GOSUB 30100 :REM TASTER BETÄTIGT?
1380 IF EIN THEN GOTO 1370
1390 BIT=4
1400 GOSUB 20200 :REM TISCHMOTOR AUS
1410 RETURN
1420 :
1500 REM UNTERPROGRAMM ABSENKEN
1510 PRINT "BOHRKOPF ABSENKEN"
1520 BIT=1
1530 GOSUB 20100 :REM MOTOR ABWÄRTS
1540 BIT=1
1550 GOSUB 30100 :REM ENDOPOSITION ?
1560 IF EIN THEN GOTO 1550
1570 BIT=1
1580 GOSUB 20200 :REM MOTOR AUS
1590 RETURN
1600 :
1700 REM UNTERPROGRAMM BEARBEITEN
1710 PRINT "WERKSTÜCK BEARBEITEN"
1720 BIT=5
1730 GOSUB 20100 :REM BOHRMOTOR EIN
1740 FOR Z=0 TO ZEIT
1750 NEXT Z
1760 BIT=5
1770 GOSUB 20200 :REM BOHRMOTOR AUS
1780 RETURN
1790 :
1800 REM UNTERPROGRAMM BILD LOESCHEN
1810 PRINT LEFT$(VT$,16);
1820 FOR I=1 TO 5
1830 PRINT " "
1840 NEXT I
1850 PRINT LEFT$(VT$,16);
1860 RETURN
1890 :
1900 REM UNTERPROGRAMM ANFANGSWERTE
1910 VT$=CHR$(19):REM CURSOR LINKS OBEN
1920 FOR I=1 TO 25
1930 VT$=VT$+CHR$(17):REM CURSOR RUNTER
1940 NEXT I
1950 PRINT LEFT$(VT$,16);
1960 PRINT "BEARBEITUNGSZEIT EINGEBEN!"
1970 PRINT "1 SEKUNDE IST UNGEFAHR 500"
1980 INPUT ZEIT
1990 GOSUB 10000 :REM EIN/AUSGABE VORBEREITEN
2000 PRINT "VERSORGUNGSSPANNUNG ANLEGEN!"
2010 PRINT "WERKZEUGMASCHINE BEREIT? (J)";
2020 GET K$
2030 IF K$(">")="J" THEN GOTO 2020
2040 PRINT LEFT$(VT$,24);
2050 PRINT "PROGRAMMSTOP MIT STOPTASTE!"
2060 RETURN
2070 :
2100 REM UNTERPROGRAMM GRAFIKSYMBOLE
2110 REM PARAMETER DER DATAZEILE:
2120 REM LZ=ZAHL FUEHRENDER LEERZEICHEN
2130 REM GZ=ZAHL DER GRAFIKSYMBOLE
2140 REM GZ GRAFIKSYMBOLE (CHR$-CODE)
2150 READ LZ : PRINT SPC(LZ);
2160 READ GZ
2170 FOR I=1 TO GZ
2180 READ GC : PRINT CHR$(GC);
2190 NEXT I
2200 PRINT
2210 RETURN
READY.

```

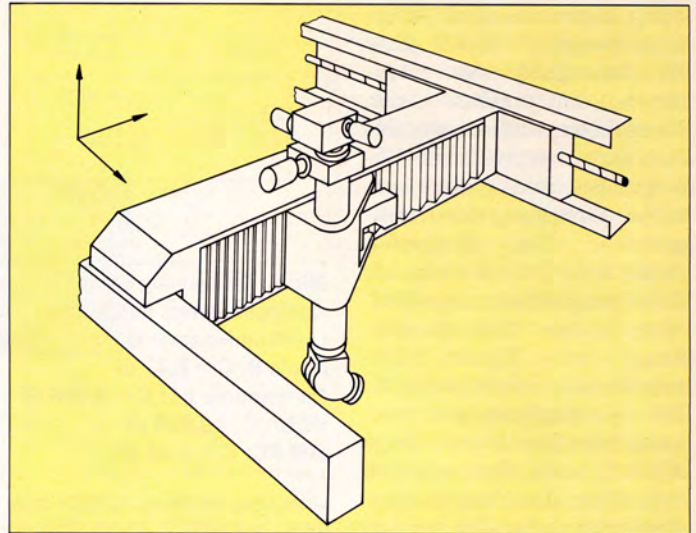


Bild 7: Skizze eines Portalroboters. Die drei Hauptbewegungen sind linear.

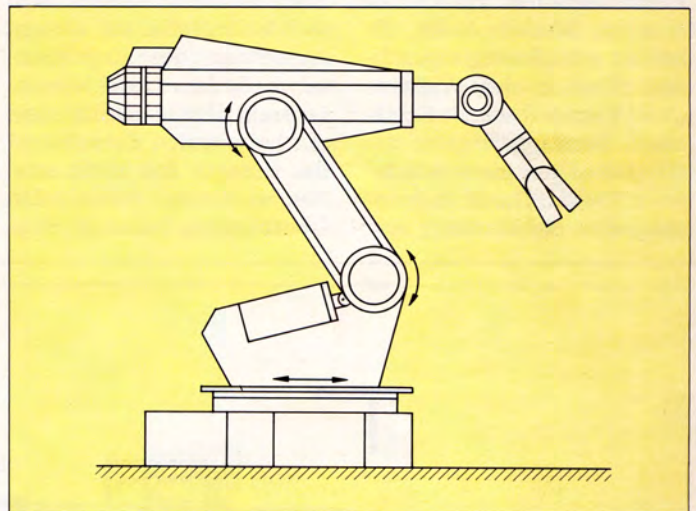


Bild 8: Der Knickarmroboter besitzt ebenfalls drei Hauptbewegungsachsen, die jedoch auf Drehgelenken beruhen.

erfordert. Die Zusammenhänge zwischen Drehbewegung und Greifarmposition in den Raumdimensionen werden allerdings im letzten Fall nicht einfach zu überblicken sein. Eine für den Nichtfachmann komplizierte, aber mathematisch vollständige Beschreibung erfordert den Einsatz von sogenannten Transformationsmatrizen. Keine Sorge, wir werden bei der Programmierung unseres Roboters kein solches Geschütz auffahren müssen.

Sie sollten nur in Erinnerung behalten, daß die Bewegung in den drei Raumdimensionen in jedem Falle drei Bewegungsachsen er-

fordert. Diese werden auch Hauptbewegungsachsen genannt. Doch wozu nun insgesamt fünf und mehr Achsen? Es ist zwar gut und schön, wenn ein Roboter einen Gegenstand von Punkt A nach Punkt B transportieren kann. Jedoch ist es fraglich, ob der Gegenstand, an Punkt B angekommen, auch die richtige Einbaulage besitzt. Vielleicht müßte der Gegenstand gedreht und gekippt werden.

Die zusätzlichen Bewegungsachsen dienen somit weniger dem Transport als der Orientierung des Gegenstandes. Um einmal den Menschen zum Vergleich heranzuziehen: Hüftdre-

hung, Oberarm- und Unterarmbewegung bei einer Greifbewegung des Menschen entsprechen den Hauptbewegungsachsen des Roboters. Die weiteren Bewegungsachsen zeigen sich in der Bewegung des Handgelenks. Das Zugreifen selbst wird jedoch nicht als Bewegungsachse mitgezählt. Eine kleine Nebenbemerkung am Rande: Die menschliche Hand weist 22 Bewegungsachsen auf und kann damit so komplizierte Abläufe bewältigen wie das Schreiben, das Führen eines Werkzeugs oder das Spielen eines Musikinstruments. Von dieser hohen Zahl von Freiheitsgraden zurück zu unserem Modellroboter. Er besitzt zwei Bewegungsachsen: Drehen des Aufbaus und Vorschub der Greifhand. Damit bewegt er die Greifhand auf dem Mantel eines Kegels. Durch Begrenzungen in der Drehung und

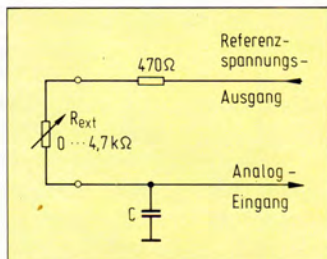


Bild 9: Anpassung eines Analogeingangs durch den Ladekondensator C.
Apple II: C = 0,47 µF
Commodore 64: C = 0,068 µF
VC20: C = 0,068 µF
IBM PC: C = 0,47 µF.

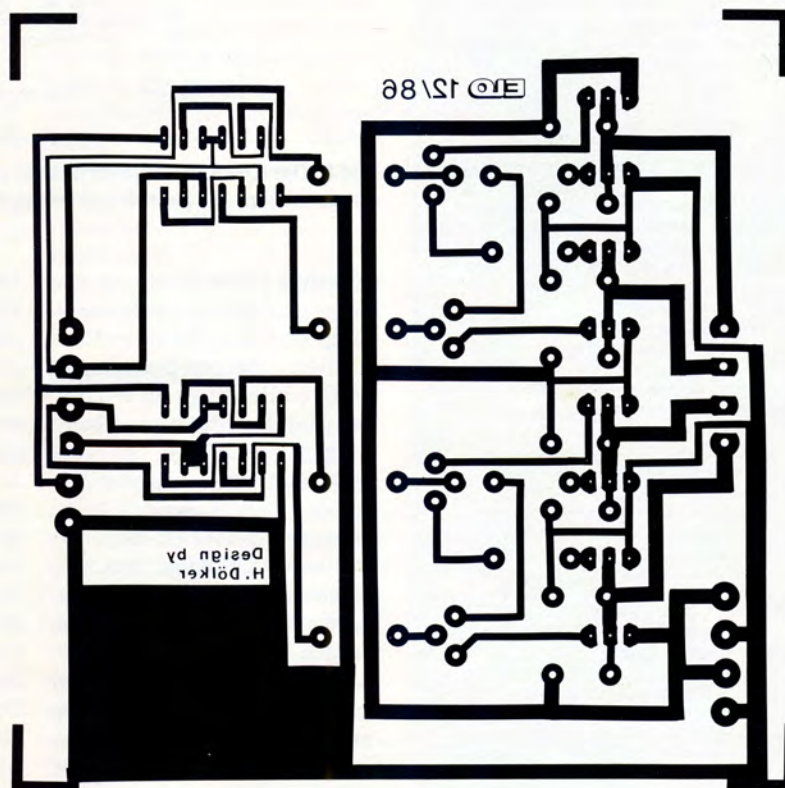
dem Vorschub kann jedoch nicht der ganze Kegelmantel überstrichen werden. Alle Zielpositionen müssen sich somit ebenfalls auf diesem Ausschnitt des Kegelmantels befinden. Diese Vereinfachung der Geometrie kann uns beim ersten Kontakt mit der Robotik nur recht sein. Sie wird uns helfen, den Überblick zu behalten. Spä-

ter wenden wir uns dann komplexeren Aufbauten zu. Auch die Greifhand, in der Robotik ein komplexes feinmechanisches Gebilde, wurde in dem Modell sinnvoll vereinfacht. Ein Elektromagnet ist in der Lage, Eisen- teile ganz nach Bedarf festzuhalten oder loszulassen. Variationsmöglichkeiten bestehen auch. Wer möchte, kann den Magneten gegen eine Glühbirne austauschen und so einen Schweißvorgang simulieren. Oder aber eine Meßsonde anbringen oder ...

Positionsmessung am Roboter

Wir hatten es schon bei der Programmierung der Werkzeugmaschine beobachtet: Zu jeder Positionierung gehört neben der *Steuerung* des Antriebs die *Messung* der Position. Die vorher besprochene Werkzeugmaschi-

ne hat Mikroschalter verwendet. Damit konnte das Erreichen einer Endlage oder einer festen Zwischenposition vermeldet werden. Bei dem Roboter erhebt sich die Forderung, auch *beliebige* Zwischenpositionen ansteuern zu können. Vorteilhaft ist wieder eine unabhängige Messung der Position. In dem Fischertechnik-Aufbau werden zur Messung beliebiger Zwischenpositionen Potentiometer eingesetzt, die von der Robotermechanik angetrieben werden. Aus der Stellung des Potentiometers und hiermit aus seinem elektrischen Widerstand kann die Position der zugeordneten Roboterachse abgelesen werden. Diese Methode der Positionsbestimmung kommt zwar in der Robotik vor, jedoch werden andere Systeme häufiger eingesetzt; allen voran Winkelschrittgeber. An die Stelle der Wider-



standsschicht und des Schleifers tritt hierbei eine Glasplatte mit ringförmig aufgedruckten Sektormustern: der innerste Ring mit einem durchscheinenden und einem undurchsichtigen Sektor, der nächste mit zwei durchscheinenden und zwei undurchsichtigen Sektoren. Der nächste Ring ist in acht Sektoren unterteilt, der darauffolgende in sechzehn und so fort, je nach gewünschter Winkelauflösung. Sie erkennen aus diesen Aufteilungen wiederum das Prinzip der binären Codierung, hier des Drehwinkels. Pro Ring wird im einfachsten Fall eine Lichtschranke verwendet. Das Bitmuster der Lichtschranken kann direkt als Digitalwert weiterverarbeitet werden. Die Positioniergenauigkeit mit Hilfe von Potentiometern kann sicherlich nicht mit jener von Winkelschrittteilgebern mithalten. Dafür

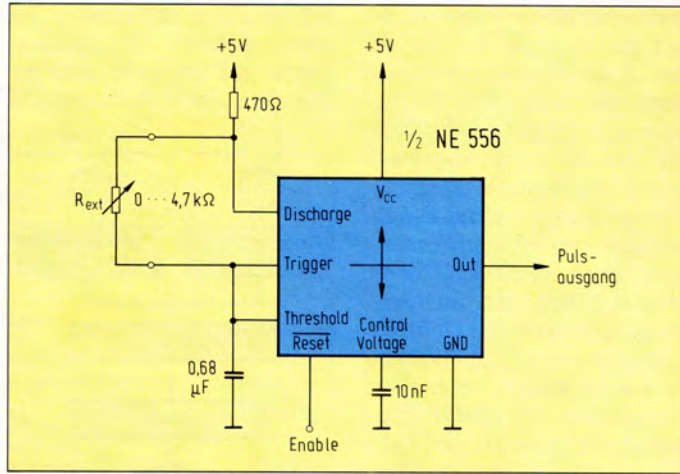


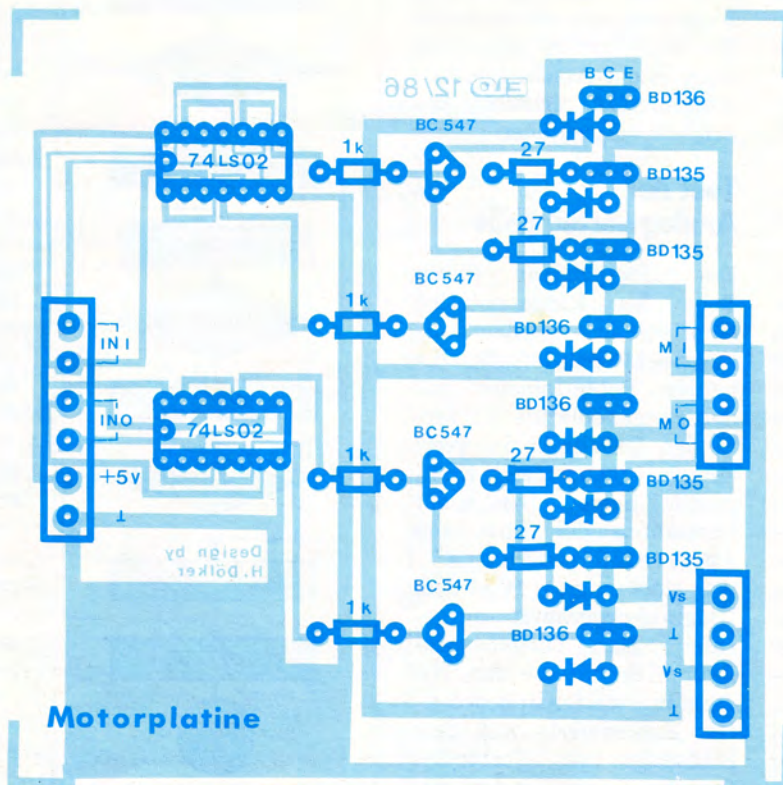
Bild 10: Oszillator-Schaltung mit dem NE556 als Analogeingang.

verbleiben Potentiometer im Rahmen des Hobbytats. Beide Positionsmeßverfahren zählen übrigens zu der Gruppe der Absolutgeber. Dies bedeutet, daß an dem Ausgang des Gebers direkt der Drehwinkel abgelesen werden kann. Dies ist nicht bei allen Systemen so: Wer-

den z. B. Schrittmotoren oder Umdrehungszähler eingesetzt, so läßt sich die Stellung der Roboterachse nur aus der Vorgeschichte ermitteln. Ein Beispiel: Start bei Position 0, 1200 Impulse vorwärts, 300 Impulse zurück, noch mal 405 Impulse zurück, 227 Impulse vor. Wo

steht der Roboter jetzt? Gewiß läßt sich die Position ausrechnen; doch wehe, der Elektronik ist einmal ein Impuls verlorengegangen, z. B. der 299igste Impuls im zweiten Schritt. Er wird in der Bilanz solange fehlen, bis wieder einmal die unabhängig zu ermittelnde Nullposition angefahren wird. Diesem Nachteil stehen jedoch die geringeren Kosten im Vergleich zum Winkelschrittteilgeber gegenüber. Wegen der Positionsbestimmung durch Auf- und Abzählen werden solche Positionsmelder auch Inkrementalgeber genannt.

Doch zurück zu unserem Absolutgeber Potentiometer. Was uns noch fehlt, ist die Umsetzung des elektrischen Widerstandes in Zahlenwerte des Computers. Bei einigen Computern (Apple II, Commodore 64, IBM PC) ist dies kein Problem, da sie für diese Aufgabe ausgerü-



stet sind. Sie besitzen Eingänge für Drehregler, die lediglich je nach Computer auf den Wertebereich der verwendeten Potentiometer eingestellt werden müssen. Wir erinnern nochmals an den Aufbau der Eingabeplatine. In **Bild 9** geben wir einen Ausschnitt der Schaltung wieder, der die Anpassung eines Drehreglereingangs durch den Kondensator C zeigt. Dessen Wert ist je nach Computertyp der Tabelle zu entnehmen.

Bei Computern ohne Drehreglereingang müssen wir selbst einen solchen erzeugen. Dies erfolgt durch eine Oszillatorschaltung mit dem Universal-Zeitgeber-IC NE556 (der NE556 besteht gerade aus zwei Systemen des bekannteren NE555). **Bild 10** zeigt den entsprechenden Ausschnitt der Schaltung der Eingabeplatine. Die Periodendauer der Schwingung hängt von dem externen Widerstandswert ab:

$$t_{\text{low}} = 0.693 R_{\text{ext}} C$$

$$t_{\text{high}} = 0.693 (R_{\text{ext}} + 470) C$$

$$t_{\text{gesamt}} = 0.693 (2 R_{\text{ext}} + 470) C$$

Die entstehende Rechteckschwingung führen wir den noch freien Steuerknüppel-eingängen zu.

Auch diese Steuerknüppel-eingänge gestalten wir wieder abschaltbar. Die Freigabeleitung wird mit dem Reset-Eingang des Zeitgebers verbunden.

Es gilt nun, die Periodendauer der Schwingung zu bestimmen. Insbesondere interessiert uns t_{low} (das Signal ist nach dem Inverter dann high!), denn für diese Komponente ist der Zusammenhang mit R_{ext} besonders einfach. Zur Bestimmung verwenden wir ein kurzes Maschinenprogramm, das den Eingang testet. Das Programm wartet in einer Schleife zunächst, falls gerade noch ein High-Signal, das zu einem unbekannten Zeitpunkt vorher begonnen haben kann, am Eingang ansteht. Danach gilt es, die Low-Phase abzuwarten. Mit deren Ende beginnt der ei-

gentliche Meßzyklus, wobei auf das erneute Erscheinen des Low-Signals gewartet wird. Bei allen drei Wartephase wird mitgezählt, wie oft der Eingang abgefragt wurde. Spätestens nach 256 Überprüfungen ohne Pegeländerung wird das Unterprogramm verlassen, denn offensichtlich ist an dem Eingang kein Potentiometer angeschlossen (unendliche Periodendauer). Schwingt jedoch der Oszillator ordnungsgemäß, so ist der Zählerstand in der dritten Phase, der vollständigen High-Periode, maßgeblich. Er wird in das BASIC-Programm zurückgegeben, denn er ist letztlich ein Maß für die Potentiometerstellung.

Das Maschinenprogramm wird durch das Unterprogramm INIT geladen, das im letzten Heft beschrieben wurde. Wir senden Ihnen gern den Quelltext des Programms (Z80-Code) für den NDR-Klein-Computer zu, da es in diesem Falle besonders übersichtlich ist. Der Vergleich mit dem Programmablaufplan in **Bild 11** zeigt ebenfalls die einfache Struktur, die es sicherlich auch Nicht-Assembler-Programmierern erlaubt, die Abläufe zu verstehen.

Test der Analogwerteingabe

Zur Überprüfung dieses Teils der Eingabeplatine empfiehlt sich, vor dem Aufbau des Modells lediglich die beiden Potentiometer mit der Eingabeplatine zu verbinden. Die zur Analogwert-eingabe gehörigen Kommandos liegen schon in der Sammlung der Ein- und Ausgabeunterprogramme vor; sie werden z. B. in dem Diagnoseprogramm aufgerufen. Das Diagnoseprogramm druckt die den Widerstandswerten zugeordneten Zahlenwerte auf dem Bildschirm aus. Verbinden Sie die Potentiometer mit der Eingabeplatine und starten Sie das Programm. Nehmen Sie die Potentiome-

ter zur Hand und drehen Sie deren Achsen. Die Zahlenwerte sollten sich ändern. Bei dem einen Endanschlag

sollten Sie nahezu den Wert 0 erhalten. Da in Serie mit dem Potentiometer ein Widerstand von 470 Ω liegt, der einen Kurzschluß des Schaltkreises verhindert, wird die Null nicht ganz erreicht; Werte um 20 werden sich einstellen. Am anderen Ende des Potentiometerwinkels wird der Wert nahezu bei 255 liegen. Auch hier wurde ein Sicherheitsabstand eingehalten, da manche Umwandelprogramme keine Zahlen größer als 255 erzeugen können. Würde ein ganzer Winkelbereich vor dem Anschlag des Potentiometers Zahlenwerte von 255 erzeugen, so bestünde die Gefahr, daß eine Positions-messung nicht eindeutig und deshalb die Steuerung des Roboters unzuverlässig wäre. Die Dimensionierung ist daher so angelegt, daß in aller Regel Zahlenwerte um 200 im Endanschlag erzielt werden.

Zum ersten Test sollten Sie die Platine mit dem angegebenen Wert des Kondensators bestücken. Wenn Sie nicht die genannten Resultate erhalten, heißt es wieder, die Platine zu prüfen. Achten Sie insbesondere darauf, ob Sie die Drahtbrücken richtig gesetzt haben. Diese bestimmen, ob ein Wandler im Computer benutzt werden soll (C für Apple II, Commodore 64, IBM PC) oder ob der Wandler auf der Platine aktiviert wird (S für NDR-Klein und Schneider-CPC). Besitzer von Commodore-Computern müssen wir ein klein wenig enttäuschen: Der Analogeingang ist nicht so präzise, wie Sie vielleicht erwartet hätten. Allerdings sind die Positionieraufgaben in den folgenden Roboterprogrammen so geschrieben, daß eine kleine Ungenauigkeit noch keine Nachteile bringt.

Dr. Roland Schulé

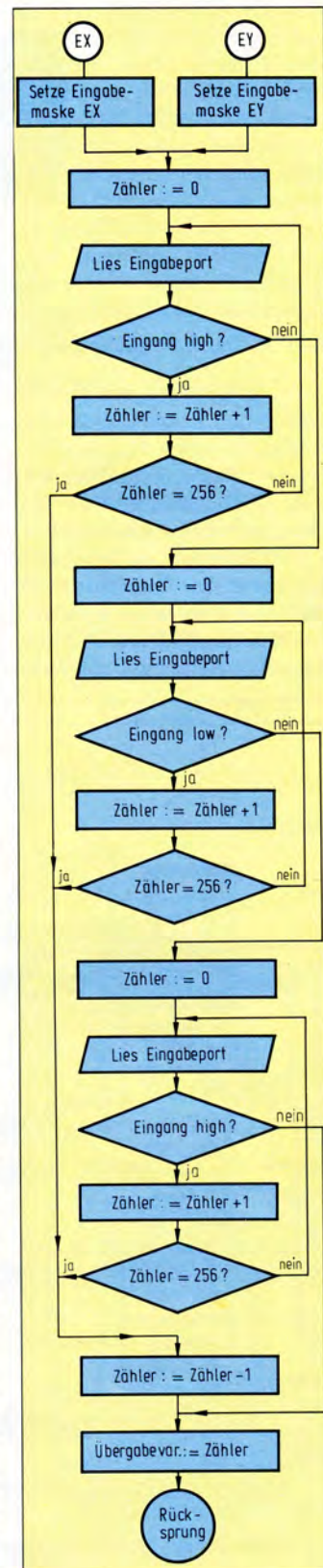


Bild 11: Maschinenprogramm (Z80-Code) zur Auswertung des Oszillatorsignals.


```

100 MEMORY &9FFF
110 REM Steuerung der Werkzeugmaschine
120 REM
130 REM (C) Dr.R.Schule, 1986
140 REM
150 REM Funktion: Die Werkzeugmaschine
160 REM wird durch einen wiederholenden
170 REM Bewegungsablauf mit den Phasen:
180 REM Anheben der Bohrmaschine
190 REM Drehen des Tisches
200 REM Absenken der Bohrmaschine
210 REM Bearbeitungsvorgang
220 REM gesteuert.
230 REM
240 REM Belegung der Ausgaenge:
250 REM Ausgang 0 - Anheben
260 REM Ausgang 1 - Absenken
270 REM Ausgang 2 - nicht benutzt
280 REM Ausgang 3 - nicht benutzt
290 REM Ausgang 4 - Drehtellermotor
300 REM Ausgang 5 - Bohrmaschine
310 REM Ausgang 6 - nicht benutzt
320 REM Ausgang 7 - Eingabesteuerung
330 REM
340 REM Belegung der Eingaeenge
350 REM Eingang 0 - Position oben
360 REM Eingang 1 - Position unten
370 REM Eingang 2 - Position Drehteller
380 REM Eingang 3 - nicht benutzt
390 :
400 REM Titelmeldung
410 INK 0,26 :REM Hintergrund weiss
420 INK 1,0 :REM Schriftfarbe schwarz
430 INK 2,6 : INK 3,13 :REM rot u. grau
440 MODE 1 :REM Schirm loeschen
450 LOCATE 1,3
460 PRINT " Werkzeugmaschine";
470 GOSUB 2150 :REM Drucke Grafiksymbole
480 DATA 8,3,3,133,1
490 PRINT " -----";
500 GOSUB 2150
510 DATA 7,3,3,150,133
520 GOSUB 2150
530 DATA 23,3,209,157,133
540 GOSUB 2150
550 DATA 24,2,147,133
560 GOSUB 2150
570 DATA 19,7,140,32,140,32,140,32,133
580 GOSUB 2150
590 DATA 16,13,143,143,2,231,1,143,143,143,143,
    143,3,32,133
600 GOSUB 2150
610 DATA 16,10,211,209,32,211,211,32,209,209,32,133
620 GOSUB 2150
630 DATA 15,13,208,208,208,208,208,208,208,208,
    208,208,208,208,1
640 :
1000 REM Hauptprogramm WERKZEUGMASCHINE
1010 GOSUB 1910 :REM Anfangswerte setzen
1020 CLS #1 :REM Bildfenster loeschen
1030 GOSUB 1110 :REM Anheben
1040 GOSUB 1310 :REM Drehen
1050 GOSUB 1510 :REM Absenken
1060 GOSUB 1710 :REM Bearbeiten
1070 GOTO 1020 :REM Ruecksprung
1080 :
1100 REM Unterprogramm ANHEBEN
1110 PRINT #1," Bohrkopf anheben"
1120 bit=0
1130 GOSUB 20100 :REM Motor aufwaerts
1140 bit=0
1150 GOSUB 30100 :REM Endposition?
1160 IF ein THEN GOTO 1150
1170 bit=0
1180 GOSUB 20200 :REM Motor aus
1190 RETURN
1200 :
1300 REM Unterprogramm DREHEN
1310 PRINT #1," Rundtisch drehen"
1320 bit=4
1330 GOSUB 20100 :REM Tischmotor an
1340 bit=2
1350 GOSUB 30100 :REM Taster frei?
1360 IF NOT ein THEN GOTO 1350
1370 GOSUB 30100 :REM Taster betaetigt?
1380 IF ein THEN GOTO 1370
1390 bit=4
1400 GOSUB 20200 :REM Tischmotor aus

```

```

1410 RETURN
1420 :
1500 REM Unterprogramm ABSENKEN
1510 PRINT #1," Bohrkopf absenken"
1520 bit=1
1530 GOSUB 20100 :REM Motor abwaerts
1540 bit=1
1550 GOSUB 30100 :REM Endposition?
1560 IF ein THEN GOTO 1550
1570 bit=1
1580 GOSUB 20200 :REM Motor aus
1590 RETURN
1600 :
1700 REM Unterprogramm BEARBEITEN
1710 PRINT #1," Werkstueck bearbeiten"
1720 bit=5
1730 GOSUB 20100 :REM Bohrmotor ein
1740 FOR z=0 TO zeit
1750 NEXT z
1760 bit=5
1770 GOSUB 20200 :REM Bohrmotor aus
1780 RETURN
1790 :
1900 REM Unterprogramm ANFANGSWERTE
1910 WINDOW #1,1,40,16,22
1920 PRINT #1," Bearbeitungszeit eingeben !"
1930 PRINT #1," 1 Sekunde ist ungefaehr 1000"
1940 INPUT #1,zeit
1950 GOSUB 10000 :REM Ein/Ausgabe vorbereiten
1960 PRINT #1," Versorgungsspannung anlegen!"
1970 PRINT #1," Werkzeugmaschine bereit? (j)"
1980 k$=INKEY$
1990 IF k$<>"j" THEN GOTO 1980
2000 bit=7
2010 GOSUB 20100 :REM Eingaeenge freigeben
2020 LOCATE 1,24
2030 PRINT " Programmstop mit 2xESC"
2040 RETURN
2050 :
2100 REM Unterprogramm GRAFIKSYMBOLE
2110 REM Parameter der DATA-Zeile:
2120 REM lz=Zahl fuehrender Leerstellen
2130 REM gz=Zahl der Grafiksymbole
2140 REM gz Grafiksymbole (CHR$-Code) bzw. PEN
2150 READ lz : PRINT SPC(lz);
2160 READ gz
2170 FOR i=1 TO gz
2180 READ gc
2190 IF gc<16 THEN PEN gc ELSE PRINT CHR$(gc);
2200 NEXT i
2210 PRINT
2220 RETURN
2230 :
10000 REM Unterprogramm INIT

```


Einführung in die Robotik:

Der Roboter lernt

Mit der Theorie der Achsen und Freiheitsgrade sind wir nun vertraut. Jetzt soll unser Modellroboter zeigen, was er kann.

Über die Ausgabe- und Motorplatine wird der Minimotor auf der Grundplatte angesteuert (Bild 1). Über ein Untersetzungsgetriebe wird der Drehkranz angetrieben, auf dem der gesamte Aufbau sitzt. Im Mittelloch des Drehkranzes ist ein Potentiometer befestigt, und zwar so, daß der Körper des Potis feststeht, die Achse aber mit den senkrechten Säulen des Roboters verbunden ist. Da das Poti mit dem Analogeingang EY verbunden ist, wird der Computer anhand des sich ändernden Widerstandswertes während einer Drehbewegung immer über die Position informiert. Dieser Regelkreis muß nun nur noch über ein Programm in die gewünschten Bahnen gelenkt werden.

Im einfachsten Fall gehen wir von einer Dreibegeentscheidung aus, wie sie in dem Programmablaufplan von Bild 2 gezeigt ist. Der Roboter soll eine bestimmte Lage einnehmen, die durch den dazugehörigen Analogwert beschrieben ist. Dieser Wert ist in der Variablen SOLL abgelegt. Im ersten Schritt erfolgt eine Messung der Position, die das Ergebnis IST liefert. Nun können drei Fälle unterschieden werden:

SOLL > IST : Motor im Linkslauf drehen lassen.

SOLL = IST : Motor ausschalten.

SOLL < IST : Motor im Rechtslauf drehen lassen.

Was bedeutet nun Linkslauf und Rechtslauf? Die Bezeichnungen sind eigentlich willkürlich gewählt. Wir wollen festlegen, daß bei der Draufsicht von oben auf den Roboter sich der Aufbau im

Uhrzeigersinn bewegt, wenn Bit 0 der Ausgabeplatine gesetzt ist und Bit 1 nicht. Dies bezeichnen wir als Rechtslauf. Sollte der Motor sich in

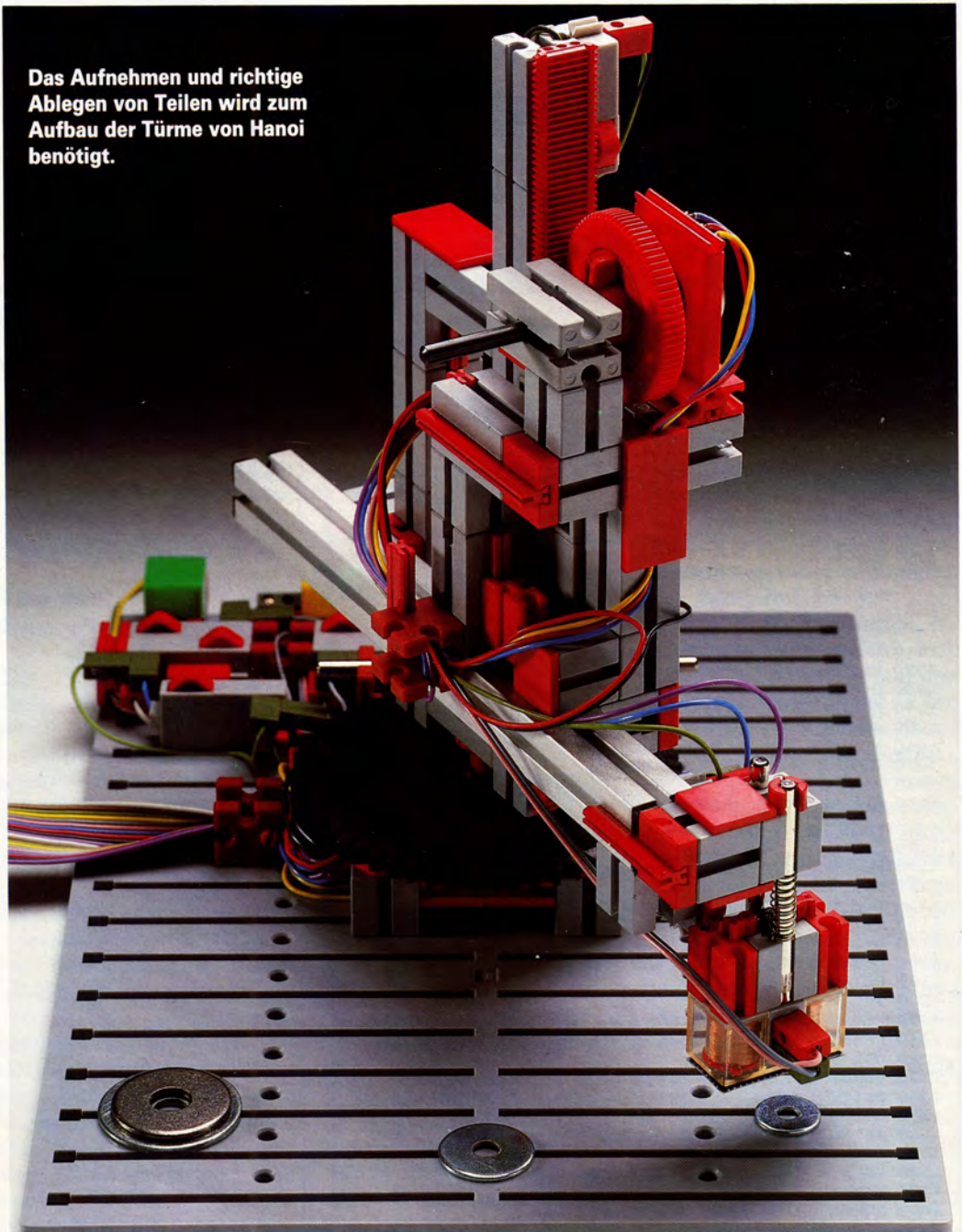
die andere Richtung drehen, vertauschen Sie einfach die beiden Anschlüsse an dem Motor. Nun muß aber auch das Potentiometer in der richtigen Polarität angeschlossen sein. Bewegt sich der Roboter im Uhrzeigersinn, so müssen die Analogwerte sinken. Sie können

dies leicht mit dem Diagnoseprogramm kontrollieren.

Sollte die beschriebene Zuordnung der Analogwerte nicht zutreffen, belassen Sie den Mittelabgriff des Potentiometers und schließen die zweite Leitung an den anderen Endabgriff an.

Nur wenn die Zuordnung korrekt getroffen ist, wird sich der Roboter auf die Position SOLL zubewegen. Im anderen Fall wird er gerade von der Position SOLL weglaufen und unweigerlich bis zum Endanschlag des Poten-

Das Aufnehmen und richtige Ablegen von Teilen wird zum Aufbau der Türme von Hanoi benötigt.



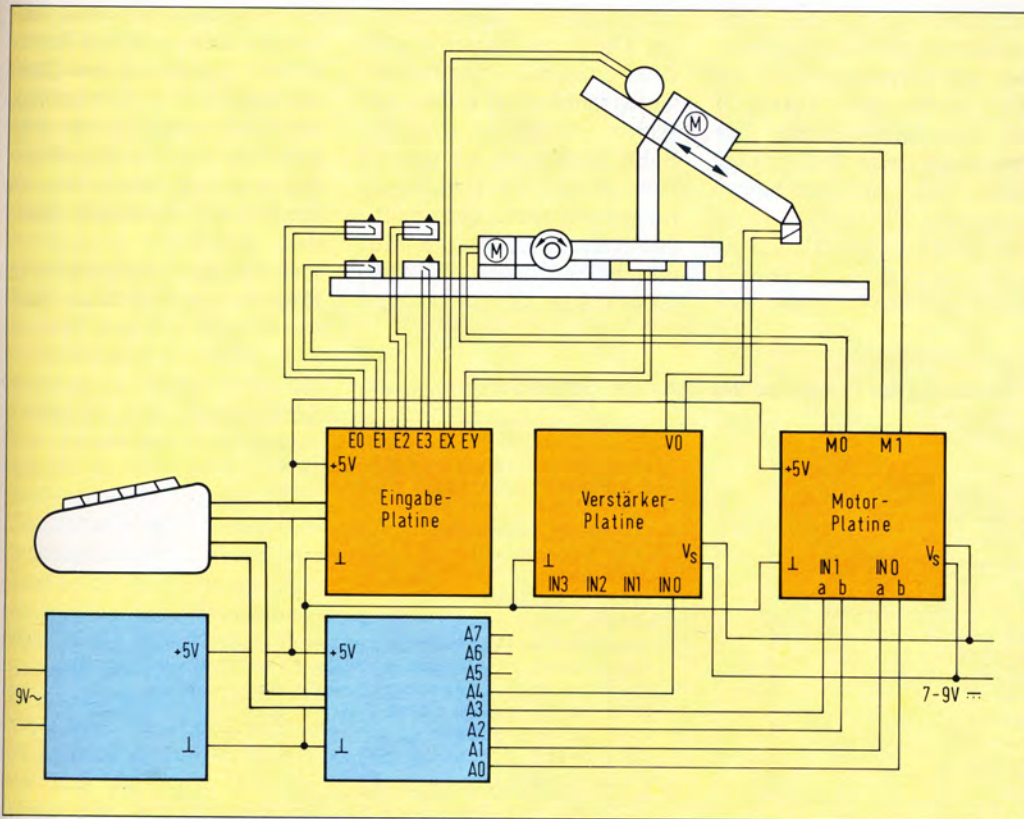
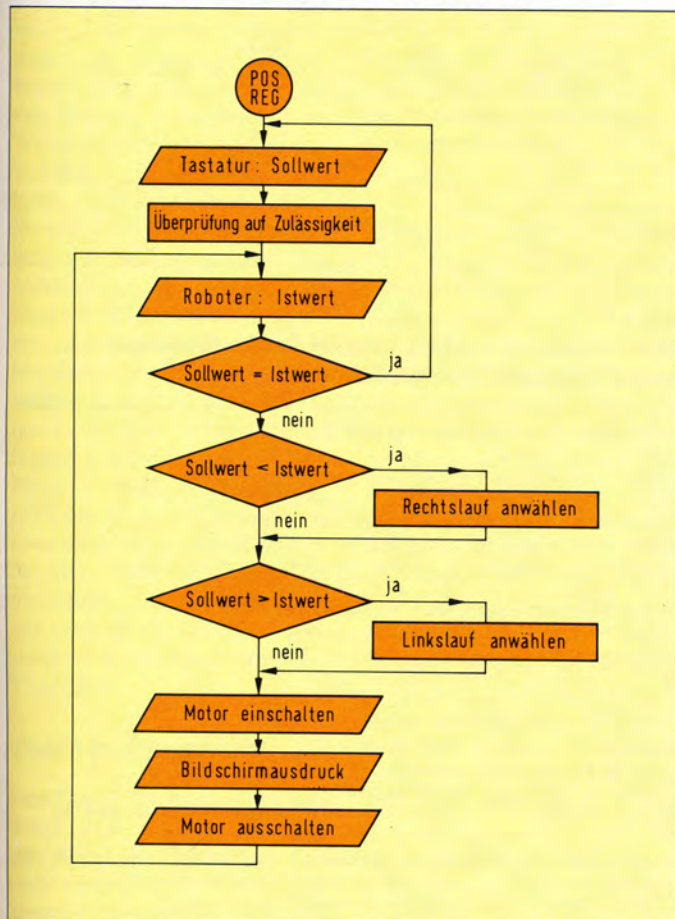


Bild 1: Blockschaltbild zum fischertechnik-Roboter, hier ein Modell mit schräger Ebene.

Bild 2: Programmablaufplan zur Positionsregelung.



tiometers voranschreiten. Gegebenenfalls kann sogar das Potentiometer beschädigt werden.

Wir sollten nun das Programm ausprobieren. Die Position SOLL wird mit einem INPUT-Befehl vorgegeben. Zu dieser Eingabe verzweigt das Programm auch immer im mittleren der drei obengenannten Fälle, so daß eine Position nach der anderen eingegeben werden kann. Nach der Eingabe erfolgt eine Kontrolle auf die Zulässigkeit der Eingabewerte. Die Zahlenkonstanten der Abfrage werden daher gleich zu Programm-anfang eingegeben. Sie sollten die von Ihnen bereits ermittelten kleinsten und größten Analogwerte verwenden.

Experimentieren Sie eine Weile mit diesem Aufbau. Beobachten Sie, wie schnell das Programm den Roboter in die Sollposition einschwenkt. Kommt es zu einem Überschwingen oder bleibt der Roboter sofort an der Zielposition stehen?

Ändern Sie nun das Pro-

gramm so ab, daß Sie die zweite Achse des Roboters verwenden. Zunächst legen wir wieder den Drehsinn von Motor und Potentiometer fest. Bei Rechtslauf (Bit 2 high, Bit 3 low) wird der Arm nach oben transportiert. Wie zuvor soll sich dann auch die mittransportierte Potentiometerachse im Uhrzeigersinn drehen und die Analogwerte absinken. Die Anpassung an diese Forderung kann wie zuvor durch Tauschen der Anschlüsse erfolgen. In dem Programm müssen wir lediglich noch die Variable BIT an die andere Motor-nummer anpassen und den Analogkanal EX abfragen. **Bild 3** enthält die Zeilen, die im Programm geändert werden müssen.

Zu unserer Überraschung wird das Programm, obwohl ansonsten identisch, zu einem ganz anderen Regelverhalten führen. Der Grund ist in der geänderten Zahnradübersetzung zu suchen. Wenn es vorhin noch zu gar keinem oder nur geringfügigem Überschwingen kam, tritt dieser Effekt nun wesentlich deutlicher auf. Bei der Programmierung des Roboters werden wir diesen Effekt im Auge behalten müssen.

Roboter-Ouvertüre

Zunächst wollen wir eine ganz einfache Aufgabe für den Roboter studieren. Ein Teil soll am Platz A aufgenommen und nach Platz D transportiert werden, wie es schematisch in **Bild 5** aufgezeigt ist. Am Platz D wird das Teil abgesetzt. Der Roboter kehrt anschließend an Platz A zurück, um das nächste Teil aufzunehmen. Um wie im Foto „Türme von Hanoi“ spielen zu können, müssen drei Positionen angefahren werden.

Bei der Programmierung des Roboters ist zu bedenken, daß die Teile normalerweise immer erst aus einem Kollisionsbereich herausgezogen werden müssen, bevor sie

transportiert werden können. Dies ist in Bild 5 durch die Bahnen A-B und C-D dargestellt. Abhängig davon, ob das Teil einer tiefen Kiste oder von ebener Fläche entnommen bzw. abgelegt wird, werden diese Strecken länger oder kürzer sein. Der eigentliche Transport ist auf die Strecke B-C

zurückverlagert. Bei dem fischertechnik-Roboter werden die Strecken A-B und C-D durch den Vorschub des Armes abgefahren. Der Transport längs B-C erfolgt durch die Schwenkbewegung.

Die einzelnen Positionen A, B, C und D sind jeweils durch die Stellung der Ro-

boterachsen gekennzeichnet. Die Drehung wird in der Zeichnung mit y bezeichnet, die Armhöhe mit x. Die einzelnen Sollwerte für die Bahn lauten x1, x2 und x3 bzw. y1 und y2. Das Steuerungsprogramm, das wir für den Roboter benötigen, muß die beiden Positionsregelprogramme für die x- und

die y-Achse beinhalten. Außerdem ist eine kleine Routine zur Steuerung des Greifermagneten vorzusehen. Die Sollwerte, die wir bislang von der Tastatur eingegeben haben, müssen nun im Programm verankert werden.

Der Programmablaufplan in Bild 6 gewährleistet dies.

Bild 3: Positionsregelprogramm zur Steuerung der Drehachse des Roboters (Schneider).

```

110 REM Positionsregelung 1
120 REM
150 REM Funktion: Dreht den Aufbau des
160 REM Roboters so, dass der vorgege-
170 REM bene Analogwert Y eingestellt
180 REM wird. Die Eingabe wird auf Zu-
190 REM laessigkeit geprueft.
210 REM
220 REM Belegung der Ausgaenge:
230 REM Ausgang 0 - Drehung (rechts)
240 REM Ausgang 1 - Drehung (links)
250 REM Ausgang 2 - nicht benutzt
260 REM Ausgang 3 - nicht benutzt
270 REM Ausgang 4 - nicht benutzt
280 REM Ausgang 5 - nicht benutzt
290 REM Ausgang 6 - nicht benutzt
300 REM Ausgang 7 - nicht benutzt
310 REM
320 REM Belegung der Eingaeenge:
330 REM Eingang 0-3 - nicht benutzt
340 REM Eingang X - nicht benutzt
350 REM Eingang Y - Drehwinkel
370 REM Titelmeldung
380 INK 0,26 :REM Hintergrund weiss
390 MODE 1 :REM Bild loeschen
400 INK 1,0 :REM Schrift schwarz
410 INK 2,6 : INK 3,13 :REM rot u. grau
420 LOCATE 1,3
430 PRINT " Positionsregelung";
440 GOSUB 2150 :REM Drucke Grafiksymbole
450 DATA 3,5,209,205,2,231,1
460 PRINT " -----";
470 GOSUB 2150
480 DATA 4,5,205,3,214,143,215
490 GOSUB 2150
500 DATA 23,4,143,143,143,215
510 GOSUB 2150
520 DATA 23,6,143,1,205,3,213,212
530 GOSUB 2150
540 DATA 23,6,143,32,1,205,205,3
550 GOSUB 2150
560 DATA 23,7,143,32,32,1,205,205,3
570 GOSUB 2150
580 DATA 17,17,143,143,143,2,231,1,143,143,143,143,
143,209,3,143,1,211,3
590 GOSUB 2150
600 DATA 17,9,233,32,233,32,233,32,32,32,233
610 GOSUB 2150
620 DATA 15,16,1,208,208,208,208,208,208,208,208,
208,208,208,208,208,208
630 PRINT
980 :
990 REM Hauptprogramm Positionsregelung
1000 GOSUB 1950:REM Anfangswerte setzen
1010 LOCATE 1,16
1020 INPUT " Sollwert der Drehung";soll
1030 IF soll>maximum THEN CLS : END
1040 IF soll<minimum THEN CLS : END
1050 GOSUB 30300 :REM lies Y-Wert
1060 IF soll=y THEN GOTO 1170
1070 IF soll<y THEN bit=0
1080 IF soll>y THEN bit=1
1090 GOSUB 20100 :REM Motor ein
1110 LOCATE 1,18
1120 PRINT " Istwert der Drehung ";USING"###";y
1130 GOSUB 20200 :REM Motor aus
1140 GOTO 1050 :REM Ruecksprung
1150 :
1160 REM neuer Sollwert
1170 LOCATE 1,16
1180 PRINT "
1190 GOTO 1010
1890 :
1900 REM Anfangswerte setzen
1950 INPUT " groesster Y-Wert";maximum
1960 INPUT " kleinster Y-Wert";minimum
1970 GOSUB 10000 :REM Ein/Ausgabe vorbereiten
1980 LOCATE 1,16
1990 PRINT " Roboter bereit? (j)"
2000 k$=INKEY$
2010 IF k$<>"j" THEN GOTO 2000
2020 LOCATE 1,22
2030 PRINT " Programmstop mit 2xESC"
2040 PRINT " oder unzuLaessigem Sollwert."
2050 RETURN
2090 :
2100 REM Unterprogramm GRAFIKSymBOLE
2110 REM Parameter der DATA-Zeile:
2120 REM lz=Zahl fuehrender Leerstellen
2130 REM gz=Zahl der Grafiksymbole
2140 REM gz Grafiksymbole (CHR$(Code) bzw. PEN
2150 READ lz : PRINT SPC(lz);
2160 READ gz
2170 FOR i=1 TO gz
2180 READ gc
2190 IF gc<16 THEN PEN gc ELSE PRINT CHR$(gc);
2200 NEXT i
2210 PRINT
2220 RETURN

```

Bild 4:
Durch Eingabe dieser Zeilen steuert das Positionsregelprogramm die Armhöhe des Roboters.

```

110 REM Positionsregelung 2
220 REM Belegung der Ausgaenge:
230 REM Ausgang 0 - nicht benutzt
240 REM Ausgang 1 - nicht benutzt
250 REM Ausgang 2 - Arm hoch (rechts)
260 REM Ausgang 3 - Arm runter (links)
320 REM Belegung der Eingaeenge:
340 REM Eingang X - Armhoehe
350 REM Eingang Y - nicht benutzt
1020 INPUT " Sollwert der Armhoehe";soll
1050 GOSUB 30200 :REM lies X-Wert
1060 IF soll=x THEN GOTO 1170
1070 IF soll<x THEN bit=2
1080 IF soll>x THEN bit=3
1120 PRINT " Istwert der Armhoehe ";USING"###";x
1900 REM Anfangswerte setzen
1950 INPUT " groesster X-Wert";maximum
1960 INPUT " kleinster X-Wert";minimum

```


Dabei werden die Bahnelemente in Form einer Tabelle beschrieben. Jedem Bahnpunkt ist eine Zeile der Tabelle zugeordnet. Der erste Eintrag in die Zeile benennt die Roboterachse, die eingestellt werden soll (0 = Drehung, 1 = Armlänge, 2 = Greifermagnet). Der Wert bestimmt das Unterprogramm, das die Einstellung vornehmen soll. Der zweite Eintrag in der Zeile ist der Sollwert der Einstellung. Die Zeilen der Bewegungstabelle werden in einer Wiederholungschleife abgearbeitet. Die Tabelle ist in dem Roboterprogramm in Form von DATA-Zeilen abgelegt. Diese werden zum Programmstart in die eigentlichen Felder der Tabelle ACHS und SOLL eingelesen. Die DATA-Zeilen im abgedruckten Programm enthalten Werte, die sich in einem Experiment des Autors ergaben. Wenn die Potentiometer Ihres Roboters anders eingestellt sind, wird diese Bewegung nicht unbedingt sinnvoll sein. Es gilt daher, die Bewegungstabelle schleunigst der eigenen Aufgabenstellung und dem eigenen Roboter anzupassen. Wie erstellen wir jedoch die Tabelle? Sie könnten die oben beschriebenen Programme zur Sollwerteinstellung der beiden Achsen benutzen, um die erforderlichen Positionen anzusteuern. Anschließend notieren Sie sich die Sollwerte (am besten gleich in der richtigen Tabellenform) und übertragen Sie sie in die DATA-Zeilen des Roboterprogramms. Wir wollen uns nun aber eine bequemere Methode der Tabellenerzeugung überlegen. Dazu erweitern wir das einfache Roboterprogramm zu dem Teach-In-Programm.

Teach-In-Programm

Das Teach-In-Programm benutzt eine Handsteuerung des Roboters, die schon für sich alleine recht viel Spaß bereitet. Hierfür werden vier

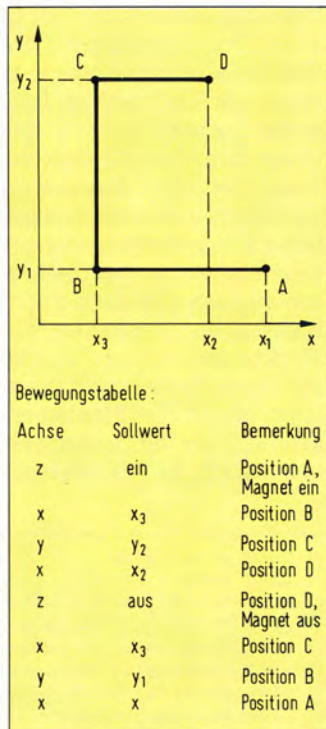
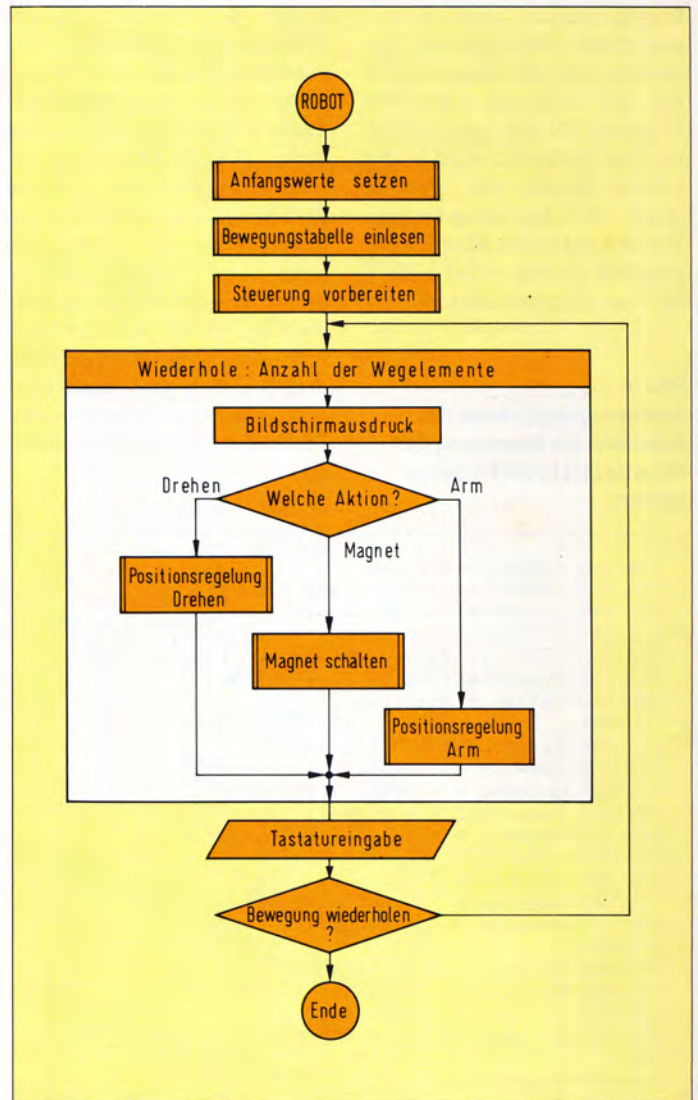


Bild 5:
Die Aufgabenstellung des Roboters. Ein Teil soll bei A abgeholt und bei D abgelegt werden. Die Bahn führt zur Vermeidung von Kollisionen über B und C.

Bild 6: ►
Der Programmablaufplan zeigt, wie übersichtlich sich die Steuerung eines Roboters programmieren lässt.

Mikrotaster an die Eingänge der Eingabeplatine angeschlossen. Die Funktion der Taster:
Mit jeder Betätigung des Tasters an Eingang 0 wird die nächste Roboterfunktion angewählt. Welche Funktion gerade aktiv ist, kann dem Bildschirm entnommen werden. Zusätzlich wird, außer beim ersten Mal, die jeweils vorangegangene Bewegung in die Bewegungstabelle eingetragen.
Ein Low-Signal an Eingang 1 aktiviert den Roboter. Je nach gewählter Roboterfunktion dreht sich der Roboter im Uhrzeigersinn, hebt den Arm oder schaltet den Greifermagneten ein. Die Bewegung wird solange ausgeführt, wie der Taster gedrückt ist.



Ein Low-Signal an Eingang 2 aktiviert den Roboter in der jeweils entgegengesetzten Funktion. Er dreht sich gegen den Uhrzeigersinn, senkt den Arm bzw. schaltet den Greifermagneten ab. Mit Betätigung des Tasters am Eingang 3 wird die Handsteuerung beendet. Das Programm geht zum Wiedergabebetrieb über. Mit den erstgenannten drei Tastern steuern wir den Roboter, so daß er die gewünschte Bewegung ausführt. Dabei können wir uns zunutze machen, daß eine Abspeicherung immer erst bei Anwahl der nächsten Achse mit Taster 0 erfolgt. Dies erlaubt vorher beliebig viele Korrekturen mit den Tastern 1 und 2. Anders als die Roboterach-

sen wird der Greifermagnet behandelt. Hier dienen die Eingänge 1 und 2 als Steuerungseingänge eines Software-Flipflops. Jede Betätigung von Eingang 1 schaltet den Magneten ein und läßt ihn auch eingeschaltet, selbst wenn die Taste inzwischen losgelassen wird. Entsprechend schaltet die Betätigung von Eingang 2 den Magneten aus.
Den meisten Aufwand kostet die Programmierung des Wahlschalters an Eingang 0. Wenn er gedrückt wird, zeichnet das Programm zunächst einmal die bislang erreichte Position auf. Hierzu wird in die erste Spalte der Tabelle (Feld ACHS) die Roboterfunktion – 0 für die Drehung, 1 für die Armhöhe, 2 für den Magneten – einge-

tragen. Danach wird der zugeordnete Analogkanal gemessen, also der Eingang EY für die Drehung oder der Eingang EX für die Armlänge. Der Meßwert wird in die zweite Spalte der Tabelle (Feld SOLL) eingetragen. Wenn zuletzt der Magnet geschaltet wurde, wird eine 1 für den eingeschalteten, eine

0 für den ausgeschalteten Magneten eingetragen. Abschließend wird der Tabellenzeiger um eins weitergezählt, d. h. er zeigt eine Zeile tiefer. Wurde jedoch die zuletzt angewählte Achse überhaupt nicht aktiviert, so unterbleibt der oben beschriebene Abspeichervorgang. Diese Abfrage ist notwendig, denn bei der Auswahl der Roboterachse werden Sie sich gegebenenfalls erst bis zur gewünschten Funktion durchtasten müssen.

Eine weitere Besonderheit in der Programmierung des Wahlschalters an Eingang 0: Wenn ein Tastendruck festgestellt wurde, wartet das Programm darauf, daß die Taste wieder losgelassen wird, bevor es weitermacht. Diese Programmierweise zur Erkennung von Impulsflanken kennen Sie doch noch? Richtig, der Drehteller der Werkzeugmaschine wurde auf diese Weise positioniert. Hier dient die Flankenerkennung der Vermeidung einer mehrfachen Abspeiche-

rung einer Bewegung bei zu langem Tastendruck.

Weiter wird noch der Eingang 4 geprüft, der, wenn aktiviert, die Wiedergabe der gelernten Bewegung durch den Roboter auslöst. Wir gehen also in den zuvor schon getesteten Programmteil über.

Mit all diesen Merkmalen ist das Teach-In-Programm eines der umfangreichsten, aber auch leistungsfähigsten Programme der „Einführung in die Robotik“ geworden. Dabei ist es immer noch

Bild 7:
Steuerungsprogramm des Roboters. Die Bewegung des Roboters ist in DATA-Zeilen codiert.

```

110 REM Robotersteuerung 1
150 REM Funktion: Steuert den Roboter
160 REM in seinen beiden Bewegungsach-
170 REM sen sowie der Greiferfunktion
180 REM nach der Bewegungstabelle. Die
190 REM Bewegungstabelle wird aus DATA-
200 REM Zeilen eingelesen.
210 REM
220 REM Belegung der Ausgänge:
230 REM Ausgang 0 - Drehung (rechts)
240 REM Ausgang 1 - Drehung (links)
250 REM Ausgang 2 - Armlänge (hoch)
260 REM Ausgang 3 - Armlänge (runter)
270 REM Ausgang 4 - Greifermagnet
280 REM Ausgang 5 - nicht benutzt
290 REM Ausgang 6 - nicht benutzt
300 REM Ausgang 7 - Eingabesteuerung
310 REM
320 REM Belegung der Eingänge:
330 REM Eingang 0-3 - nicht benutzt
340 REM Eingang X - Armposition
350 REM Eingang Y - Drehwinkel
360 :
370 REM Titelmeldung
380 INK 0,26 :REM Hintergrund weiss
390 MODE 1 :REM Bild loeschen
400 INK 1,0 :REM Schrift schwarz
410 INK 2,6 : INK 3,13 :REM rot u. grau
420 PRINT
430 PRINT " Robotersteuerung 1";
440 GOSUB 3250 :REM Drucke Grafiksymbole
450 DATA 2,5,209,205,2,231,1
460 PRINT " -----";
470 GOSUB 3250
480 DATA 3,5,205,3,214,143,215
490 GOSUB 3250
500 DATA 23,4,143,143,143,215
510 GOSUB 3250
520 DATA 23,6,143,1,205,3,213,212
530 GOSUB 3250
540 DATA 23,6,143,32,1,205,205,3
550 GOSUB 3250
560 DATA 23,7,143,32,32,1,205,205,3
570 GOSUB 3250
580 DATA 17,17,143,143,143,2,231,1,143,143,143,143,
143,209,3,143,1,211,3
590 GOSUB 3250
600 DATA 17,9,233,32,233,32,233,32,32,233
610 GOSUB 3250
620 DATA 15,16,1,208,208,208,208,208,208,208,208,
208,208,208,208,208,208,208
630 :
990 REM Hauptprogramm Robotersteuerung 1
1000 GOSUB 1920 :REM Anfangswerte setzen
1020 GOSUB 2210 :REM Bewegungsdaten einlesen
1030 GOSUB 3010 :REM Steuerung vorbereiten
1040 REM Roboterbewegung ausfuehren
1050 FOR w=0 TO weg
1060 ax=achs(w)
1070 s=soll(w)
1080 CLS #1 :REM Fenster loeschen
1090 PRINT #1," Wegelement Nr. ";USING"###";w
1100 PRINT #1," Roboterachse ";ax
1110 PRINT #1," Sollwert ";USING"###";s
1120 ON ax+1 GOSUB 1310,1510,1710
1130 NEXT w
1140 REM Ende der Bewegung
1150 LOCATE #2,1,3
1160 PRINT #2," wiederholen? (j/n) "
1170 k$=INKEY$
1180 IF k$="n" THEN END
1190 IF k$(">")="j" THEN GOTO 1170
1200 LOCATE #2,1,3
1210 PRINT #2," Programmstop mit 2xESC"
1220 GOTO 1050
1290 :
1300 REM Drehbewegung
1310 GOSUB 30300 :REM Y-Wert
1320 IF s=y THEN RETURN
1330 IF s<y THEN bit=0 :REM Rechtslauf
1340 IF s>y THEN bit=1 :REM Linkslauf
1350 GOSUB 20100 :REM Motor ein
1360 GOSUB 30300 :REM Y-Wert
1370 IF (bit=0) AND (s>y) THEN GOSUB 20200 : RETURN
1380 IF (bit=1) AND (s<y) THEN GOSUB 20200 : RETURN
1390 LOCATE #1,1,4
1400 PRINT #1," Istwert ";USING"###";y
1410 GOTO 1360
1490 :
1500 REM Armlänge
1510 GOSUB 30200 :REM X-Wert
1520 IF s=x THEN RETURN
1530 IF s<x THEN bit=2 :REM Rechtslauf
1540 IF s>x THEN bit=3 :REM Linkslauf
1550 GOSUB 20100 :REM Motor ein
1560 GOSUB 30200 :REM X-Wert
1570 IF (bit=2) AND (s>x) THEN GOSUB 20200 : RETURN
1580 IF (bit=3) AND (s<x) THEN GOSUB 20200 : RETURN
1590 LOCATE #1,1,4
1600 PRINT #1," Istwert ";USING"###";x
1610 GOTO 1560
1690 :
1700 REM Magnet schalten
1710 bit=4
1720 ON s+1 GOSUB 20200,20100
1730 LOCATE #1,1,3
1740 PRINT #1," Magnet ";
1750 IF s=0 THEN PRINT #1,"aus" ELSE PRINT #1,"ein"
1760 REM Verzögerungsschleife
1770 FOR z=0 TO 1000
1780 NEXT z
1790 RETURN
1890 :
1900 REM Anfangswerte setzen
1910 REM Dimensionierung der Grenzwerte
1920 DIM maximum(2),minimum(2)
1930 REM Dimensionierung der Wegelemente

```


ausbaufähig. Die Teach-In-Phase und die Ausführ-Phase müssen z. B. nicht strikt aufeinander folgen. Vielmehr können sie Bestandteile eines Menüs sein, das noch andere Wahlpunkte zur Pflege von Bewegungstabellen enthält: Ausdrucken, Speichern und Laden, Editieren, Ausführung von Teilbewegungen usw. Ihrer Phantasie sind in dieser Beziehung keine Grenzen gesetzt. Wer seinen Computer genau kennt, wird die Steuerung des Roboters sicherlich

noch mit Bildschirmgrafiken unterstützen können. Dabei handelt es sich keineswegs um eine Spielerei. Die Simulation von Roboterbewegungen am Bildschirm gehört zu den fortschrittlichsten Methoden der Einsatzplanung von Robotern.

Dr. Roland Schulé

```
1940 DIM achs(500),soll(500)
1950 REM Bildschirmfenster
1960 WINDOW #1,1,40,17,20
1970 WINDOW #2,1,40,22,24
1980 REM Grenzwerte abfragen
1990 INPUT " grösster X-Wert (Arm) ";maximum(1)
2000 INPUT " kleinster X-Wert ";minimum(1)
2010 INPUT " grösster Y-Wert (Dreh.)";maximum(0)
2020 INPUT " kleinster Y-Wert ";minimum(0)
2030 PRINT " Magnet 0 - aus 1 - ein"
2040 maximum(2)=1
2050 minimum(2)=0
2060 RETURN
2190 :
2200 REM Bewegungsdaten einlesen
2210 PRINT #1," Bewegungsdaten lesen!"
2220 READ weg :REM Zahl der Wegelemente-1
2230 DATA 7
2240 FOR w=0 TO weg
2250 READ achs(w),soll(w)
2260 IF soll(w)<minimum(achs(w)) THEN END
2270 IF soll(w)>maximum(achs(w)) THEN END
2280 NEXT w
2290 RETURN
2300 :
2310 REM Bewegungsdaten (Beispiel)
2320 DATA 1,150
2330 DATA 0,130
2340 DATA 1,80
2350 DATA 2,1
2360 DATA 1,150
2370 DATA 0,90
2380 DATA 1,80
2390 DATA 2,0
2990 :
3000 REM Ein/Ausgabe vorbereiten
3010 GOSUB 10000
3020 CLS #2
3030 LOCATE #2,1,3
3040 PRINT #2," Roboter bereit? (j) "
3050 k$=INKEY$
3060 IF k$<>"j" THEN GOTO 3050
3070 LOCATE #2,1,3
3080 PRINT #2," Programmstop mit 2xESC"
3090 RETURN
3190 :
3200 REM Unterprogramm GRAFIKSymBOLE
3210 REM Parameter der DATA-Zeile:
3220 REM lz=Zahl fuhrender Leerstellen
3230 REM gz=Zahl der Grafiksymbbole
3240 REM gz Grafiksymbbole (CHR$(Code) bzw. PEN
3250 READ lz : PRINT SPC(lz);
3260 READ gz
3270 FOR i=1 TO gz
3280 READ gc
3290 IF gc<16 THEN PEN gc ELSE PRINT CHR$(gc);
3300 NEXT i
3310 PRINT
3320 RETURN
```

```
110 REM Robotersteuerung 2
320 REM Belegung der Eingänge:
330 REM Eingang 0 - Bewegungssache
331 REM Eingang 1 - Rechtslauf
332 REM Eingang 2 - Linkslauf
333 REM Eingang 3 - Ende der Eingabe
430 PRINT " Robotersteuerung 2";

1010 GOSUB 3010 :REM Steuerung vorbereiten

2200 REM Aufbau der Bewegungstabelle
2210 CLS #1
2220 PRINT #1," Roboterachsen: 0 - Drehung"
2230 PRINT #1," 1 - Armlänge 2 - Magnet"
2240 weg=0 : ax=0 : m=0 : beweg=0
2250 LOCATE #2,1,1
2260 PRINT #2," Wegelement Nr. ";weg
2270 PRINT #2," Achse (0/1/2) ";ax
2290 bit=0 :REM Bewegungssache ?
2300 GOSUB 30100
2310 IF ein THEN GOTO 2470
2320 GOSUB 30100 :REM Ende Tastendruck?
2330 IF NOT ein THEN GOTO 2320
2340 IF beweg=0 THEN GOTO 2420
2360 REM abspeichern
2370 achs(weg)=ax
2380 soll(weg)=s
2390 weg=weg+1:beweg=0
2410 REM Bewegungssache weiterschalten
2420 ax=ax+1
2430 IF ax=3 THEN ax=0
2440 GOSUB 2850 :REM Statusausdruck
2450 GOTO 2250
2470 bit=1 :REM Rechtsdrehung?
2480 GOSUB 30100
2490 IF ein THEN GOTO 2570
2500 beweg=1 :REM Flagge fuer Aenderung
2510 IF ax=2 THEN m=1
2520 GOSUB 2850 :REM Statusausdruck
2530 IF s<minimum(ax) THEN GOTO 2570
2540 bit=2*ax :REM Motoranwahl
2550 GOSUB 20100 :REM Einschalten
2560 GOTO 2470 :REM Ruecksprung
2570 bit=2*ax
2580 REM Motor ausschalten
2590 IF ax<>2 THEN GOSUB 20200
2610 bit=2 :REM Linksdrehung?
2620 GOSUB 30100
2630 IF ein THEN GOTO 2740
2640 beweg=1 :REM Flagge fuer Aenderung
2650 IF ax=2 THEN m=0
2660 GOSUB 2850 :REM Statusausdruck
2670 IF ax<>2 THEN GOTO 2700
2680 bit=4 :REM Sonderfall Magnet
2690 GOTO 2750
2700 IF s>maximum(ax) THEN GOTO 2740
2710 bit=2*ax+1 :REM Motoranwahl
2720 GOSUB 20100 :REM einschalten
2730 GOTO 2610
2740 bit=2*ax+1
2750 GOSUB 20200 :REM ausschalten
2770 bit=3 :REM Ende der Aufzeichnung?
2780 GOSUB 30100
2790 IF ein THEN GOTO 2250
2800 achs(weg)=ax :REM letztes Wegelement
2810 soll(weg)=s
2820 RETURN
2830 :
2840 REM Statusausdruck
2850 IF ax=0 THEN GOSUB 30300:s=y
2860 IF ax=1 THEN GOSUB 30200:s=x
2870 IF ax=2 THEN s=m
2880 LOCATE 1,24
2890 s$=RIGHT$(" "+STR$(s),3)
2900 IF (ax=2) AND (s=0) THEN s$="aus"
2910 IF (ax=2) AND (s=1) THEN s$="ein"
2920 PRINT " Sollwert ";s$
2930 RETURN
```

Bild 8: Durch Austausch der Routine zum Einlesen der Bewegungsdaten gegen die Teach-In-Routine wird die Bewegungsprogrammierung des Roboters komfortabler.


```

110 REM POSITIONSREGELUNG 1
150 REM FUNKTION: DREHT DEN AUFBAU DES
160 REM ROBOTERS SO, DASS DER VORGEGE-
170 REM BENE ANALOGWERT Y EINGESTELLT
180 REM WIRD. DIE EINGABE WIRD AUF ZU-
190 REM LAESSIGKEIT GEPRUEFT.
220 REM BELEGUNG DER AUSGAENGE:
230 REM AUSGANG 0 - DREHUNG (RECHTS)
240 REM AUSGANG 1 - DREHUNG (LINKS)
250 REM AUSGANG 2 - NICHT BENUTZT
260 REM AUSGANG 3 - NICHT BENUTZT
270 REM AUSGANG 4 - NICHT BENUTZT
280 REM AUSGANG 5 - NICHT BENUTZT
290 REM AUSGANG 6 - NICHT BENUTZT
300 REM AUSGANG 7 - NICHT BENUTZT
320 REM BELEGUNG DER EINGAENGE:
330 REM EINGANG 0-3 - NICHT BENUTZT
340 REM EINGANG X - NICHT BENUTZT
350 REM EINGANG Y - DREHWINKEL
380 POKE 53281,1 :REM HINTERGRUND WEISS
390 PRINT CHR$(147); :REM BILD LOESCHEN
400 PRINT CHR$(31); :REM SCHRIFT BLAU
410 PRINT : PRINT
420 PRINT" POSITIONSREGELUNG";
430 GOSUB 2150 :REM DRUCKE GRAFIKSYMBOL
440 DATA 3,6,144,167,109,150,113,31
450 PRINT" -----";
460 GOSUB 2150
470 DATA 4,8,144,109,152,18,169,32,127,146
480 GOSUB 2150
490 DATA 23,6,18,32,32,32,127,146
500 GOSUB 2150
510 DATA 23,8,18,32,146,144,109,152,127,169
520 GOSUB 2150
530 DATA 23,8,18,32,146,32,144,109,109,152
540 GOSUB 2150
550 DATA 23,9,18,32,146,32,32,144,109,109,152
560 GOSUB 2150
570 DATA 17,23,18,32,32,32,146,150,113,144,18
580 DATA 96,96,96,96,96,146,167,152,18,32,146,144,165,152
590 GOSUB 2150
600 DATA 17,17,18,163,146,32,18,163,146,32,18,163,146,
32,32,32,18,163,146
610 GOSUB 2150
620 DATA 15,16,144,163,163,163,163,163,163,163,163,163,
163,163,163,163,163
630 PRINT CHR$(31)
990 REM HAUPTPROGRAMM POSITIONSREGELUNG
1000 GOSUB 1910:REM ANFANGSWERTE SETZEN
1010 PRINT LEFT$(VT$,16);
1020 INPUT " SOLLWERT DER DREHUNG";SOLL
1030 IF SOLL>MAXIMUM THEN PRINT CHR$(147):END
1040 IF SOLL<MINIMUM THEN PRINT CHR$(147):END
1050 GOSUB 30300 :REM LIES Y-WERT
1060 IF SOLL=Y THEN GOTO 1170
1070 IF SOLL<Y THEN BIT=0
1080 IF SOLL>Y THEN BIT=1
1090 GOSUB 20100 :REM MOTOR EIN
1100 Y$=RIGHT$( " "+STR$(Y),3)
1110 PRINT LEFT$(VT$,18);
1120 PRINT " ISTWERT DER DREHUNG ";Y$
1130 GOSUB 20200 :REM MOTOR AUS
1140 GOTO 1050 :REM RUECKSPRUNG
1160 REM NEUER SOLLWERT
1170 PRINT LEFT$(VT$,16);
1180 PRINT "
1190 GOTO 1010
1900 REM ANFANGSWERTE SETZEN
1910 VT$=CHR$(19):REM CURSOR LINKS OBEN
1920 FOR I=1 TO 25
1930 VT$=VT$+CHR$(17):REM CURSOR RUNTER
1940 NEXT I
1950 INPUT " GROESSTER Y-WERT";MAXIMUM
1960 INPUT " KLEINSTER Y-WERT";MINIMUM
1970 GOSUB 10000 :REM EIN/AUSGABE VORBEREITEN
1980 PRINT
1990 PRINT " ROBOTER BEREIT? (J) "
2000 GET K$
2010 IF K$<>"J" THEN GOTO 2000
2020 PRINT LEFT$(VT$,22);
2030 PRINT " PROGRAMMSTOP MIT STOPTASTE "
2040 PRINT " ODER UNZULAESSIGEM SOLLWERT. "
2050 RETURN
2100 REM UNTERPROGRAMM GRAFIKSYMBOL
2120 REM LZ=ZAHL FUEHRENDER LEERZEICHEN
2130 REM GZ=ZAHL DER GRAFIKSYMBOL
2140 REM GZ GRAFIKSYMBOL (CHR$-CODE)
2150 READ LZ : PRINT SPC(LZ);
2160 READ GZ
2170 FOR I=1 TO GZ
2180 READ GC : PRINT CHR$(GC);
2190 NEXT I
2200 PRINT
2210 RETURN

```

Positions- regelung des Roboters mit dem C-64.

Um die Posi-
tionsregelung
abzuändern,
müssen nur
diese Zeilen ge-
ändert werden.

```

110 REM POSITIONSREGELUNG 2
220 REM BELEGUNG DER AUSGAENGE:
230 REM AUSGANG 0 - NICHT BENUTZT
240 REM AUSGANG 1 - NICHT BENUTZT
250 REM AUSGANG 2 - ARM HOCH (RECHTS)
260 REM AUSGANG 3 - ARM RUNTER (LINKS)
320 REM BELEGUNG DER EINGAENGE:
340 REM EINGANG X - ARMHOEHE
350 REM EINGANG Y - NICHT BENUTZT
1020 INPUT " SOLLWERT DER ARMHOEHE";SOLL
1050 GOSUB 30200 :REM LIES X-WERT
1060 IF SOLL=X THEN GOTO 1170
1070 IF SOLL<X THEN BIT=2
1080 IF SOLL>X THEN BIT=3
1120 PRINT " ISTWERT DER ARMHOEHE ";X$
1950 INPUT " GROESSTER X-WERT";MAXIMUM
1960 INPUT " KLEINSTER X-WERT";MINIMUM

```

```

110 REM ROBOTERSTEUERUNG 1
150 REM FUNKTION: STEUERT DEN ROBOTERS
160 REM IN SEINEN BEIDEN BEWEGUNGSACH-
170 REM SEN SOWIE DER GREIFERFUNKTION
180 REM NACH DER BEWEGUNGSTABELLE. DIE
190 REM BEWEGUNGSTABELLE WIRD AUS DATA-
200 REM ZEILEN EINGELESEN.
220 REM BELEGUNG DER AUSGAENGE:
230 REM AUSGANG 0 - DREHUNG (RECHTS)
240 REM AUSGANG 1 - DREHUNG (LINKS)
250 REM AUSGANG 2 - ARMLAENGE (HOCH)
260 REM AUSGANG 3 - ARMLAENGE (RUNTER)
270 REM AUSGANG 4 - GREIFERMAGNET
280 REM AUSGANG 5 - NICHT BENUTZT
290 REM AUSGANG 6 - NICHT BENUTZT
300 REM AUSGANG 7 - NICHT BENUTZT
320 REM BELEGUNG DER EINGAENGE:
330 REM EINGANG 0-3 - NICHT BENUTZT
340 REM EINGANG X - ARMPOSITION
350 REM EINGANG Y - DREHWINKEL
380 POKE 53281,1 :REM HINTERGRUND WEISS
390 PRINT CHR$(147); :REM BILD LOESCHEN
400 PRINT CHR$(31); :REM SCHRIFT BLAU
410 PRINT
420 PRINT" ROBOTERSTEUERUNG 1";
430 GOSUB 3250:REM DRUCKE GRAFIKSYMBOL
440 DATA 2,6,144,167,109,150,113,31
450 PRINT" -----";
460 GOSUB 3250
470 DATA 3,8,144,109,152,18,169,32,127,146
480 GOSUB 3250
490 DATA 23,6,18,32,32,32,127,146
500 GOSUB 3250
510 DATA 23,8,18,32,146,144,109,152,127,169
520 GOSUB 3250
530 DATA 23,8,18,32,146,32,144,109,109,152
540 GOSUB 3250
550 DATA 23,9,18,32,146,32,32,144,109,109,152
560 GOSUB 3250
570 DATA 17,23,18,32,32,32,146,150,113,144,18
580 DATA 96,96,96,96,96,146,167,152,18,32,146,144,165,152
590 GOSUB 3250
600 DATA 17,17,18,163,146,32,18,163,146,32,18,163,
146,32,32,18,163,146
610 GOSUB 3250
620 DATA 15,16,144,163,163,163,163,163,163,163,163,
163,163,163,163,163
630 PRINT CHR$(31);
1000 GOSUB 1920:REM ANFANGSWERTE SETZEN
1020 GOSUB 2210:REM BEWEGUNGSDATEN LESEN
1030 GOSUB 3010:REM STEUERUNG VORBEREITEN
1040 REM ROBOTERBEWEGUNG AUSFUEHREN
1050 FOR W=0 TO WEG
1060 AX=ACHS(W)
1070 S=SOLL(W)
1080 PRINT LEFT$(VT$,17);
1090 PRINT " WEGELEMENT NR. ";
1100 W$=RIGHT$( " "+STR$(W),3) :PRINT W$
1110 PRINT " ROBOTERACHSE ";AX
1120 PRINT " SOLLWERT ";
1130 S$=RIGHT$( " "+STR$(S),3) :PRINT S$
1140 ON AX+1 GOSUB 1310,1510,1710
1150 NEXT W
1160 REM BEWEGUNG ZU ENDE
1170 PRINT LEFT$(VT$,24);
1180 PRINT " WIEDERHOLEN? (J/N) "
1190 GET K$
1200 IF K$="N" THEN END
1210 IF K$<>"J" THEN GOTO 1190
1220 PRINT LEFT$(VT$,24);
1230 PRINT " PROGRAMMSTOP MIT STOPTASTE "
1240 GOTO 1050
1300 REM DREHBEWEGUNG
1310 GOSUB 30300 :REM Y-WERT
1320 IF S=Y THEN RETURN

```

Roboter-
steuerung
mit dem C-64.


```

1330 IF S<Y THEN BIT=0 :REM RECHTSLAUF
1340 IF S>Y THEN BIT=1 :REM LINKSLAUF
1350 GOSUB 20100 :REM MOTOR EIN
1360 GOSUB 30300 :REM Y-WERT
1370 IF (BIT=0) AND (S)=Y THEN GOSUB 20200 : RETURN
1380 IF (BIT=1) AND (S<=Y) THEN GOSUB 20200 : RETURN
1390 PRINT LEFT$(VT$,20);
1400 PRINT " ISTWERT ";
1410 Y$=RIGHT$(" "+STR$(Y),3) :PRINT Y$
1420 GOTO 1360
1500 REM ARMLAENGE
1510 GOSUB 30200 :REM X-WERT
1520 IF S=X THEN RETURN
1530 IF S<X THEN BIT=2 :REM RECHTSLAUF
1540 IF S>X THEN BIT=3 :REM LINKSLAUF
1550 GOSUB 20100 :REM MOTOR EIN
1560 GOSUB 30200 :REM X-WERT
1570 IF (BIT=2) AND (S)=X THEN GOSUB 20200 : RETURN
1580 IF (BIT=3) AND (S<=X) THEN GOSUB 20200 : RETURN
1590 PRINT LEFT$(VT$,20);
1600 PRINT " ISTWERT ";
1610 X$=RIGHT$(" "+STR$(X),3) :PRINT X$
1620 GOTO 1560
1700 REM MAGNET SCHALTEN
1710 BIT=4
1720 ON S+1 GOSUB 20200,20100
1730 PRINT LEFT$(VT$,20);
1740 PRINT " MAGNET ";
1750 IF S=0 THEN M$="AUS"
1760 IF S=1 THEN M$="EIN"
1770 PRINT M$
1780 REM VERZOEGERUNGSSCHLEIFE
1790 FOR Z=0 TO 500
1800 NEXT Z
1810 RETURN
1900 REM ANFANGSWERTE SETZEN
1910 REM DIMENSIONIERUNG DER GRENZWERTE
1920 DIM MAXIMUM(2),MINIMUM(2)
1930 REM DIMENSIONIERUNG DER WEGELEМЕНTE
1940 DIM ACHS(500),SOLL(500)
1950 REM CURSOR BEWEGUNGEN
1960 VT$=CHR$(19)
1970 FOR I=1 TO 25
1980 VT$=VT$+CHR$(17) :REM VERTIKAL
1990 NEXT I
2000 REM GRENZWERTE ABFRAGEN
2010 INPUT " GROESSTER X-WERT (ARM) " ;MAXIMUM(1)
2020 INPUT " KLEINSTER X-WERT " ;MINIMUM(1)
2030 INPUT " GROESSTER Y-WERT (DREH.) " ;MAXIMUM(0)
2040 INPUT " KLEINSTER Y-WERT " ;MINIMUM(0)
2050 MAXIMUM(2)=1
2060 MINIMUM(2)=0
2070 RETURN
2200 REM BEWEGUNGSDATEN EINLESEN
2210 PRINT LEFT$(VT$,18);
2220 PRINT " BEWEGUNGSDATEN LESEN!"
2230 READ WEG :REM ZAHL DER WEGELEМЕНTE-1
2240 DATA 7
2250 FOR W=0 TO WEG
2260 READ ACHS(W),SOLL(W)
2270 IF SOLL(W)<MINIMUM(ACHS(W)) THEN END
2280 IF SOLL(W)>MAXIMUM(ACHS(W)) THEN END
2290 NEXT W
2300 RETURN
2320 REM BEWEGUNGSDATEN (BEISPIEL)
2330 DATA 1,50
2340 DATA 0,130
2350 DATA 1,120
2360 DATA 2,1
2370 DATA 1,50
2380 DATA 0,90
2390 DATA 1,120
2400 DATA 2,0
3000 REM EIN/AUSGABE VORBEREITEN
3010 GOSUB 10000
3020 PRINT LEFT$(VT$,24);
3030 PRINT " ROBOTER BEREIT? (J) "
3040 GET K$
3050 IF K$(">J") THEN GOTO 3040
3060 PRINT LEFT$(VT$,24);
3070 PRINT " PROGRAMMSTOP MIT STOPTASTE"
3080 RETURN
3200 REM UNTERPROGRAMM GRAFIKSYMBOLE
3210 REM PARAMETER DER DATAZEILE:
3220 REM LZ=ZAHL FUEHRENDER LEERZEICHEN
3230 REM GZ=ZAHL DER GRAFIKSYMBOLE
3240 REM GZ GRAFIKSYMBOLE (CHR$-CODE)
3250 READ LZ : PRINT SPC(LZ);
3260 READ GZ
3270 FOR I=1 TO GZ
3280 READ GC : PRINT CHR$(GC);
3290 NEXT I
3300 PRINT

```

```

110 REM ROBOTERSTEUERUNG 2
310 :
320 REM BELEGUNG DER EINGAENGE:
330 REM EINGANG 0 - BEWEGUNGSACHSE
331 REM EINGANG 1 - RECHTSLAUF
332 REM EINGANG 2 - LINKSLAUF
333 REM EINGANG 3 - ENDE DER EINGABE
360 :
420 PRINT " ROBOTERSTEUERUNG 2";
980 :
1010 GOSUB 3010:REM STEUERUNG VORBEREITEN
2190 :
2200 REM AUFBAU DER BEWEGUNGSTABELLE
2210 PRINT LEFT$(VT$,18);
2220 PRINT " ROBOTERACHSEN: 0 - DREHUNG"
2230 PRINT " 1 - ARMLAENGE 2 - MAGNET "
2240 WEG=0 : AX=0 : M=0 : BEWEG=0
2250 PRINT LEFT$(VT$,22);
2260 PRINT " WEGELEMENT NR. " ;WEG
2270 PRINT " ACHSE (0/1/2) " ;AX
2280 :
2290 BIT=0 :REM BEWEGUNGSACHSE ?
2300 GOSUB 30100
2310 IF EIN THEN GOTO 2470
2320 GOSUB 30100 :REM ENDE TASTENDRUCK?
2330 IF NOT EIN THEN GOTO 2320
2340 IF BEWEG=0 THEN GOTO 2420
2350 :
2360 REM ABSPEICHERN
2370 ACHS(WEG)=AX
2380 SOLL(WEG)=S
2390 WEG=WEG+1:BEWEG=0
2400 :
2410 REM BEWEGUNGSACHSE WEITERSCHALTEN
2420 AX=AX+1
2430 IF AX=3 THEN AX=0
2440 GOSUB 2850:REM STATUSAUSDRUCK
2450 GOTO 2250
2460 :
2470 BIT=1 :REM RECHTS-DREHUNG?
2480 GOSUB 30100
2490 IF EIN THEN GOTO 2570
2500 BEWEG=1 :REM FLAGGE FUEER AENDERUNG
2510 IF AX=2 THEN M=1
2520 GOSUB 2850:REM STATUSAUSDRUCK
2530 IF S<MINIMUM(AX) THEN GOTO 2570
2540 BIT=2*AX :REM MOTORANWAHL
2550 GOSUB 20100 :REM EINSCHALTEN
2560 GOTO 2470:REM RUECKSPRUNG
2570 BIT=2*AX
2580 REM MOTOR AUSSCHALTEN
2590 IF AX<>2 THEN GOSUB 20200
2600 :
2610 BIT=2 :REM LINKS-DREHUNG?
2620 GOSUB 30100
2630 IF EIN THEN GOTO 2740
2640 BEWEG=1 :REM FLAGGE FUEER AENDERUNG
2650 IF AX=2 THEN M=0
2660 GOSUB 2850:REM STATUSAUSDRUCK
2670 IF AX<>2 THEN GOTO 2700
2680 BIT=4 :REM SONDERFALL MAGNET
2690 GOTO 2750
2700 IF S>MAXIMUM(AX) THEN GOTO 2740
2710 BIT=2*AX+1 :REM MOTORANWAHL
2720 GOSUB 20100 :REM EINSCHALTEN
2730 GOTO 2610
2740 BIT=2*AX+1
2750 GOSUB 20200 :REM AUSSCHALTEN
2760 :
2770 BIT=3 :REM ENDE DER AUFZEICHNUNG?
2780 GOSUB 30100
2790 IF EIN THEN GOTO 2250
2800 ACHS(WEG)=AX :REM LETZTES WEGELEMENT
2810 SOLL(WEG)=S
2820 RETURN
2830 :
2840 REM STATUSAUSDRUCK
2850 IF AX=0 THEN GOSUB 30300:S=Y
2860 IF AX=1 THEN GOSUB 30200:S=X
2870 IF AX=2 THEN S=M
2880 PRINT LEFT$(VT$,24);
2890 S$=RIGHT$(" "+STR$(S),3)
2900 IF (AX=2) AND (S=0) THEN S$="AUS"
2910 IF (AX=2) AND (S=1) THEN S$="EIN"
2920 PRINT " SOLLWERT " ;S$
2930 RETURN
2990 :

```


Einführung in die Robotik:

Die Robotfabrik

Sicher kennen Sie aus Fernsehreportagen die menschenleeren und dennoch überaus betriebsamen Werkhallen automatisierter (Auto-)Fabriken. Unser Minisystem kennt einige der Aufgaben, die damit verbunden sind. Nun soll alles harmonisch wie ein Orchester zusammengefügt werden.

Zwei prinzipielle Wege kann man beschreiten, will man eine große Anzahl von Automaten zur Zusammenarbeit veranlassen. Möglich wäre die Steuerung durch einen einzigen leistungsfähigen Computer, doch was ist, wenn der „spinnt“. Ganz zu schweigen von dem Programmpaket, das die Eigenheiten sämtlicher anzusteuender Maschinen enthalten muß.

Meist geht man den wesentlich einfacheren Weg und gibt jeder Maschine ihren eigenen Steuercomputer. Lediglich die Verbindung untereinander und die Ablaufsteuerung wird von einem Schaltwartencomputer übernommen.

Das Zusammenspiel mit anderen Computern und damit die Verwirklichung eines komplizierten Systems, kön-

nen wir auch auf der Basis unserer „Modellsteuerung“ vornehmen. Sie benötigen zu dem folgenden Experiment zwei Computer, für weitere Erforschungen gar noch mehr. Wir schlagen vor, daß Sie sich mit einem anderen Computerfreund zusammmentun. Auch stellt das folgende Experiment sicher eine gute Aufgabe für Computer-AGs an Schulen dar. Die Computer müssen übrigens gar nicht identisch sein. Da die Ausgabeplatine TTL-Pegel liefert und die Eingabeplatine TTL-Pegel verarbeitet, können Sie mit diesen beiden Platinen neben der Modellsteuerung auch für den Datenaustausch sorgen. Vergessen Sie die Masseleitungen zwischen den Platinen nicht. So ist der Weg frei, z.B. einen CPC für die Robotersteuerung zu benut-

zen, einen C64 für die Bearbeitungsstation, einen Apple II für die Fördereinrichtung usw.

In unserem Beispiel zeigen wir die Übergabe zwischen zwei Computern.

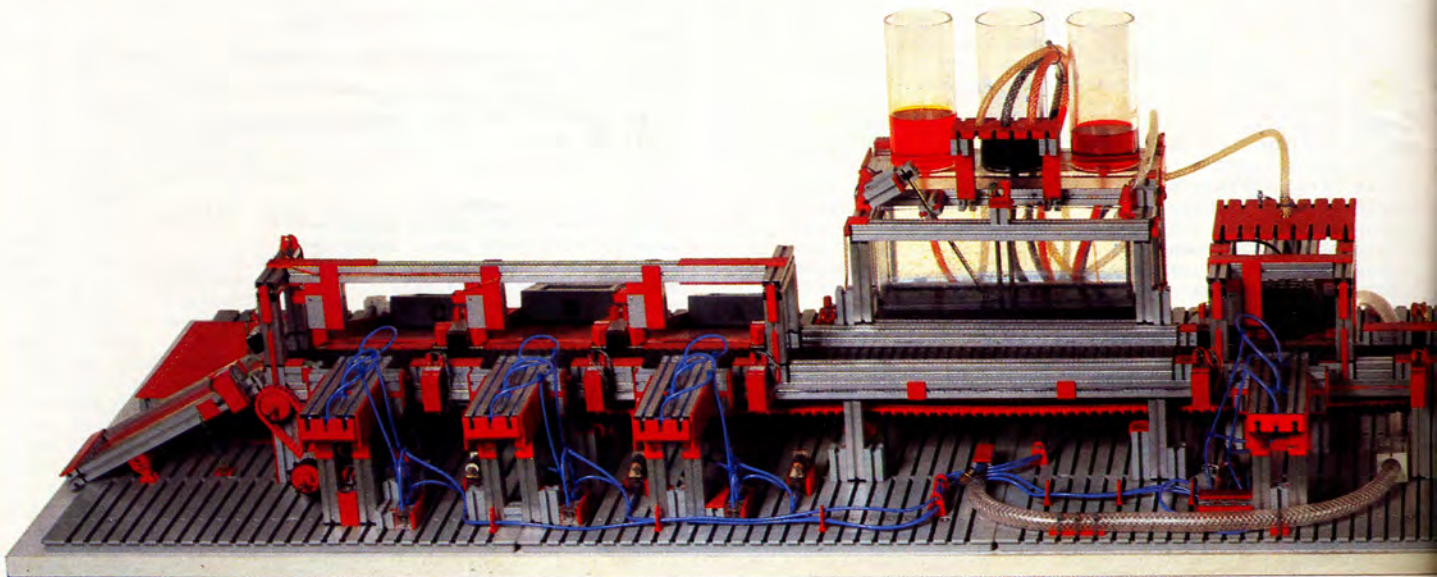
Wie **Bild 1** zeigt, kann aus einer Vielzahl solcher Zweierverknüpfungen auch eine ganze Kette aufgebaut werden. **Bild 2** zeigt eine solche Anordnung.

Bleiben wir bei einem konkreten und überschaubaren Beispiel: Computer A steuert eine Fördereinrichtung, z.B. ein Förderband. Computer B steuert einen Roboter, der die durch das Förderband angelieferten Teile dort abnehmen und an eine andere Stelle transportieren soll. Schauen wir uns zunächst das Förderband im Detail an (**Bild 3**). Der Antriebsmotor wird über eine Ausgabeplatine und Verstärkerplatine gesteuert. Die Motorplatine könnte ebenso gut anstelle der Verstärkerplatine verwendet werden, ist jedoch nicht unbedingt nötig, da der Fördermotor immer nur in einer Richtung dreht. Weiterhin steuert der Computer auch eine Anzei-

gelampe, die dem menschlichen Bediener signalisiert, wann das nächste Teil auf das Band gelegt werden kann.

Auch die Eingabeplatine wird benutzt. Zum einen ist daran ein Kommandotaster angeschlossen. Durch Betätigen dieses Tasters zeigt der Bediener dem Computer A an, daß ein Teil auf das Förderband gelegt wurde. Zum anderen ist die Bandüberwachung angeschlossen. Am Ende des Förderbandes befindet sich eine Fotozelle, wie die Detailaufnahme in **Bild 2** zeigt. Wenn das antransportierte Teil dort anlangt, wird die Fotozelle überdeckt. Der Fotowiderstand ändert seinen Widerstandswert mit der Menge des einfallenden Lichtes. Bei intensivem Lichteinfall wird er niederohmig; auf etwa 100 Ω kann der Widerstandswert absinken. Im völligen Dunkel steigt der Widerstandswert auf einige zig k Ω an. Die Abdunklung durch die Münze wird keine solch extremen Widerstandsänderungen hervorrufen, liefert aber dennoch ein brauchbares Meßsignal.

Die Steuerungsaufgabe für Computer A können wir dem Programmablaufplan in **Bild 4** entnehmen. Zu Beginn sei das Förderband leer. Ein Gegenstand (z.B. ein Groschen) wird auf das



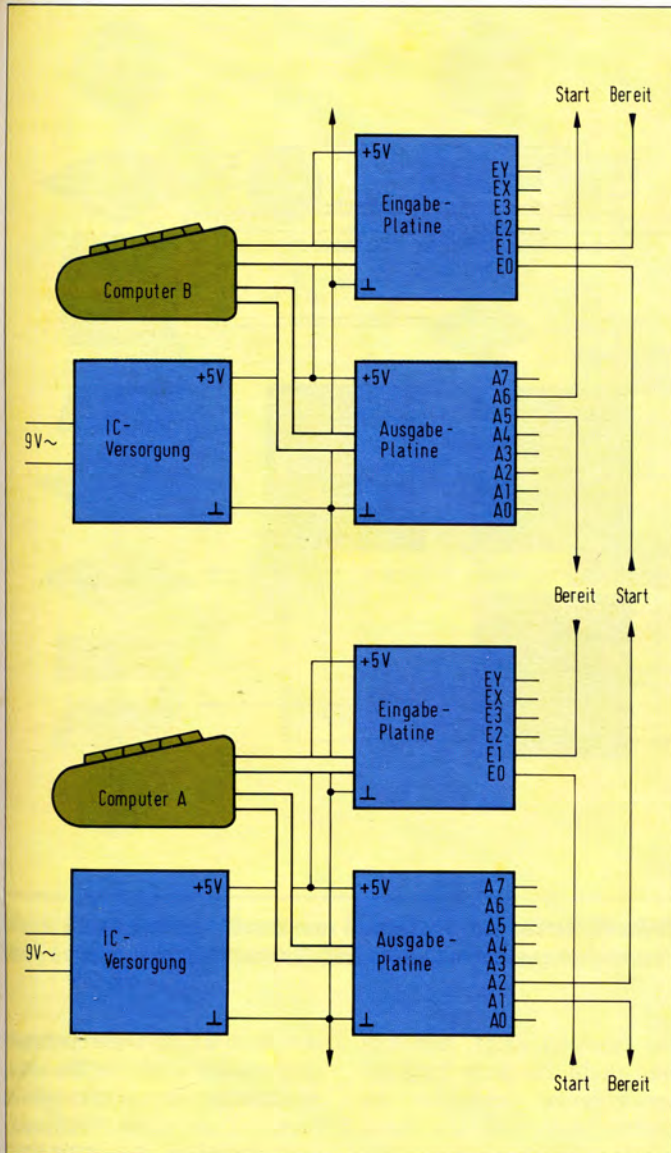


Bild 1: Signalaustausch zwischen zwei Computern A und B mit Hilfe der Signale START und BEREIT. Die Kette kann oben und unten beliebig verlängert werden.

Band gelegt und die Kommandotaste **START** gedrückt. Danach schaltet der Computer die Bereit-Anzeige aus und transportiert den Gegenstand ein Stück weit auf das Band. Die entsprechende Zeitsteuerung erfolgt über eine FOR...NEXT-Schleife, deren Zählgrenze entsprechend angepaßt wird.

Das Teil wird nicht weiter transportiert, denn am Ende des Bandes gerät es in den Arbeitsbereich des Roboters, wo es behindern könnte. Nun kann aber die Bereit-Anzeige wieder eingeschaltet werden, da der Zufuhrbereich frei ist. Werden allerdings noch eine Reihe weiterer Teile abgelegt, so ist das Band letztlich gefüllt. Es kann vorläufig nicht weiter transportiert werden und die Bereit-Anzeige bleibt auch aus.

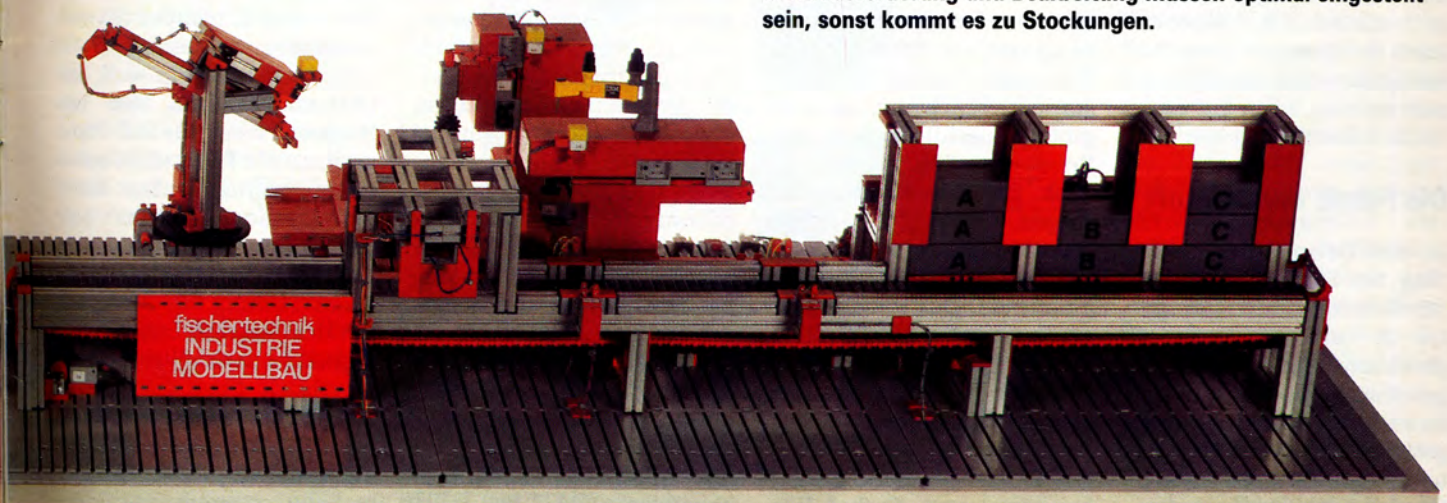
Wann erfolgt nun die Übergabe an den Roboter? Auch hier findet ein Spiel mit Bereit-Anzeige und Kommandosignal statt (**Bild 5**). Wenn der Roboter (Computer B) seine vorherige Aufgabe erledigt hat, legt er ein High-Signal auf den Ausgang **BEREIT**. Dieser ist wiederum mit einem Eingang der Förderbandsteuerung (Computer A) verbunden. Auf dieses Signal hin wird das Förderband weitertransportiert, bis der erste Gegenstand auf

die Übergabeposition über der Fotozelle kommt. Dort angekommen wird der Fördermotor wieder abgestellt. Nun kann einerseits die Bereit-Anzeige wieder eingeschaltet werden (falls sie es nicht bereits war), denn die anderen Teile auf dem Band sind ebenfalls vorgerutscht und haben den Zufuhrbereich freigegeben. Andererseits wird dem Roboter die Verfügbarkeit des Gegenstands mitgeteilt. Diesmal ist es der Ausgang **START** des Computers A, der mit einem High-Signal versehen wird. Der Ausgang ist mit einem Eingang der Eingabeplatine des Computers B verbunden, um ihm so das Startsignal zuzuspielen. Das Startsignal des Computers A wird wieder gelöscht, wenn der Computer B sein Bereitsignal auch wieder zurückgenommen hat. Ein Low-Pegel an **BEREIT** bedeutet ja gleichzeitig auch, daß Computer B auf das Startsignal reagiert hat und die Aufgabe sich in Bearbeitung befindet.

Bild 6 auf Seite 76 zeigt das Programm des Computers A. Wie schon zuvor benutzen wir die standardisierten Ein- und Ausgabeunterprogramme, die in ELO Heft 11/86 abgedruckt waren. Diese müssen entsprechend des Typs des Computers A aus-
gesucht werden und zu dem

Bild 2: Modellaufbau einer Industriesteuerung.

Weiterbeförderung und Bearbeitung müssen optimal eingestellt sein, sonst kommt es zu Stockungen.



hier abgedruckten Teil, der im wesentlichen so für alle Computertypen gilt, hinzugefügt werden.

Das Programm des Computers B (Bild 7) ähnelt weitgehend dem Roboterprogramm aus der letzten Folge. Lediglich die Synchronisation mit Förderband über die Signale BEREIT und START ist hinzugefügt worden. START startet jedesmal einen neuen Durchlauf der Roboterbewegung, zu deren Beginn das Bereitsignal gelöscht wird. Am Ende des Durchlaufs wird das Bereitsignal wieder gesetzt und auf ein erneutes Startsignal gewartet.

Händeschütteln per Computer

Ist Ihnen aufgefallen, daß die Bedienung des Förderbandes auf der Zufuhrseite, wo der Computer A sich mit einem menschlichen Benutzer abstimmen muß, genauso angelegt ist, wie die Abstimmung mit dem Computer B? Die Signale START und BEREIT haben offensichtlich eine tiefgreifendere Bedeutung. Eine derartige Synchronisation zweier computergesteuerter Systeme wird „handshake“ genannt, was direkt übersetzt Händeschütteln bedeutet. Handshake wird auch bei vielen anderen Steuerungsaufgaben benutzt. Es muß auch nicht immer ein Transport von materiellen Gegenständen sein, die unser Beispiel so anschaulich machen. Auch der Transport von Datensignalen von einem Gerät zum anderen kann auf diese Weise vonstatten gehen.

Die Fabrik der Zukunft

In dem Beispiel der Steuerung des Förderbandes und des Roboters hatten Computer A und Computer B gleichberechtigte Rollen eingenommen. Auf diese Weise ist auch die Verkettung einer größeren Zahl von Computern möglich. Lösen sie z. B. folgende erweiterte Aufga-

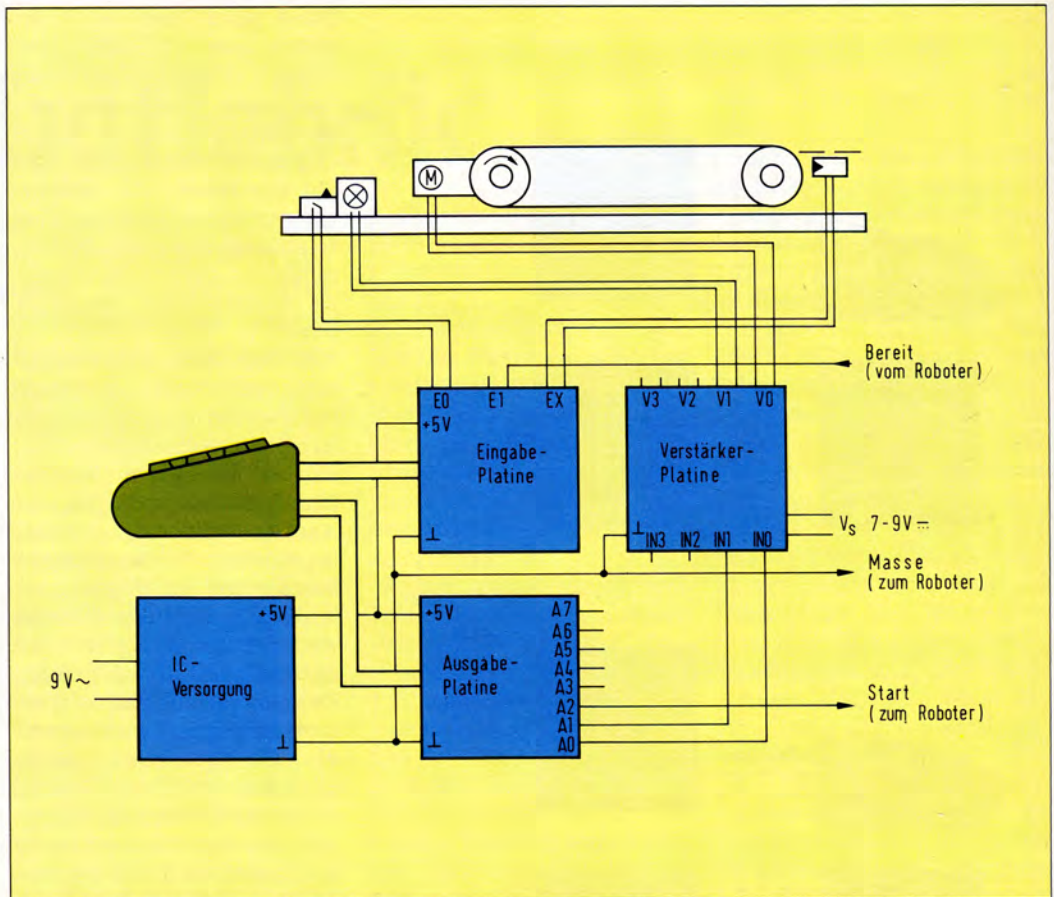


Bild 3: Blockschaltbild zum Anschluß des Förderbandes. Die Signale zum Steuern kommen vom Roboter, der Computer bereitet sie, seinem Programm entsprechend, für das Förderband auf.

be: Computer A bedient wie zuvor das Förderband. Aufbau und Programm bleiben unverändert. Der Roboter entnimmt – gesteuert von Computer B – das Teil vom Förderband. Diesmal wird es aber nicht in die Rutsche abgelegt, sondern einer entsprechend modifizierten Werkzeugmaschine zugeführt, die durch Computer C gesteuert wird. Die Werkzeugmaschine schwenkt das Teil unter die Bearbeitungsstation, bearbeitet es und gibt es anschließend an den Roboter zurück. Dieser muß auch für den Abtransport sorgen.

Damit greift der Roboter zweimal in den Bewegungsablauf ein. Das Roboterprogramm muß sowohl diese beiden Teilaufgaben untereinander als auch mit den Aufgaben der beiden anderen Computer synchronisieren. Solche getrennte Aufgaben nennt man „Tasks“. Wie

das Beispiel zeigt, benötigt man nicht für jede Task einen eigenen Computer. Es können auch mehrere Tasks in einem Computer ablaufen.

Schreiben Sie bei einer solchen Aufgabenstellung die zwei Programme als Unterprogramme hintereinander in den Arbeitsspeicher. Achten Sie darauf, daß kein Konflikt bei der Namensgebung der Variablen entsteht. Unterprogramme, die beide Programme benötigen, brauchen nur einmal formuliert werden (zumindest solange sie nicht in ihrer Ausführung unterbrochen werden können). Nun fehlt nur noch ein Hauptprogramm, das feststellt, ob die Bedingungen zur Ausführung eines der Unterprogramme erfüllt sind und dieses Unterprogramm startet. Auch solche Unterprogramme können miteinander kommunizieren. In diesem Fall müs-

sen aber keine Datenleitungen verlegt werden. Da beide Unterprogramme im gleichen Computer zuhause sind, können Sie START- und BEREIT-Meldungen über gemeinsame Nachrichtenvariablen (sog. Semaphoren) austauschen.

Was wir hier beschrieben haben, ist fast ein Multi-Tasking-System, die Basis der hohen Kunst der Steuerungsprogrammierung.

„Fast“ deswegen, weil in unserem Beispiel und im Rahmen eines BASIC-Programms die Tasks nicht quasi-gleichzeitig laufen können. Dennoch lernt man bei der Programmierung solcher Aufgaben recht viel über Multi-Tasking.

Bei dem beschriebenen Aufbau ist es nachteilig, daß ein Überblick über die Steuerung nicht so gut möglich ist, da die Informationen von allen beteiligten Bildschirmen zusammengetragen

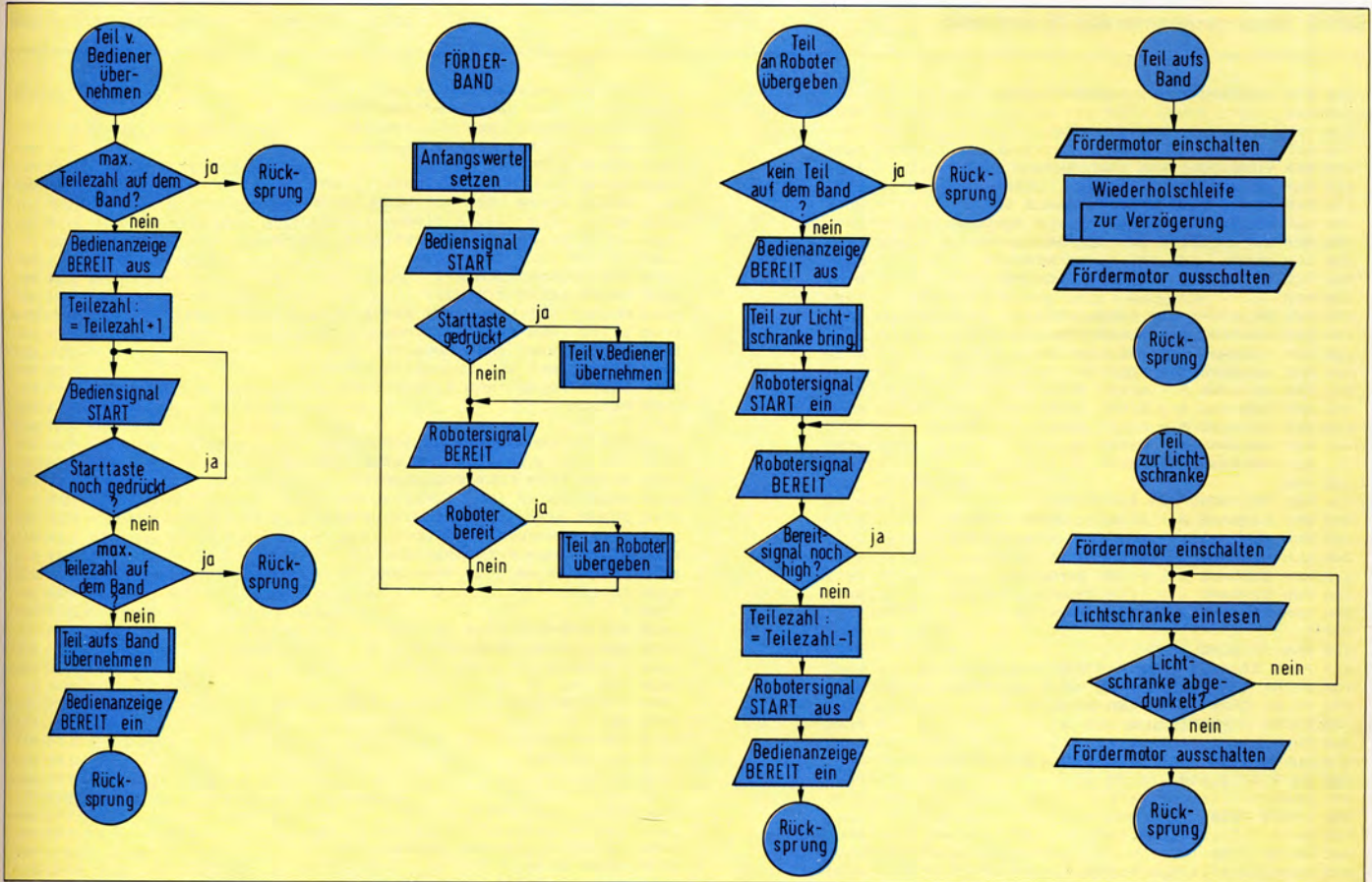


Bild 4: Programmablaufplan zum Steuerprogramm des Förderbands; es steuert den Fördermotor, verwaltet die Teilezahl auf dem Band und führt die Kommunikation mit dem Bediener und dem Roboter durch.

werden müssen. Oft baut man daher zur Steuerung großer Anlagen hierarchische Systeme.

Dies bedeutet, daß den einzelnen Computern eine Zuständigkeitsebene zugeordnet wird. Der Computer auf der höheren Ebene teilt den untergeordneten Computern Tasks zu und startet diese. Er läßt sich auch von dem Abschluß einer Task berichten, ggf. mit einer Wertung wie „erfolgreich abgeschlossen“ oder „nicht erfolgreich abgeschlossen“. Der übergeordnete Computer mag vielleicht auch auf die Erfüllung einer Task in einer bestimmten Zeit achten, um so unvorhergesehene Störfälle zu erkennen. Ein derart übergeordneter Computer bedient dann meist die Schaltwarte. Sie können auch eine solche Anlage aufbauen. Zu diesem Zweck werden die START- und BEREIT-Signale aufgetrennt und je-

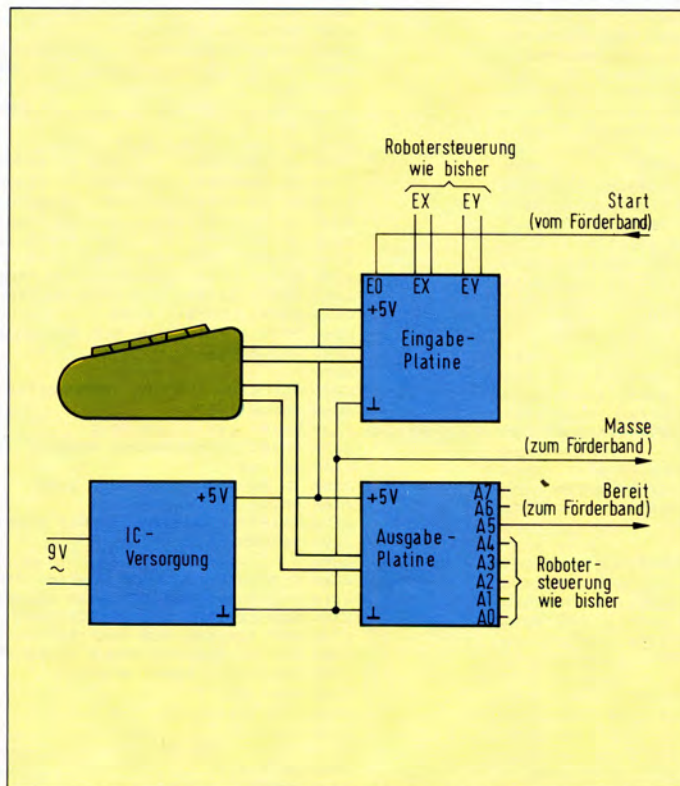


Bild 5: Die Verkabelung des Roboters wird um die Signale BEREIT und START ergänzt.

weils zu dem Schaltwartencomputer geführt. Er übernimmt nun das Durchschalten der Signale. Einen direkten Nutzen im Steuerungsablauf hat dieser Computer nicht. Da aber in ihm alle Informationen über den Steuerungsablauf vorliegen, kann er diese übersichtlich auf dem Bildschirm darstellen.

Wir wollen mit diesem eindrucksvollen Vorschlag vorläufig unsere Serie „Einführung in die Robotik“ beenden. Dies heißt nicht, daß in der ELO nichts mehr über Robotik berichtet werden würde. Im Gegenteil: wir hoffen, mit dieser Einführung gezeigt zu haben, daß auch mit einfachen Mitteln Experimente in der Robotik möglich sind. Und daß dies viele Computeristen zu eigenen Taten hinreißen wird, über die in loser Folge in der ELO berichtet werden kann.

Dr. Roland Schulé

Bild 6: Steuerprogramm des Förderbands.

```

110 REM STEUERUNG DES FOERDERBANDES
120 REM
130 REM (C) DR.R.SCHULE, 1986
140 REM
150 REM FUNKTION: DAS FOERDERBAND WIRD
160 REM IM VERBUND BETRIEBEN. UEBER
170 REM START- UND BEREITSIGNALE ER-
180 REM FOLGT DIE KOMMUNIKATION AUF DER
190 REM ZUFUHRSEITE MIT DEM BEDIENPER-
200 REM SONAL, AUF DER ENTNAHMSEITE
210 REM MIT EINER ROBOTERSTEUERUNG.
220 REM
230 REM BELEGUNG DER AUSGAENGE:
240 REM AUSGANG 0 - FOERDERBANDMOTOR
250 REM AUSGANG 1 - BEDIENANZEIGE BEREIT
260 REM AUSGANG 2 - ROBOTERSIGNAL START
270 REM AUSGANG 3 - NICHT BENUTZT
280 REM AUSGANG 4 - NICHT BENUTZT
290 REM AUSGANG 5 - NICHT BENUTZT
300 REM AUSGANG 6 - NICHT BENUTZT
310 REM AUSGANG 7 - NICHT BENUTZT
320 REM
330 REM BELEGUNG DER EINGAENGE:
340 REM EINGANG 0 - BEDIENSIGNAL START
350 REM EINGANG 1 - ROBOTERSIGNAL BEREIT
360 REM EINGANG 2 - NICHT BENUTZT
370 REM EINGANG 3 - NICHT BENUTZT
380 REM EINGANG 4 - FOERDERBANDABSTUENG
390 REM EINGANG 5 - NICHT BENUTZT
400 :
410 REM TITELMELDUNG
420 POKE 53281,1 : REM HINTERGRUND WEISS
430 PRINT CHR$(147) : REM BILD LOESCHEN
440 PRINT CHR$(31) : REM SCHRIFT BLAU
450 PRINT : PRINT
460 PRINT "FOERDERBAND":
470 GOSUB 2350 : REM DRUCKE GRAFIKSIMBOLE
480 DATA 17,3,150,110,31
490 PRINT "-----";
500 GOSUB 2350
510 DATA 7,11,150,164,164,164,164,164,164,164,164,110
520 GOSUB 2350
530 DATA 16,17,174,176,144,117,99,99,99,99
540 DATA 99,99,99,99,99,99,185,150,113
550 GOSUB 2350
560 DATA 16,20,144,172,187,186,99,99,99,99
570 DATA 99,99,99,99,99,99,107,152,18,32,146,179
580 GOSUB 2350
590 DATA 16,23,18,161,146,161,32,18,32,146,32,32,32
600 DATA 32,32,32,32,32,18,32,146,32,18,32,146
610 GOSUB 2350
620 DATA 15,20,18,162,162,162,162,162,162,162,162
630 DATA 162,162,162,162,162,162,162,162,146,31
640 :
990 REM HAUPTPROGRAMM FOERDERBAND
1000 GOSUB 1820 : REM ANFANGSWERTE
1010 BIT=0 : REM BEDIENSIGNAL "START"
1020 GOSUB 30100 : REM TESTEN
1030 REM TEIL VOM BEDIENER UEBERNEHMEN
1040 IF NOT EIN THEN GOTO 1210
1050 BIT=1 : REM ROBOTERSIGNAL "BEREIT"
1060 GOSUB 30100 : REM TESTEN
1070 REM TEIL AN ROBOTER UEBERGEBEN
1080 IF EIN THEN GOSUB 1510
1090 GOTO 1010 : REM RUECKSPRUNG
1190 :
1200 REM UNTERPROGRAMM TEIL EINGELEGT
1210 IF SPEICHER=ZAHL THEN RETURN
1215 BIT=1 : REM BEDIENANZEIGE "BEREIT"
1220 GOSUB 20200 : REM AUSSCHALTEN
1230 SPEICHER=SPEICHER+1
1240 PRINT LEFT$(VT$,22);
1250 PRINT "TEILEZAHL AUF DEM BAND";RIGHT$(" "+STR$(SPEICHER),2)
1260 BIT=0 : REM BEDIENSIGNAL "START"
1270 GOSUB 30100 : REM RUECKFLANKE TESTEN
1280 IF NOT EIN THEN GOTO 1270
1290 IF SPEICHER=ZAHL THEN RETURN
1300 REM TEIL AUF BAND UEBERNEHMEN
1310 GOSUB 1420
1320 BIT=1 : REM BEDIENANZEIGE "BEREIT"
1330 GOSUB 20100 : REM EINSCHALTEN
1340 RETURN
1390 :
1400 REM UNTERPROGRAMM BANDTRANSPORT 1
1410 REM TEILE EIN STUECK VORSCHIEBEN
1420 BIT=0 : REM FOERDERBANDMOTOR
1430 GOSUB 20100 : REM EINSCHALTEN
1440 FOR ZEIT=0 TO TRANSPRT
1450 NEXT ZEIT
1460 BIT=0 : REM FOERDERBANDMOTOR
1470 GOSUB 20200 : REM AUSSCHALTEN
1480 RETURN
1490 :
1500 REM UNTERPROGRAMM ROBOTER BEREIT
1510 IF SPEICHER=0 THEN RETURN
1520 BIT=1 : REM BEDIENANZEIGE "BEREIT"
1530 GOSUB 20200 : REM AUSSCHALTEN
1540 REM TEIL ZUR LICHTSCHRANKE BRINGEN
1550 GOSUB 1720
1560 BIT=2 : REM ROBOTERSIGNAL "START"
1570 GOSUB 20100 : REM AUF HIGH SETZEN
1580 BIT=1 : REM ROBOTERSIGNAL "BEREIT"
1590 GOSUB 30100 : REM RUECKFLANKE TESTEN
1600 IF EIN IHEN GOTO 1590
1610 SPEICHER=SPEICHER-1
1620 PRINT LEFT$(VT$,22);
1630 PRINT "TEILEZAHL AUF DEM BAND";RIGHT$(" "+STR$(SPEICHER),2)
1640 BIT=2 : REM ROBOTERSIGNAL "START"
1650 GOSUB 20200 : REM AUSSCHALTEN
1660 BIT=1 : REM BEDIENANZEIGE "BEREIT"
1670 GOSUB 20100 : REM EINSCHALTEN
1680 RETURN
1690 :
1700 REM UNTERPROGRAMM BANDTRANSPORT 2
1710 REM TRANSPORT BIS ZUR LICHTSCHRANKE
1720 BIT=0 : REM FOERDERBANDMOTOR
1730 GOSUB 20100 : REM EINSCHALTEN
1740 GOSUB 30200 : REM LICHTSCHRANKE
1750 IF X<SCHWELLE THEN GOTO 1740
1760 BIT=0 : REM FOERDERBANDMOTOR
1770 GOSUB 20200 : REM AUSSCHALTEN
1780 RETURN
1790 :
1800 REM ANFANGSWERTE SETZEN
1810 REM CURSOR-BEWEGUNGEN
1820 VT$=CHR$(19) : REM CURSOR HEIMPOS.
1830 FOR I=1 TO 25
1840 VT$=VT$+CHR$(17) : REM CURSOR RUNTER
1850 NEXT I
1860 REM FEHLWERTE SETZEN
1870 SPEICHER=0
1880 SCHWELLE=128
1890 TRANSPRT=500
1900 PRINT LEFT$(VT$,10);
1910 INPUT "MAXIMALE TEILEZAHL";ZAHL
1920 REM EIN-/AUSGABE VORBEREITEN
1930 GOSUB 2210
1940 REM FEHLWERTE ANPASSEN
1950 PRINT LEFT$(VT$,12);
1960 PRINT "F1 - SCHWELLE HOEHER"
1970 PRINT "F3 - SCHWELLE TIEFER"
1980 PRINT "F5 - TRANSPORT LAENGER"
1990 PRINT "F7 - TRANSPORT KUERZER"
2000 PRINT "F6 - PROBEITRANSPORT"
2010 PRINT "F8 - PROGRAMMSTART"
2020 PRINT LEFT$(VT$,19);
2030 GOSUB 30200:REM LICHTSCHRANKE
2040 PRINT "LICHTSCHRANKE: ";RIGHT$(" "+STR$(X),4);
2050 PRINT "SCHWELLE: ";RIGHT$(" "+STR$(SCHWELLE),4)
2060 PRINT "TRANSPORT : ";RIGHT$(" "+STR$(TRANSPORT),4);
2070 GET K$
2080 IF K$="" THEN GOTO 2020
2090 IF K$=CHR$(133) THEN IF SCHWELLE<248 THEN SCHWELLE=SCHWELLE+8
2100 IF K$=CHR$(134) THEN IF SCHWELLE>8 THEN SCHWELLE=SCHWELLE-8
2110 IF K$=CHR$(135) THEN TRANSPRT=TRANSPRT+10
2120 IF K$=CHR$(136) THEN IF TRANSPRT>10 THEN TRANSPRT=TRANSPRT-10
2130 IF K$=CHR$(139) THEN GOSUB 1420
2140 IF K$=CHR$(140) THEN GOTO 2020
2150 BIT=1 : REM BEDIENANZEIGE "BEREIT"
2160 GOSUB 20100 : REM EINSCHALTEN
2170 PRINT LEFT$(VT$,22);
2180 PRINT "TEILEZAHL AUF DEM BAND";RIGHT$(" "+STR$(SPEICHER),2)
2185 RETURN
2190 :
2200 REM EIN-/AUSGABE VORBEREITEN
2210 GOSUB 10000
2220 PRINT LEFT$(VT$,24);
2230 PRINT "FOERDERBAND BEREIT? (J)"
2240 GET K$
2250 IF K$="J" THEN GOTO 2240
2260 PRINT LEFT$(VT$,24);
2270 PRINT "PROGRAMMSIOP MIT STOPPASIE"
2280 RETURN
2290 :
2300 REM UNTERPROGRAMM GRAFIKSIMBOLE
2310 REM PARAMETER DER DATA-ZEILE:
2320 REM LZ=ZAHL DER LEERZEICHEN
2330 REM GZ=ZAHL DER GRAFIKSIMBOLE
2340 REM GZ GRAFIKSIMBOLE (CHR$(CODE))
2350 READ LZ : PRINT SPC(LZ);
2360 READ GZ
2370 FOR I=1 TO GZ
2380 READ GC
2390 PRINT CHR$(GC);
2400 NEXT I
2410 PRINT
2420 RETURN
2430 :

```


Bild 7: Steuerprogramm für den Roboter.

```

READY.
110 REM ROBSTERSTEUERUNG 4
120 REM
130 REM (C) DR.R.SCHULE, 1986
140 REM
150 REM FUNKTION: STEUERT DEN ROBSTER
160 REM IN SEINEN BEIDEN BEWEGUNGSACH-
170 REM SEN SOWIE DER GREIFERFUNKTION
180 REM NACH DER BEWEGUNGSTABELLE. DIE
190 REM BEWEGUNGSTABELLE WIRD AUS DATA-
200 REM ZEILEN EINGELESEN. DER ROBSTER
210 REM WIRD IN SEINEM ABLAUF MIT DER
220 REM STEUERUNG DES FOERDERBANDS AB-
230 REM STIMMT.
240 REM
250 REM BELEGUNG DER AUSGAENGE:
260 REM AUSGANG 0 - DREHUNG (RECHTS)
270 REM AUSGANG 1 - DREHUNG (LINKS)
280 REM AUSGANG 2 - ARMLAENGE (HOCH)
290 REM AUSGANG 3 - ARMLAENGE (RUNTER)
300 REM AUSGANG 4 - GREIFERMAGNET
310 REM AUSGANG 5 - SIGNAL "BEREIT"
320 REM AUSGANG 6 - NICHT BENUTZT
330 REM AUSGANG 7 - NICHT BENUTZT
340 REM
350 REM BELEGUNG DER EINGAENGE:
360 REM EINGANG 0 - SIGNAL "START"
370 REM EINGANG 1-3 - NICHT BENUTZT
380 REM EINGANG X - ARMPOSITION
390 REM EINGANG Y - DREHWINKEL
400 :
410 REM TITELMELDUNG
420 POKE 53281,1 :REM HINTERGRUND WEISS
430 PRINT CHR$(147); :REM BILD LOESCHEN
440 PRINT CHR$(31); :REM SCHRIFT BLAU
450 PRINT
460 PRINT " ROBSTERSTEUERUNG 4";
470 GOSUB 6050:REM DRUCKE GRAFIKSYMBOLS
480 DATA 2,6,144,167,109,150,113,31
490 PRINT " -----";
500 GOSUB 6050
510 DATA 3,8,144,109,152,18,169,32,127,146
520 GOSUB 6050
530 DATA 23,6,18,32,32,32,127,146
540 GOSUB 6050
550 DATA 23,8,18,32,146,144,109,152,127,169
560 GOSUB 6050
570 DATA 23,8,18,32,146,32,144,109,109,152
580 GOSUB 6050
590 DATA 23,9,18,32,146,32,32,144,109,109,152
600 GOSUB 6050
610 DATA 17,23,18,32,32,32,146,150,113,144,18
620 DATA 96,96,96,96,96,146,167,152,18,32,146,144,165,152
630 GOSUB 6050
640 DATA 17,17,18,163,146,32,19,163,146,32,18,163,146,32,32,32,18,163,146
650 GOSUB 6050
660 DATA 15,16,144,163,163,163,163,163,163,163,163,163,163,163,163,163,163,163,163,163
670 PRINT CHR$(31);
680 :
690 REM HAUPTPROGRAMM ROBSTERSTEUERUNG 4
1000 GOSUB 1920:REM ANFANGSWERTE SETZEN
1010 REM ROBSTERBEWEGUNG AUSFUEHREN
1020 BIT=5 :REM SIGNAL "BEREIT"
1030 GOSUB 20100 :REM AUF HIGH SETZEN
1040 PRINT LEFT$(VT$,22);
1050 PRINT " ROBSTER WARTET"
1060 BIT=0 :REM SIGNAL "START" TESTEN
1070 GOSUB 30100
1080 IF NOT EIN THEN GOTO 1070
1090 BIT=5 :REM SIGNAL "BEREIT"
1100 GOSUB 20200 :REM AUF LOW SETZEN
1110 PRINT LEFT$(VT$,22);
1120 PRINT " ROBSTER AKTIV "
1130 FOR W=0 TO WEG
1140 AX=ACHS(W)
1150 S=SOLL(W)
1160 PRINT LEFT$(VT$,18);
1170 PRINT " ROBSTERACHSE "
1180 AX%=RIGHT$(" "+STR$(AX),3) :PRINT AX%
1190 PRINT " SOLLWERT "
1200 S%=RIGHT$(" "+STR$(S),3) :PRINT S%
1210 ON AX+1 GOSUB 1310,1510,1710
1220 NEXT W
1230 GOTO 1020
1290 :
1300 REM DREHBEWEGUNG
1310 GOSUB 30300 :REM Y-WERT
1320 IF S=Y THEN RETURN
1330 IF S<Y THEN BIT=0 :REM RECHTSLAUF
1340 IF S>Y THEN BIT=1 :REM LINKSLAUF
1350 GOSUB 20100 :REM MOTOR EIN
1360 GOSUB 30300 :REM Y-WERT
1370 IF (BIT=0) AND (S=Y) THEN GOSUB 20200 : RETURN
1380 IF (BIT=1) AND (S<Y) THEN GOSUB 20200 : RETURN
1390 PRINT LEFT$(VT$,20);
1400 PRINT " ISTWERT "
1410 Y%=RIGHT$(" "+STR$(Y),3) :PRINT Y%
1420 GOTO 1360
1490 :
1500 REM ARMLAENGE
1510 GOSUB 30200 :REM X-WERT
1520 IF S=X THEN RETURN
1530 IF S<X THEN BIT=2 :REM RECHTSLAUF
1540 IF S>X THEN BIT=3 :REM LINKSLAUF
1550 GOSUB 20100 :REM MOTOR EIN
1560 GOSUB 30200 :REM X-WERT
1570 IF (BIT=2) AND (S=X) THEN GOSUB 20200 : RETURN
1580 IF (BIT=3) AND (S<X) THEN GOSUB 20200 : RETURN
1590 PRINT LEFT$(VT$,20);
1600 PRINT " ISTWERT "
1610 X%=RIGHT$(" "+STR$(X),3) :PRINT X%
1620 GOTO 1560
1690 :
1700 REM MAGNET SCHALTEN
1710 BIT=4
1720 ON S+1 GOSUB 20200,20100
1730 PRINT LEFT$(VT$,20);
1740 PRINT " MAGNET "
1750 IF S=0 THEN M$="AUS"
1760 IF S=1 THEN M$="EIN"
1770 PRINT M$
1780 RETURN
1890 :
1900 REM ANFANGSWERTE SETZEN
1910 REM DIMENSIONIERUNG DER GRENZWERTE
1920 DIM MAXIMUM(2),MINIMUM(2)
1930 REM DIMENSIONIERUNG DER WEGELEМЕНTE
1940 DIM ACHS(500),SOLL(500)
1950 REM CURSOR BEWEGUNGEN
1960 VT$=CHR$(19)
1970 FOR I=1 TO 25
1980 VT$=VT$+CHR$(17) :REM VERTIKAL
1990 NEXT I
2000 REM GRENZWERTE ABFRAGEN
2010 INPUT " GROESSTER X-WERT (ARM) " :MAXIMUM(1)
2020 INPUT " KLEINSTER X-WERT " :MINIMUM(1)
2030 INPUT " GROESSTER Y-WERT (DREH.) " :MAXIMUM(0)
2040 INPUT " KLEINSTER Y-WERT " :MINIMUM(0)
2050 MAXIMUM(2)=1
2060 MINIMUM(2)=0
2070 REM BEWEGUNGSDATEN EINLESEN
2080 PRINT LEFT$(VT$,18);
2090 PRINT " BEWEGUNGSDATEN LESEN!"
2100 READ WEG :REM ZAHL DER WEGELEМЕНTE-1
2110 DATA 7
2120 FOR W=0 TO WEG
2130 READ ACHS(W),SOLL(W)
2140 IF SOLL(W)<MINIMUM(ACHS(W)) THEN END
2150 IF SOLL(W)>MAXIMUM(ACHS(W)) THEN END
2160 NEXT W
2170 :
2180 REM BEWEGUNGSDATEN (BEISPIEL)
2190 DATA 1,50
2200 DATA 0,130
2210 DATA 1,120
2220 DATA 2,1
2230 DATA 1,50
2240 DATA 0,90
2250 DATA 1,120
2260 DATA 2,0
2270 :
2280 REM EINGABE BEENDET
2290 GOSUB 10000 :REM EIN/AUSGABE VORBEREITEN
2300 PRINT LEFT$(VT$,20);
2310 PRINT " ROBSTER BEREIT? (J)
2320 GET K$
2330 IF K$(">") THEN GOTO 2320
2340 PRINT LEFT$(VT$,24);
2350 PRINT " PROGRAMMSTOP MIT STOPTASTE";
2360 RETURN
5990 :
6000 REM UNTERPROGRAMM GRAFIKSYMBOLS
6010 REM PARAMETER DER DATAZEILE:
6020 REM LZ=ZAHL FUEHRENDER LEERZEICHEN
6030 REM GZ=ZAHL DER GRAFIKSYMBOLS
6040 REM GZ GRAFIKSYMBOLS (CHR$-CODE)
6050 READ LZ : PRINT SPC(LZ);
6060 READ GZ
6070 FOR I=1 TO GZ
6080 READ GC : PRINT CHR$(GC);
6090 NEXT I
6100 PRINT
6110 RETURN
READY.

```


Robotersteuerung mit dem CPC.

```

100 MEMORY &9FFF
110 REM Robotersteuerung 4
120 REM
130 REM (C) Dr.R.Schule, 1986
140 REM
150 REM Funktion: Steuert den Roboter
160 REM in seinen beiden Bewegungsach-
170 REM sen sowie der Greiferfunktion.
180 REM nach der Bewegungstabelle. Die
190 REM Bewegungstabelle wird aus DATA-
200 REM Zeilen eingelesen. Der Roboter
210 REM wird in seinem Ablauf mit der
220 REM Steuerung des Foerderbands ab-
230 REM abgestimmt.
240 REM
250 REM Belegung der Ausgaenge:
260 REM Ausgang 0 - Drehung (rechts)
270 REM Ausgang 1 - Drehung (links)
280 REM Ausgang 2 - Armlaenge (hoch)
290 REM Ausgang 3 - Armlaenge (runter)
300 REM Ausgang 4 - Greifermagnet
310 REM Ausgang 5 - Signal BEREIT
320 REM Ausgang 6 - nicht benutzt
330 REM Ausgang 7 - Eingabesteuerung
340 REM
350 REM Belegung der Eingaeenge:
360 REM Eingang 0 - Signal START
370 REM Eingang 1-3 - nicht benutzt
380 REM Eingang X - Armposition
390 REM Eingang Y - Drehwinkel
400 :
410 REM Titelmeldung
420 INK 0,26 :REM Hintergrund weiss
430 MODE 1 :REM Bild loeschen
440 INK 1,0 :REM Schrift schwarz
450 INK 2,6 : INK 3,13 :REM rot u. grau
460 PRINT
470 PRINT " Robotersteuerung 4";
480 GOSUB 6050 :REM Drucke Grafiksymbole
490 DATA 2,5,209,205,2,231,1
500 PRINT " -----";
510 GOSUB 6050
520 DATA 3,5,205,3,214,143,215
530 GOSUB 6050
540 DATA 23,4,143,143,143,215
550 GOSUB 6050
560 DATA 23,6,143,1,205,3,213,212
570 GOSUB 6050
580 DATA 23,6,143,32,1,205,205,3
590 GOSUB 6050
600 DATA 23,7,143,32,32,1,205,205,3
610 GOSUB 6050
620 DATA 17,17,143,143,143,2,231,1,143,143,143,
143,143,209,3,143,1,211,3
630 GOSUB 6050
640 DATA 17,9,233,32,233,32,233,32,32,233
650 GOSUB 6050
660 DATA 15,16,1,208,208,208,208,208,208,208,
208,208,208,208,208,208
680 :
990 REM Hauptprogramm Robotersteuerung 1
1000 GOSUB 1920 :REM Anfangswerte setzen
1010 REM Roboterbewegung ausfuehren
1020 bit=5 :REM Signal BEREIT
1030 GOSUB 20100 :REM auf high setzen
1040 LOCATE 1,22
1050 PRINT " Roboter wartet "
1060 bit=0 :REM Signal START testen
1070 GOSUB 30100
1080 IF NOT ein THEN GOTO 1070
1090 bit=5 :REM Signal BEREIT
1100 GOSUB 20200 :REM auf low setzen
1110 LOCATE 1,22
1120 PRINT " Roboter aktiv "
1130 FOR w=0 TO weg
1140 ax=achs(w)
1150 s=soll(w)
1160 CLS #1 :REM Fenster loeschen
1170 PRINT #1," Wegelement Nr. ";USING"###";w
1180 PRINT #1," Roboterachse ";ax
1190 PRINT #1," Sollwert ";USING"###";s
1200 ON ax+1 GOSUB 1310,1510,1710
1210 NEXT w
1220 GOTO 1020
1290 :
1300 REM Drehbewegung
1310 GOSUB 30300 :REM Y-Wert
1320 IF s=y THEN RETURN
1330 IF s<y THEN bit=0 :REM Rechtslauf
1340 IF s>y THEN bit=1 :REM Linkslauf
1350 GOSUB 20100 :REM Motor ein

```

```

1360 GOSUB 30300 :REM Y-Wert
1370 IF (bit=0) AND (s>y) THEN GOSUB 20200 : RETURN
1380 IF (bit=1) AND (s<y) THEN GOSUB 20200 : RETURN
1390 LOCATE #1,1,4
1400 PRINT #1," Istwert ";USING"###";y
1410 GOTO 1360
1490 :
1500 REM Armlaenge
1510 GOSUB 30200 :REM X-Wert
1520 IF s=x THEN RETURN
1530 IF s<x THEN bit=2 :REM Rechtslauf
1540 IF s>x THEN bit=3 :REM Linkslauf
1550 GOSUB 20100 :REM Motor ein
1560 GOSUB 30200 :REM X-Wert
1570 IF (bit=2) AND (s>x) THEN GOSUB 20200 : RETURN
1580 IF (bit=3) AND (s<x) THEN GOSUB 20200 : RETURN
1590 LOCATE #1,1,4
1600 PRINT #1," Istwert ";USING"###";x
1610 GOTO 1560
1690 :
1700 REM Magnet schalten
1710 bit=4
1720 ON s+1 GOSUB 20200,20100
1730 LOCATE #1,1,3
1740 PRINT #1," Magnet ";
1750 IF s=0 THEN PRINT #1,"aus" ELSE PRINT #1,"ein"
1760 REM Verzoeigerungsschleife
1770 FOR z=0 TO 1000
1780 NEXT z
1790 RETURN
1890 :
1900 REM Anfangswerte setzen
1910 REM Dimensionierung der Grenzwerte
1920 DIM maximum(2),minimum(2)
1930 REM Dimensionierung der Wegelemente
1940 DIM achs(500),soll(500)
1950 REM Bildschirmfenster
1960 WINDOW #1,1,40,17,20
1970 WINDOW #2,1,40,22,24
1980 REM Grenzwerte abfragen
1990 INPUT " groesster X-Wert (Arm) ";maximum(1)
2000 INPUT " kleinster X-Wert ";minimum(1)
2010 INPUT " groesster Y-Wert (Dreh.)";maximum(0)
2020 INPUT " kleinster Y-Wert ";minimum(0)
2030 PRINT " Magnet 0 - aus 1 - ein"
2040 maximum(2)=1
2050 minimum(2)=0
2060 REM Bewegungsdaten einlesen
2070 PRINT #1," Bewegungsdaten lesen!"
2080 READ weg :REM Zahl der Wegelemente-1
2090 DATA 7
2100 FOR w=0 TO weg
2110 READ achs(w),soll(w)
2120 IF soll(w)<minimum(achs(w)) THEN END
2130 IF soll(w)>maximum(achs(w)) THEN END
2140 NEXT w
2150 :
2160 REM Bewegungsdaten (Beispiel)
2170 DATA 1,150
2180 DATA 0,130
2190 DATA 1,80
2200 DATA 2,1
2210 DATA 1,150
2220 DATA 0,90
2230 DATA 1,80
2240 DATA 2,0
2250 :
2260 REM Eingabe beendet
2270 GOSUB 10010 :REM Ein/Ausgabe vorbereiten
2280 CLS #2
2290 LOCATE #2,1,3
2300 PRINT #2," Roboter bereit? (j) "
2310 k$=INKEY$
2320 IF k$<>"j" THEN GOTO 2310
2330 LOCATE #2,1,3
2340 PRINT #2," Programmstop mit 2xESC"
2350 RETURN
5990 :
6000 REM Unterprogramm GRAFIKSYMBOLE
6010 REM Parameter der DATA-Zeile:
6020 REM lz=Zahl fuehrender Leerstellen
6030 REM gz=Zahl der Grafiksymbole
6040 REM gz Grafiksymbole (CHR$-Code) bzw. PEN
6050 READ lz : PRINT SPC(lz);
6060 READ gz
6070 FOR i=1 TO gz
6080 READ gc
6090 IF gc<16 THEN PEN gc ELSE PRINT CHR$(gc);
6100 NEXT i
6110 PRINT
6120 RETURN
10000 :
10010 REM Unterprogramm INIT
10020 REM Version Schneider CPC

```



```

10030 :
10040 DIM b(7),c(7),f(3):REM Bitmuster
10050 REM b(i) zum Setzen von Bits
10060 REM c(i) zum Loeschen von Bits
10070 REM f(i) zum Maskieren der Eingänge
10080 FOR i=0 TO 7
10090 b(i)=2^i: c(i)=255-b(i)
10100 NEXT i
10110 f(0)=1: f(1)=2: REM Masken fuer
10120 f(2)=8: f(3)=64:REM Joystickeingang
10130 REM Druckeradresse zur Ausgabe
10140 a=61439
10150 REM Einsprungsadressen der
10160 REM Maschinenprogramme zur
10170 REM Eingabe der Digital- und
10180 REM Analogwerte
10190 e1234=&A001:REM Eingang e1 bis e4
10200 ex=&A005:REM Eingang Pot-X
10210 ey=&A009:REM Eingang Pot-Y
10220 REM Maschinenprogramm ablegen
10230 FOR i=0 TO 101
10240 READ code:POKE e1234+i,code
10250 NEXT i
10260 REM Ausgabeleitungen abschalten
10270 aus=0:REM Ausgänge abschalten
10280 OUT a,aus
10290 UNBREAk GUSUB 10460
10300 RETURN
10310 DATA &1e,&01,&18,&06,&1e,&10,&18,&02
10320 DATA &1e,&20,&f3,&01,&0e,&f4,&ed,&49
10330 DATA &06,&f6,&ed,&78,&e6,&30,&4f,&f6
10340 DATA &c0,&ed,&79,&ed,&49,&04,&3e,&92
10350 DATA &ed,&79,&c5,&cb,&f1,&06,&f6,&ed
10360 DATA &49,&06,&f4,&cb,&43,&20,&27,&16
10370 DATA &00,&ed,&78,&a3,&28,&06,&14,&20
10380 DATA &f8,&15,&18,&1c,&16,&00,&ed,&78
10390 DATA &a3,&20,&06,&14,&20,&f8,&15,&18
10400 DATA &0f,&16,&00,&ed,&78,&a3,&28,&08
10410 DATA &14,&20,&f8,&15,&18,&02,&ed,&50
10420 DATA &c1,&3e,&82,&ed,&79,&05,&ed,&49
10430 DATA &7a,&32,&00,&a0,&fb,&c9
10440 :
10450 REM Ein- und Ausgaben abschalten
10460 OUT a,0
10470 END
20000 :
20010 REM Unterprogramm AUSGABE
20020 REM Version Schneider CPC
20030 :
20040 REM Eingabeparameter bit (0...6)
20050 REM 20100 = Einsprung 'SETZEN'
20100 aus=aus OR b(bit)
20110 OUT a,aus
20120 RETURN
20190 REM 20200 = Einsprung 'LOESCHEN'
20200 aus=aus AND c(bit)
20210 OUT a,aus
20220 RETURN
30000 :
30010 REM Unterprogramm EINGABE
30020 REM Version Schneider CPC
30030 :
30040 REM Eingabe Digitalwerte
30050 REM Eingabeparameter bit (0...3)
30060 REM Rueckgabe 'TRUE', wenn HIGH
30070 REM Rueckgabe 'FALSE', wenn LOW
30080 REM 30100 = Einsprung
30100 OUT a,aus OR 128
30110 CALL e1234
30120 OUT a,aus AND 127
30130 ein=(PEEK(e1234-1) AND f(bit))<>0
30140 RETURN
30150 :
30170 REM Eingabe Analogwerte
30180 REM Rueckgabe 0...255
30190 REM 30200 = Einsprung X-Wert
30200 OUT a,aus OR 128
30210 REM Verzögerung zum Anschwingen
30220 FOR x=0 TO 3:NEXT
30230 CALL ex
30240 OUT a,aus AND 127
30250 x=PEEK(e1234-1)
30260 CLEAR INPUT
30270 RETURN
30290 REM 30300 = Einsprung Y-Wert
30300 OUT a,aus OR 128
30310 REM Verzögerung zum Anschwingen
30320 FOR y=0 TO 3:NEXT
30330 CALL ey
30340 OUT a,aus AND 127
30350 y=PEEK(e1234-1)
30360 CLEAR INPUT
30370 RETURN

```

Hier steuert der C-64 auch den Greifermagneten an.

```

110 REM ROBOTERSTEUERUNG 4
120 REM
130 REM (C) DR.R.SCHULE, 1986
140 REM
150 REM FUNKTION: STEUERT DEN ROBOTER
160 REM IN SEINEN BEIDEN BEWEGUNGSACH-
170 REM SEN SOWIE DER GREIFERFUNKTION
180 REM NACH DER BEWEGUNGSTABELLE. DIE
190 REM BEWEGUNGSTABELLE WIRD AUS DATA-
200 REM ZEILEN EINGELESEN. DER ROBOTER
210 REM WIRD IN SEINEM ABLAUF MIT DER
220 REM STEUERUNG DES FOERDERBANDS AB-
230 REM STIMMT.
240 REM
250 REM BELEGUNG DER AUSGÄNGE:
260 REM AUSGANG 0 - DREHUNG (RECHTS)
270 REM AUSGANG 1 - DREHUNG (LINKS)
280 REM AUSGANG 2 - ARMLÄNGE (HOCH)
290 REM AUSGANG 3 - ARMLÄNGE (RUNTER)
300 REM AUSGANG 4 - GREIFERMAGNET
310 REM AUSGANG 5 - SIGNAL "BEREIT"
320 REM AUSGANG 6 - NICHT BENUTZT
330 REM AUSGANG 7 - NICHT BENUTZT
340 REM
350 REM BELEGUNG DER EINGÄNGE:
360 REM EINGANG 0 - SIGNAL "START"
370 REM EINGANG 1-3 - NICHT BENUTZT
380 REM EINGANG X - ARMPOSITION
390 REM EINGANG Y - DREHWINKEL
400 :
410 REM TITELMELDUNG
420 POKE 53281,1:REM HINTERGRUND WEISS
430 PRINT CHR$(147):REM BILD LOESCHEN
440 PRINT CHR$(31):REM SCHRIFT BLAU
450 PRINT
460 PRINT" ROBOTERSTEUERUNG 4";
470 GOSUB 6050:REM DRUCKE GRAFIKSYMBOL
480 DATA 2,6,144,167,109,150,113,31
490 PRINT" -----";
500 GOSUB 6050
510 DATA 3,8,144,109,152,18,169,32,127,146
520 GOSUB 6050
530 DATA 23,6,18,32,32,32,127,146
540 GOSUB 6050
550 DATA 23,8,18,32,146,144,109,152,127,169
560 GOSUB 6050
570 DATA 23,8,18,32,146,32,144,109,109,152
580 GOSUB 6050
590 DATA 23,9,18,32,146,32,32,144,109,109,152
600 GOSUB 6050
610 DATA 17,23,18,32,32,32,146,150,113,144,18
620 DATA 96,96,96,96,96,146,167,152,18,32,146,146,165,152
630 GOSUB 6050
640 DATA 17,17,18,163,146,32,18,163,146,32,18,163,146,
32,32,32,18,163,146
650 GOSUB 6050
660 DATA 15,16,144,163,163,163,163,163,163,163,163,163,
163,163,163,163,163
670 PRINT CHR$(31);
980 :
990 REM HAUPTPROGRAMM ROBOTERSTEUERUNG 4
1000 GOSUB 1920:REM ANFANGSWERTE SETZEN
1010 REM ROBOTERBEWEGUNG AUSFUEHREN
1020 BIT=5:REM SIGNAL "BEREIT"
1030 GOSUB 20100:REM AUF HIGH SETZEN
1040 PRINT LEFT$(VT$,22);
1050 PRINT" ROBOTER WARTET"
1060 BIT=0:REM SIGNAL "START" TESTEN
1070 GOSUB 30100
1080 IF NOT EIN THEN GOTO 1070
1090 BIT=5:REM SIGNAL "BEREIT"
1100 GOSUB 20200:REM AUF LOW SETZEN
1110 PRINT LEFT$(VT$,22);
1120 PRINT" ROBOTER AKTIV "
1130 FOR W=0 TO WEG
1140 AX=ACHS(W)
1150 S=SOLL(W)
1160 PRINT LEFT$(VT$,18);
1170 PRINT " ROBOTERACHSE ";
1180 AX$=RIGHT$(" "+STR$(AX),3):PRINT AX$
1190 PRINT " SOLLWERT ";
1200 S$=RIGHT$(" "+STR$(S),3):PRINT S$
1210 ON AX+1 GOSUB 1310,1510,1710
1220 NEXT W
1230 GOTO 1020
1290 :
1300 REM DREHBEWEGUNG
1310 GOSUB 30300:REM Y-WERT
1320 IF S=Y THEN RETURN
1330 IF S<Y THEN BIT=0:REM RECHTSLAUF

```



```

1430 GOSUB 20100 :REM einschalten
1440 FOR zeit=0 TO transport
1450 NEXT zeit
1460 bit=0 :REM Foerderbandmotor
1470 GOSUB 20200 :REM ausschalten
1480 RETURN
1490 :
1500 REM Roboter bereit
1510 IF speicher=0 THEN RETURN
1520 bit=1 :REM Bedienanzeige BEREIT
1530 GOSUB 20200 :REM ausschalten
1540 REM Teil zur Lichtschranke bringen
1550 GOSUB 1720
1560 bit=2 :REM Robotersignal START
1570 GOSUB 20100 :REM auf High setzen
1580 bit=1 :REM Robotersignal BEREIT
1590 GOSUB 30100 :REM Rueckflanke testen
1600 IF ein THEN GOTO 1590
1610 speicher=speicher-1
1620 PRINT #1," Teilezahl auf dem Band";USING"###";speicher
1630 bit=2 :REM Robotersignal START
1640 GOSUB 20200 :REM ausschalten
1650 bit=1 :REM Bedienanzeige BEREIT
1660 GOSUB 20100 :REM einschalten
1670 RETURN
1690 :
1700 REM Bandtransport 2
1710 REM Transport bis zur Lichtschranke
1720 bit=0 :REM Foerderbandmotor
1730 GOSUB 20100 :REM einschalten
1740 GOSUB 30200 :REM Lichtschranke
1750 IF x<schwelle THEN GOTO 1740
1760 bit=0 :REM Foerderbandmotor
1770 GOSUB 20200 :REM ausschalten
1780 RETURN
1790 :
1800 REM Anfangswerte setzen
1810 REM Bildschirmfenster
1820 WINDOW #1,1,40,22,22
1830 WINDOW #2,1,40,24,24
1840 REM Fehlwerte setzen
1850 speicher=0
1860 schwelle=128
1870 transport=500
1880 LOCATE 1,10
1890 INPUT" maximale Teilezahl";zahl
1900 REM Ein-/Ausgabe vorbereiten
1910 GOSUB 2210
1920 REM Fehlwerte anpassen
1930 LOCATE 1,12
1940 PRINT" f7 - Schwelle hoeher"
1950 PRINT" f4 - Schwelle tiefer"
1960 PRINT" f1 - Transport laenger"
1970 PRINT" f0 - Transport kuerzer"
1980 PRINT" f8 - Probetransport"
1990 PRINT" f9 - Programmstart"
2000 LOCATE 1,19
2010 GOSUB 30200 :REM Lichtschranke
2020 PRINT" Lichtschranke: ";USING"#####";x;
2030 PRINT" Schwelle: ";USING"#####";schwelle
2040 PRINT" Transport : ";USING"#####";transport
2050 k$=INKEY$
2060 IF k$="" THEN GOTO 2050
2070 IF k$="7" THEN IF schwelle<248 THEN schwelle=schwelle+8
2080 IF k$="4" THEN IF schwelle>8 THEN schwelle=schwelle-8
2090 IF k$="1" THEN transport=transport+10
2100 IF k$="0" THEN IF transport>10 THEN transport=transport-10
2110 IF k$="8" THEN GOSUB 1420
2120 IF k$="9" THEN RETURN
2130 GOTO 2000
2190 :
2200 REM Ein/Ausgabe vorbereiten
2210 GOSUB 10000
2220 PRINT #2," Foerderband bereit? (j)"
2230 k$=INKEY$
2240 IF k$<>"j" THEN GOTO 2230
2250 PRINT #2," Programmstop mit 2xESC "
2260 RETURN
2290 :
2300 REM Unterprogramm GRAFIKSYMBOLS
2310 REM Parameter der DATA-Zeile:
2320 REM lz=Zahl fuehrender Leerstellen
2330 REM gz=Zahl der Grafiksymbbole
2340 REM gz Grafiksymbbole (CHR$(Code) bzw. PEN
2350 READ lz : PRINT SPC(lz);
2360 READ gz
2370 FOR i=1 TO gz
2380 READ gc
2390 IF gc<16 THEN PEN gc ELSE PRINT CHR$(gc);
2400 NEXT i
2410 PRINT
2420 RETURN
2430 :
10000 REM Unterprogramm INIT

```

```

10010 REM Version Schneider CPC
10020 :
10030 DIM b(7),c(7),f(3) :REM Bitmuster
10040 REM b(i) zum Setzen von Bits
10050 REM c(i) zum Loeschen von Bits
10060 REM f(i) zum Maskieren der Eingange
10070 FOR i=0 TO 7
10080 b(i)=2^i : c(i)=255-b(i)
10090 NEXT i
10100 f(0)=1 : f(1)=2 : REM Masken fuer
10110 f(2)=8 : f(3)=64 :REM Joystickeingang
10120 REM Druckeradresse zur Ausgabe
10130 a=61439
10140 REM Einsprungadressen der
10150 REM Maschinenprogramme zur
10160 REM Eingabe der Digital- und
10170 REM Analogwerte
10180 e1234=&A001 :REM Eingang e1 bis e4
10190 ex=&A005 :REM Eingang Pot-X
10200 ey=&A009 :REM Eingang Pot-Y
10210 REM Maschinenprogramm ablegen
10220 FOR i=0 TO 101
10230 READ code : POKE e1234+i,code
10240 NEXT i
10250 REM Ausgabeleitungen abschalten
10260 aus=0 :REM Ausgaenge abschalten
10270 OUT a,aus
10280 ON BREAK GOSUB 10450
10290 RETURN
10300 DATA &1e,&01,&18,&06,&1e,&10,&18,&02
10310 DATA &1e,&20,&f3,&01,&0e,&f4,&ed,&49
10320 DATA &06,&f6,&ed,&78,&e6,&30,&4f,&f6
10330 DATA &c0,&ed,&79,&ed,&49,&04,&3e,&92
10340 DATA &ed,&79,&c5,&cb,&f1,&06,&f6,&ed
10350 DATA &49,&06,&f4,&cb,&43,&20,&27,&16
10360 DATA &00,&ed,&78,&a3,&28,&06,&14,&20
10370 DATA &f8,&15,&18,&1c,&16,&00,&ed,&78
10380 DATA &a3,&20,&06,&14,&20,&f8,&15,&18
10390 DATA &0f,&16,&00,&ed,&78,&a3,&28,&08
10400 DATA &14,&20,&f8,&15,&18,&02,&ed,&50
10410 DATA &c1,&3e,&82,&ed,&79,&05,&ed,&49
10420 DATA &7a,&32,&00,&a0,&fb,&c9
10430 :
10440 REM Ein- und Ausgaben abschalten
10450 OUT a,0
10460 END
10470 :
20000 REM Unterprogramm AUSGABE
20010 REM Version Schneider CPC
20020 :
20030 REM Eingabeparameter bit (0...6)
20040 REM 20100 = Einsprung 'SETZEN'
20100 aus=aus OR b(bit)
20110 OUT a,aus
20120 RETURN
20130 REM 20200 = Einsprung 'LOESCHEN'
20200 aus=aus AND c(bit)
20210 OUT a,aus
20220 RETURN
20230 :
30000 REM Unterprogramm EINGABE
30010 REM Version Schneider CPC
30020 :
30030 REM Eingabe Digitalwerte
30040 REM Eingabeparameter bit (0...3)
30050 REM Rueckgabe 'TRUE', wenn HIGH
30060 REM Rueckgabe 'FALSE', wenn LOW
30070 REM 30100 = Einsprung
30100 OUT a,aus OR 128
30110 CALL e1234
30120 OUT a,aus AND 127
30130 ein=(PEEK(e1234-1) AND f(bit))<>0
30140 RETURN
30150 :
30170 REM Eingabe Analogwerte
30180 REM Rueckgabe 0...255
30190 REM 30200 = Einsprung X-Wert
30200 OUT a,aus OR 128
30210 REM Verzoeigerung zum Anschwingen
30220 FOR x=0 TO 3 :NEXT
30230 CALL ex
30240 OUT a,aus AND 127
30250 x=PEEK(e1234-1)
30260 CLEAR INPUT
30270 RETURN
30290 REM 30300 = Einsprung Y-Wert
30300 OUT a,aus OR 128
30310 REM Verzoeigerung zum Anschwingen
30320 FOR y=0 TO 3 :NEXT
30330 CALL ey
30340 OUT a,aus AND 127
30350 y=PEEK(e1234-1)
30360 CLEAR INPUT
30370 RETURN

```



```

110 REM STEUERUNG DES FOERDERBANDES
130 REM (C) DR.R.SCHULE, 1986
140 REM
150 REM FUNKTION: DAS FOERDERBAND WIRD
160 REM IM VERBUND BETRIEBEN. UEBER
170 REM START- UND BEREITSIGNALE ER-
180 REM FOLGT DIE KOMMUNIKATION AUF DER
190 REM ZUFUHRSEITE MIT DEM BEDIENPER-
200 REM SONAL, AUF DER ENTNAHMSEITE
210 REM MIT EINER ROBOTERSTEUERUNG.
220 REM
230 REM BELEGUNG DER AUSGAENGE:
240 REM AUSGANG 0 - FOERDERBANDMOTOR
250 REM AUSGANG 1 - BEDIENANZEIGE BEREIT
260 REM AUSGANG 2 - ROBOTERSIGNAL START
270 REM AUSGANG 3 - NICHT BENUTZT
280 REM AUSGANG 4 - NICHT BENUTZT
290 REM AUSGANG 5 - NICHT BENUTZT
300 REM AUSGANG 6 - NICHT BENUTZT
310 REM AUSGANG 7 - NICHT BENUTZT
320 REM
330 REM BELEGUNG DER EINGAENGE:
340 REM EINGANG 0 - BEDIENSIGNAL START
350 REM EINGANG 1 - ROBOTERSIGNAL BEREIT
360 REM EINGANG 2 - NICHT BENUTZT
370 REM EINGANG 3 - NICHT BENUTZT
380 REM EINGANG X - FOERDERBANDABTASTUNG
390 REM EINGANG Y - NICHT BENUTZT
400 :
410 REM TITELMELDUNG
420 POKE 53281,1 :REM HINTERGRUND WEISS
430 PRINT CHR$(147); :REM BILD LOESCHEN
440 PRINT CHR$(31); :REM SCHRIFT BLAU
450 PRINT : PRINT
460 PRINT " FOERDERBAND";
470 GOSUB 2350 :REM DRUCKE GRAFIKSYMBOL
480 DATA 17,3,150,110,31
490 PRINT " -----";
500 GOSUB 2350
510 DATA 7,11,150,164,164,164,164,164,164,164,164,110
520 GOSUB 2350
530 DATA 16,17,174,176,144,117,99,99,99,99
540 DATA 99,99,99,99,99,99,105,150,113
550 GOSUB 2350
560 DATA 16,20,144,172,187,106,99,99,99,99
570 DATA 99,99,99,99,99,99,107,152,18,32,146,179
580 GOSUB 2350
590 DATA 16,23,18,161,146,161,32,18,32,146,32,32,32
600 DATA 32,32,32,32,18,32,146,32,18,32,146
610 GOSUB 2350
620 DATA 15,20,18,162,162,162,162,162,162,162,162
630 DATA 162,162,162,162,162,162,162,162,146,31
640 :
990 REM HAUPTPROGRAMM FOERDERBAND
1000 GOSUB 1820 :REM ANFANGSWERTE
1010 BIT=0 :REM BEDIENSIGNAL "START"
1020 GOSUB 30100 :REM TESTEN
1030 REM TEIL VOM BEDIENER UEBERNEHMEN
1040 IF NOT EIN THEN GOSUB 1210
1050 BIT=1 :REM ROBOTERSIGNAL "BEREIT"
1060 GOSUB 30100 :REM TESTEN
1070 REM TEIL AN ROBOTER UEBERGEHEN
1080 IF EIN THEN GOSUB 1510
1090 GOTO 1010 :REM RUECKSPRUNG
1200 REM UNTERPROGRAMM TEIL EINGELEGT
1210 IF SPEICHER=ZAHLE THEN RETURN
1215 BIT=1 :REM BEDIENANZEIGE "BEREIT"
1220 GOSUB 20200 :REM AUSSCHALTEN
1230 SPEICHER=SPEICHER+1
1240 PRINT LEFT$(VT$,22);
1250 PRINT " TEILEZAHL AUF DEM BAND";RIGHT$(" "+STR$(SPEICHER),2)
1260 BIT=0 :REM BEDIENSIGNAL "START"
1270 GOSUB 30100 :REM RUECKFLANKE TESTEN
1280 IF NOT EIN THEN GOTO 1270
1290 IF SPEICHER=ZAHLE THEN RETURN
1300 REM TEIL AUF BAND UEBERNEHMEN
1310 GOSUB 1420
1320 BIT=1 :REM BEDIENANZEIGE "BEREIT"
1330 GOSUB 20100 :REM EINSCHALTEN
1340 RETURN
1390 :
1400 REM UNTERPROGRAMM BANDTRANSPORT 1
1410 REM TEILE EIN STUECK VORSCHIEBEN
1420 BIT=0 :REM FOERDERBANDMOTOR
1430 GOSUB 20100 :REM EINSCHALTEN
1440 FOR ZEIT=0 TO TRANSPRT
1450 NEXT ZEIT
1460 BIT=0 :REM FOERDERBANDMOTOR
1470 GOSUB 20200 :REM AUSSCHALTEN
1480 RETURN
1500 REM UNTERPROGRAMM ROBOTER BEREIT
1510 IF SPEICHER=0 THEN RETURN
1520 BIT=1 :REM BEDIENANZEIGE "BEREIT"

```

```

1530 GOSUB 20200 :REM AUSSCHALTEN
1540 REM TEIL ZUR LICHTSCHRANKE BRINGEN
1550 GOSUB 1720
1560 BIT=2 :REM ROBOTERSIGNAL "START"
1570 GOSUB 20100 :REM AUF HIGH SETZEN
1580 BIT=1 :REM ROBOTERSIGNAL "BEREIT"
1590 GOSUB 30100 :REM RUECKFLANKE TESTEN
1600 IF EIN THEN GOTO 1590
1610 SPEICHER=SPEICHER-1
1620 PRINT LEFT$(VT$,22);
1630 PRINT " TEILEZAHL AUF DEM BAND";RIGHT$(" "+STR$(SPEICHER),2)
1640 BIT=2 :REM ROBOTERSIGNAL "START"
1650 GOSUB 20200 :REM AUSSCHALTEN
1660 BIT=1 :REM BEDIENANZEIGE "BEREIT"
1670 GOSUB 20100 :REM EINSCHALTEN
1680 RETURN
1690 :
1700 REM UNTERPROGRAMM BANDTRANSPORT 2
1710 REM TRANSPORT BIS ZUR LICHTSCHRANKE
1720 BIT=0 :REM FOERDERBANDMOTOR
1730 GOSUB 20100 :REM EINSCHALTEN
1740 GOSUB 30200 :REM LICHTSCHRANKE
1750 IF X<SCHWELLE THEN GOTO 1740
1760 BIT=0 :REM FOERDERBANDMOTOR
1770 GOSUB 20200 :REM AUSSCHALTEN
1780 RETURN
1800 REM ANFANGSWERTE SETZEN
1810 REM CURSOR-BEWEGUNGEN
1820 VT$=CHR$(19) :REM CURSOR HEIMPOS.
1830 FOR I=1 TO 25
1840 VT$=VT$+CHR$(17) :REM CURSOR RUNTER
1850 NEXT I
1860 REM FEHLWERTE SETZEN
1870 SPEICHER=0
1880 SCHWELLE=128
1890 TRANSPRT=500
1900 PRINT LEFT$(VT$,10);
1910 INPUT " MAXIMALE TEILEZAHL";ZAHLE
1920 REM EIN-/AUSGABE VORBEREITEN
1930 GOSUB 2210
1940 REM FEHLWERTE ANPASSEN
1950 PRINT LEFT$(VT$,12);
1960 PRINT " F1 - SCHWELLE HOEHER"
1970 PRINT " F3 - SCHWELLE TIEFER"
1980 PRINT " F5 - TRANSPORT LAENGER"
1990 PRINT " F7 - TRANSPORT KUERZER"
2000 PRINT " F6 - PROBETRANSPORT"
2010 PRINT " F8 - PROGRAMMSTART"
2020 PRINT LEFT$(VT$,19);
2030 GOSUB 30200:REM LICHTSCHRANKE
2040 PRINT " LICHTSCHRANKE: ";RIGHT$(" "+STR$(X),4);
2050 PRINT " SCHWELLE: ";RIGHT$(" "+STR$(SCHWELLE),4)
2060 PRINT " TRANSPORT : ";RIGHT$(" "+STR$(TRANSPORT),4);
2070 GET K$
2080 IF K$="" THEN GOTO 2020
2090 IF K$=CHR$(133) THEN IF SCHWELLE<248 THEN SCHWELLE=SCHWELLE+8
2100 IF K$=CHR$(134) THEN IF SCHWELLE>8 THEN SCHWELLE=SCHWELLE-8
2110 IF K$=CHR$(135) THEN TRANSPRT=TRANSPRT+10
2120 IF K$=CHR$(136) THEN IF TRANSPRT>10 THEN TRANSPRT=TRANSPRT-10
2130 IF K$=CHR$(139) THEN GOSUB 1420
2140 IF K$<>CHR$(140) THEN GOTO 2020
2150 BIT=1 :REM BEDIENANZEIGE "BEREIT"
2160 GOSUB 20100 :REM EINSCHALTEN
2170 PRINT LEFT$(VT$,22);
2180 PRINT " TEILEZAHL AUF DEM BAND";RIGHT$(" "+STR$(SPEICHER),2)
2185 RETURN
2200 REM EIN-/AUSGABE VORBEREITEN
2210 GOSUB 10000
2220 PRINT LEFT$(VT$,24);
2230 PRINT " FOERDERBAND BEREIT? (J)"
2240 GET K$
2250 IF K$<>"J" THEN GOTO 2240
2260 PRINT LEFT$(VT$,24);
2270 PRINT " PROGRAMMSTOP MIT STOBTASTE"
2280 RETURN
2300 REM UNTERPROGRAMM GRAFIKSYMBOL
2310 REM PARAMETER DER DATA-ZEILE:
2320 REM LZ=ZAHLE DER LEERZEICHEN
2330 REM GZ=ZAHLE DER GRAFIKSYMBOL
2340 REM GZ GRAFIKSYMBOL (CHR$-CODE)
2350 READ LZ : PRINT SPC(LZ);
2360 READ GZ
2370 FOR I=1 TO GZ
2380 READ GC
2390 PRINT CHR$(GC);
2400 NEXT I
2410 PRINT
2420 RETURN

```

Der C-64 übernimmt
das Förderband.

Auf Profis programmiert:

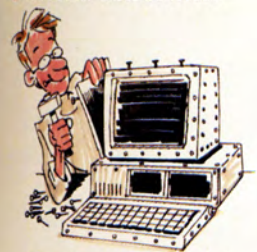


Mit **mc** lernen Sie Computer besser verstehen.

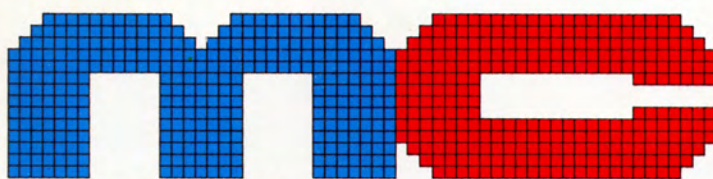
Ausführliche Funktionsbeschreibungen von Rechner-Hardware und gut kommentierte Programm-Listings werden Ihnen die richtige Freude am ernsthaften Computern bereiten.



Durch Programme in **mc** werden Sie manches Problem überhaupt nicht mehr als Problem betrachten.



Nach **mc**-Bauanleitungen löten Sie vom einfachen Interface bis zum kompletten System, was an Hardware nur schwer zu kaufen ist.



Die Mikrocomputer-Zeitschrift

Bauvorschläge:

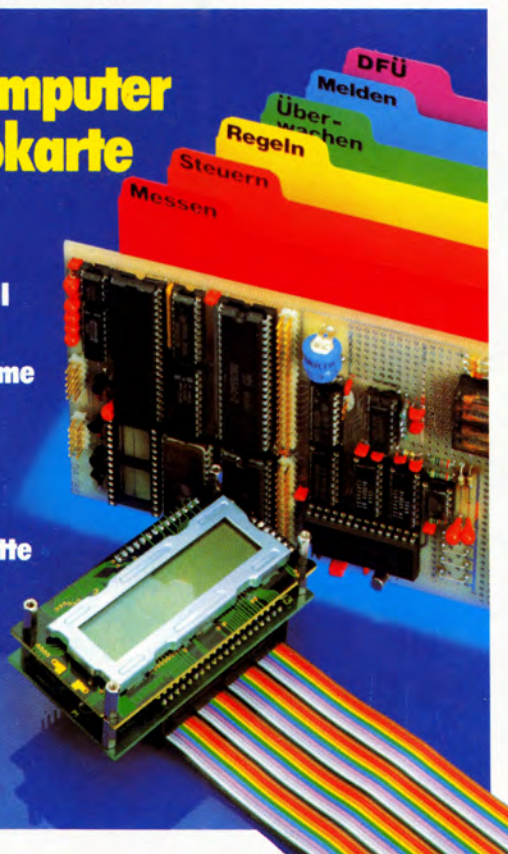
Basic-Computer auf Eurokarte

MIDI für Apple-II

Bestandsaufnahme in Sachen Btx

PC-Technik für Könner: EGA und Festplatte optimal genutzt

Tests: Schneider PC HP-18C



In **mc**-Fachaufsätzen geht's um neue Entwicklungen, um professionelle Hardware und Peripherie.



Natürlich testet **mc** Geräte und Programme. Die Ergebnisse werden aus der Sicht des professionellen Anwenders interpretiert.

Aktuelles aus der Branche zu Unternehmen, Produkten, Kongressen, Tagungen und Messen finden Sie jeden Monat in **mc**.

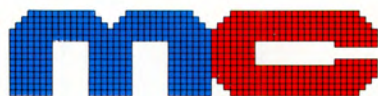
mc bringt Profis weiter.

Für DM 7,- bekommen Sie **mc** an jeder größeren Zeitschriften-Verkaufsstelle.

mc können Sie aber auch auf andere Art kennenlernen.

Kostenlos und unverbindlich.

Die Abrufkarte dafür finden Sie an der Umschlagklappe.



Die Mikrocomputer-Zeitschrift

In dieser Sonderheftreihe werden bereits in der FUNKSCHAU veröffentlichte Arbeitsblätter mit Grundsaltungen der Elektronik jeweils in einem Heft zusammengefaßt. Für den Praktiker bedeutet das eine wesentliche Arbeitshilfe.

Grundsaltungen der Elektronik

FUNKSCHAU Sonderheftreihe

In Vorbereitung:
Fernseh-Schaltungen
HF-Verstärker
Impulstechnik
Meßtechnik
Spezial-Schaltungen
Sonderthemen

Bisher erschienen:



- Nr. 1 Modulation DM 19,50**
- Der Oszillator
 - Quarzoszillatoren
 - Schwingungserzeugung mit negativen Widerständen
 - Der RC-Oszillator
 - Sinusschwingungen mit Schmitt-Trigger
 - Phasenregelkreis (PLL)
 - Der Synthesizer
 - Modulationsarten
 - Amplitudenmodulation
 - Einseitenbandmodulation
 - Quadraturmodulation
 - Phasen- und Frequenzmodulation
 - Pulsmodulation
 - Demodulation von AM-Signalen
 - Demodulation winkelmodulierter Schwingungen
 - Pulsdemodulation
 - Mischung
 - Anhang: Darstellung periodischer Funktionen durch Fourier-Reihen
 - Ausführliches Stichwortverzeichnis



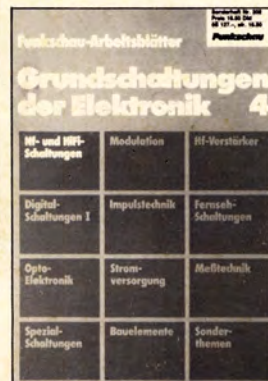
- Nr. 2 Bauelemente DM 22,50**
- Passive Bauelemente:** Widerstände, Kondensatoren, induktive Bauelemente sowie Quarze
 - Halbleiter-Bauelemente:** Grundlagen und Nomenklatur der Halbleiter, Dioden, Transistoren sowie als analoge integrierte Bauelemente, Operationsverstärker und Spannungsregler-Bausteine

- Optoelektronische Bauelemente:** Optoelektronik-Grundlagen, Foto-Widerstände, Foto-Dioden, Foto-Transistoren und -Thyristoren, LEDs und LC-Anzeigen, Optokoppler, CCD-Bauelemente und Display-Technologien. Die hier zusammengefaßten Arbeitsblätter sind nicht in der FUNKSCHAU erschienen.
- Ausführliches Stichwortverzeichnis**



- Nr. 3 Digital-Schaltungen I DM 16,-**
- Logische Grundsaltungen, Binärlogik – Binärschaltung
 - Die Logik-Familien
 - Flipflop-Grundsaltungen
 - Zähler und Teiler
 - Das Schieberegister
 - Analog-/Digital-Umsetzung
 - Digital-/Analog-Umsetzung
 - Komparatoren
 - Schaltungen für die Grundrechenarten
 - Einige besondere Verknüpfungsschaltungen
 - Der Festwertspeicher (ROM)
 - Das RAM
 - Der IEC-Bus (DIN IEC 625)
 - Ausführliches Stichwortverzeichnis

Damit Sie alle Grundsaltungen der Elektronik immer parat haben, **Sammelordner kostenlos*** (Hier können alle 12 Hefte untergebracht werden):
*Bei Bestellung ab 5 Sonderheften der Reihe „Grundsaltungen der Elektronik“



- Nr. 4 NF- und HiFi-Schaltungen DM 16,50**
- NF-Vorverstärker, Kleinsignalverstärker
 - Der Operationsverstärker – Grundlagen
 - Die Gegenkopplung
 - Die Klangregelung im NF-Verstärker
 - Die aktive Störaustattung
 - Rauschverminderung bei Tonübertragungen
 - Rundfunk-Stereoübertragung (UKW-Stereophonie)
 - FS-Stereo- und Zweittonübertragung
 - NF-Leistungsverstärker
 - Lautsprecherweichen, Aktivboxen
 - Die wichtigsten, heute angewandten Lautsprecherkonstruktionen
 - Ausführliches Stichwortverzeichnis

- Nr. 5 Optoelektronik DM 19,50**
- Licht und Strahlung
 - Leuchtdiode (LED)
 - Lumineszenz und Gasentladung
 - Flüssigkristalle
 - Lichtwellenleiter (LWL)
 - Sender und Empfänger für die Lichtleitertechnik
 - Stecker, Spleiße und Weichen für Lichtwellenleiter
 - IR-Tonübertragung, IR-Fernsteuerung
 - Optokoppler
 - Anzeigen (Displays)
 - CCD-Sensoren, Fotoelemente, pyroelektrisches Vidikon
 - Solarzelle, Solargenerator
 - Datensichtgeräte
 - Ausführliches Stichwortverzeichnis



- Nr. 6 Stromversorgung DM 18,-**
- Gleichrichter-Grundsaltungen
 - Siebung und Z-Dioden
 - Stabilisierungsschaltungen mit Regelverstärker
 - Parallel-Stabilisierungsschaltung
 - Schaltregler, Gleichspannungswandler
 - Schaltnetzteile
 - Passive Bauelemente für Schaltnetzteile
 - Aktive Bauelemente für Schaltnetzteile
 - Störspannungen und Störstrahlung
 - Einige spezielle Schaltungsbeispiele
 - Ausführliches Stichwortverzeichnis

Hier gibt es die Grundsaltungen der Elektronik:
Beim Franzis-Verlag
* per Postkarte mit Hinweis auf den gewünschten Titel. Die Rechnung wird Ihnen zugesandt
* per Scheck (Heftkosten + DM 2,-, ab 2 Sonderheften + DM 3,- Porto).

In der Schweiz:
Thali AG, Industriestraße 6, CH-6285 Hitzkirch
In Österreich:
Erb-Verlag GmbH & Co. KG, Mariahilfer Straße 71, A-1061 Wien

F Franzis-Verlag GmbH
Karlastraße 37-41
8000 München 2
Telefon (089) 5117-1