mikroelektronik

Computer

Hinweise Zur Anwendung des Lerncomputers LC 80

Teil 3

Applikation

PDF-Version: Volker Pohlers, 2004

Hinweise zur Anwendung des

Lerncomputers LC -80

Teil 3

Autor: Dipl.-Ing. Gunther Zielosko

veb mikroelektronik >karl marx< erfurt
stammbetrieb
DDR — 5010 Erfurt, Rudolfstraße 47 Telefo 5 80 Telext 061 306

Vorwort

Nach dem Erscheinen der ersten Applikationsbroschüre "Hinweise zur Anwendung des Lerncomputers LC-80" (1) gab es eine große Nachfrage speziell zu dem für den zweiten Band angekündigten Themenkreis "Programmierung und Löschung von EPROMs U 2716 C". Aua diesem Grunde hat sich der Herausgeber für eine beschleunigte Veröffentlichung einer Applikationsschrift zu diesem Thema entschieden. Diese Broschüre ist inzwischen verfügbar und kann über den

VEB Mikroelektronik "Karl Marx" Erfurt
Stammbetrieb

bezogen werden (2).

Der jetzt vorliegende dritte Teil enthält weitere typische Mikrorechneranwendungen für den Hobby-, Bildungs- und Kleinrationalisierungsbereich. Er baut auf die in den ersten beiden Teilen vermittelten Kenntnisse und die dort vorgestellten Hard- und Softwaregrundlagen auf.
Natürlich wird das Vorhandensein des LC-80 vorausgesetzt.

Es bleibt zu hoffen, daß auch dieses Buch neben dem "AHA-Effekt" eine erweiterte Anwendung von Mikrorechnerlösungen in bisher wenig betroffenen Wirtschaftszweigen bewirkt.

Inhaltverzeichnis

1. Einleitung

Wer die beiden ersten Applikationsschriften zum LC-80 bereits gelesen und die dort vorgestellten Programme in fertige Systeme umgesetzt hat, kann mit dem dritten Band jetzt einfach weitermachen.

Im vorliegenden Buch wird es um die Vertiefung der Hard- und Softwarekenntnisse gehen, wobei als Ergebnisse insbesondere universelle Schnittstellen zur Umwelt, Computerspiele aller Art, aber auch eigenständige Mikrorechnerlösungen für ein spezielles Problem entstehen werden.
Oberster Grundsatz auch dieser Broschüre ist es, alle Lösungen möglichst einfach und billig zu realisieren. Computerspezialisten mögen daher die eine oder andere unkonventionelle Programm- oder Gerätevariante verzeihen.
Aber nicht nur für den Amateur und Hobbybastler ist ein angemessenes Preis-/Leistungsverhältnis für ein gestelltes Problem von Bedeutung - auch für die gewerbliche Nutzung der Mikrorechentechnik trifft dies zu. Unter anderem aus diesem Grund wird der LC-80 in immer steigendem Maße auch für Rationalisierungszwecke eingesetzt.

Also- fangen wir einfach an, es warten interessante Aufgaben auf unseren LC-80 ...

Noch ein Hinweis!
Alle in diesem Buch aufgelisteten Programme beziehen sich auf die Ausführung des LC-80 mit einem 2 kByte EPROM U 2716 C oder Äquivalenttyp. Bei Benutzung eines Gerätes mit 2 Stück U 505 D sind alle mit einem gekennzeichneten Adressenbytes gemäß Bedienungsanleitung LC-80, S. 58 zu ändern.

2. Ein einfacher Lochstreifenleser

Im Heft 2 haben wir die Behandlung eines modernen Speichermediums - des EPROMs - kennengelernt. Jetzt wollen wir uns mit einer älteren Form der Datenspeicherung beschäftigen, dem Lochstreifen. Wozu können wir das brauchen? Ganz einfach, der Lochstreifen ist der einzige Datenträger, der alle älteren und modernen Datenverarbeitungsanlagen miteinander "kommunizieren" läßt. Es gelingt uns nämlich mit modernen Datenträgern nicht, Datenkontakt zu anderen Geräten herzustellen, da unser Magnetaufnahmeverfahren von allen in der DDR verwendeten Systemen abweicht. Der Lochstreifen überwindet dieses Handicap und ermöglicht es, Daten und Programme in den LC-80 einzuspielen. Auch hier gilt es, einfachste Lösungen anzuwenden, die trotzdem betriebssicher sind.

2.1. Grundlagen

Wie sieht ein üblicher Lochstreifen aus?
Er besitzt 8 Datenspuren (gekennzeichnet durch größere Löcher) und eine Transportspur (mit kleineren Löchern). Letztere ist durchweg gelocht und dient normalerweise als Eingriff für das Transportzahnrad. Die Anordnung der einzelnen Spuren sieht wie folgt aus (Bild 1):

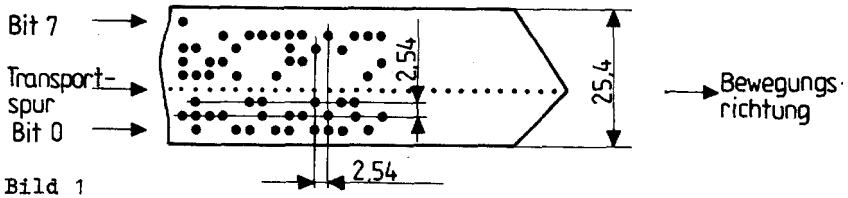


Bild 1

Wir erkennen die unsymmetrische Verteilung - einmal 5 Spuren, einmal 3 Spuren beiderseits der Transportspur. Die Zeichen (Bytes) sind senkrecht zur Transportspur zu lesen. Jedes Loch ist als logisch "1", jedes fehlende Loch als logisch "0" zu interpretieren. Damit ist eine byteweise Übernahme der Lochstreifendaten in die USER-PIO und weiter in den RAM-Speicher problemlos möglich.

Welches Abtastverfahren wählen wir?
Die Industrie kennt mechanische (z. B. mit Nadeln), optische und kapazitive Systeme. Wir entscheiden uns wegen des vergleichsweise geringen mechanischen Aufwandes für das optische Prinzip.

Wie realisieren wir die Synchronisierung zwischen Daten und Transportspur?
Indem wir auch letztere optisch abtasten. Da deren Löcher kleiner sind, sichern wir damit, daß die (größeren) Datenlöcher auf jeden Fall bereits in der richtigen Position sind, wenn ein Transportloch erkannt wird.

Wie bewegen wir den Lochstreifen?
Im einfachsten Falle mit der Hand, indem wir ihn durch unser optisches System ziehen - komfortabler mit einem kleinen Spielzeugmotor, der über ein Getriebe zwei Gummiwalzen antreibt, zwischen denen der Lochstreifen eingeklemmt ist.
Hinweise für den praktischen Aufbau im Abschnitt 2.3.

2.2. Die Elektronik

Wir benötigen für jede der 9 Spuren eine optoelektronische Auswerteschaltung. Im Interesse der sicheren Erkennung der logischen Zustände benutzen wir Fototransistoren mit nach-geschalteten CMOS-Triggern V 4093 D gemäß Schaltung in Bild 2.
Als Lichtquelle wird im einfachsten Falle eine Glühlampe in Soffittenform benutzt. Eleganter ist der Einsatz von Infrarot-Emitterdioden, wobei allerdings 9 Stück benötigt werden.
Unsere Lichtempfängereinheit muß auf engstem Raum untergebracht werden, da der Spurabstand des Lochstreifens nur 2,54 mm beträgt. Deshalb können grundsätzlich nur Fototransistoren SP 211 verwendet werden, die eine Montage im 2,5 mm-Raster zulassen. Dasselbe gilt für die Infrarot-Emitterdioden - hier wird die VQ 120 (gleiche Bauform wie SP 211! deshalb Verwechslungen vermeiden) eingesetzt. Der Widerstand R bestimmt bei gegebenen Beleuchtungsverhältnissen und Lochstreifenpapier den Eingangspegel des Triggers. Er ist so auszuwählen, daß "Loch" und "Nichtloch" sicher erkannt werden.
Da die Schaltschwelle von CMOS-Gattern etwa in der Mitte der Betriebsspannung liegt (hier also bei 2,5 V), müssen die Zustände "H" und "L" möglichst weit von dieser Mitte liegen (hochohmig messen!). Als Anhaltswert für R kann 47 kOhm genommen werden.

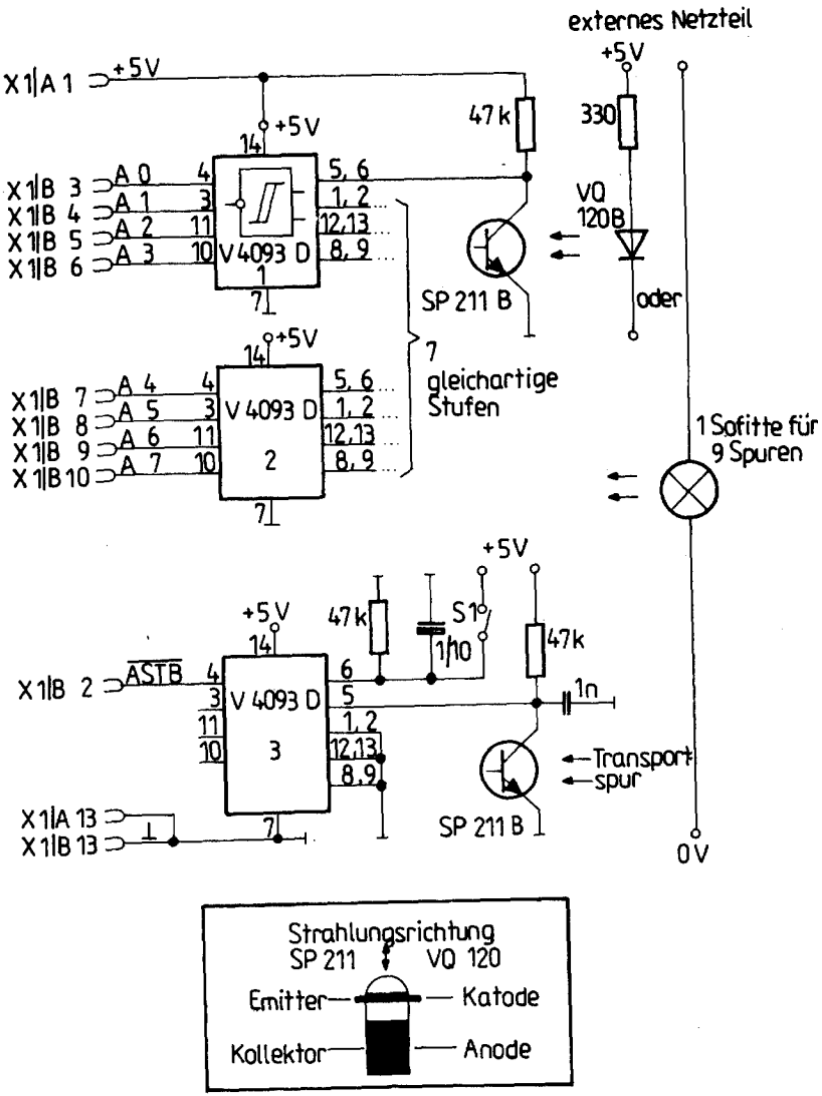


Bild 2: Elektronik für Lochstreifenleser

2.3. Mechanik

Das optische System erfordert viel Sorgfalt beim mechanischen Aufbau. Am besten ist es, sich eine genaue Bohrlehre anzufertigen, die 9 Löcher mit z. B. 1 mm Durchmesser im Abstand von genau 2,54 mm und genau auf einer Linie aufweist. Diese Bohrlehre benötigen wir zur Fixierung der Bohrungen für die Fototransistoren, die Infrarot-Emitterdioden und der Löcher der Lochstreifengleitbahn. Alle diese Löcher werden später auf genau 1,6 mm aufgebohrt (Kupferreste im Loch entfernen!).
Zuerst fertigen wir uns nun die Hauptleiterplatte an. Sie trägt die gesamte Elektronik einschließlich der Fototransistoren, der CMOS-Schaltkreise usw. Die SP 211 werden so montiert, daß die Lichtöffnungen auf der Bauelementeseite liegen. Das sind die Emitteranschlüsse, die in der Schaltung gemeinsam auf Masse liegen. Die Lötaugen für die Kollektoren sind kritisch, da wegen der 1,6 mm – Löcher nur noch sehr wenig Platz vorhanden ist. Die Fototransistoren müssen also oben und unten verlötet werden, wobei die Gefahr des Eindringens von Zinn in das Loch und damit Kurzschlußgefahr besteht.
Genauso wird die kleinere Leiterplatte hergestellt, die die Infrarot-Dioden und deren Vorwiderstände trägt. Hierbei ist es uns überlassen, ob wir Katoden oder Anoden verbinden, wir müssen das nur beim Anschluß der Spannungsquelle berücksichtigen.
Da beide Leiterplatten mit derselben Bohrschablone gebohrt wurden, ist eine hinreichende Übereinstimmung der optischen Achsen der VQ 120 und SP 211 gewährleistet. Die beiden Lochstreifenführungsplatten werden ebenfalls mit dieser Schablone gebohrt und gemäß folgendem Schema montiert (Bild 3).

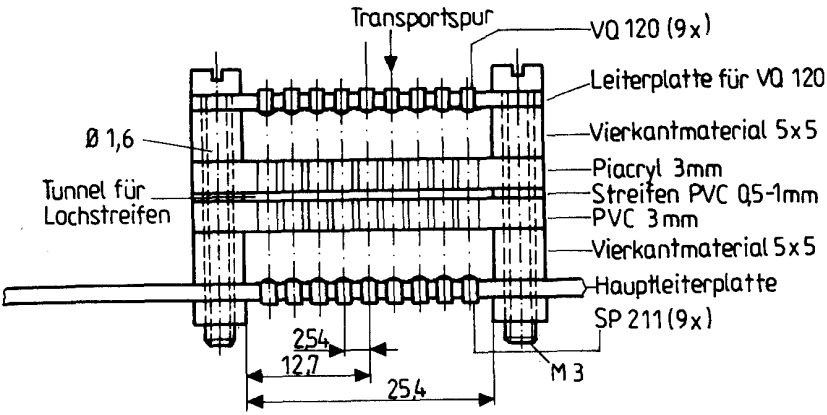


Bild 3: Querschnitt durch das optische System (M 2:1)

Der Lochstreifen gleitet dann zwischen PVC- und Piacrylplatte, die mittels Distanzeinlagen einen Abstand von ca. 0,5 ... 1 mm aufweisen. Die Länge der gesamten Führung sollte ca. 100 mm betragen.
Am Ende wird, wenn gewünscht, ein Antrieb mit kleinem Spielzeugmotor und Getriebe angeordnet. Verschiedene Transportprinzipien sind möglich:

- Antrieb des Lochstriefens durch zwei Gummiwalzen (ähnlich Tonbandantrieb), dabei muß eine Walze gefedert gelagert werden,
- Antrieb durch eine Aufwickelvorrichtung.

Wird ein Moorantrieb benutzt, ist der Einsatz eines Schalters zweckmäßig. Dieser könnte mit dem Schalter S 1 gekoppelt sein, der die Freigabe der Auswerteelektronik für die Transportspur bewirkt.

2.4. Das Programm

Ähnlich wie beim EPROM-Programmieren wollen wir auch hier einen gewissen Komfort realisieren. Deshalb wird das Programm etwas länger. Das stört uns aber nur bei der ersten Eingabe, denn danach wird es. in unserem "eigenen" EPROM abgelegt und steht dann jederzeit sofort zur Verfügung. Das Lochstreifenleseprogramm besteht aus einem Hauptprogramm und den Unterprogrammen EINGABE, DAK 3 sowie dem Interruptprogramm ISR. Das Unterprogramm DAK 3 wurde aus dem Unterprogramm DAK 2 des Betriebssystems des LC-80 abgeleitet, um kürzere Abfrage- und Anzeigezeiten zu erreichen (Anpassung an die Lochstreifengeschwindigkeit).

2.4.1. Hauptprogramm

```
2000   F3          E1: DI             Sperrung Interrupt
2001   DD 21 36 20     LD IX, 2036    Anzeige "L.S.-LES."
2005   CD 5A 04        CALL DAK 1
2008   21 F7 23        LD HL, 23F7
200B   36 48           LD M, 48       "R"
200D   2B              DEC HL                RAM-Adresse
200E   36 6F           LD M, 6F       "A"
2010   21 ED 23        LD BC, 23ED    Auf 23 ED steht Low-Teil und
                                      auf 23 EE High-Teil der
                                      Startadresse im RAM
2013   CD A7 20        CALL EINGABE
2016   CD 3C 20        CALL INITIAL
2019   21 ED 23    E2: LD HL, 23ED  ⎫ Umladen der ak-
201C   5E              LD E, M      ⎬ tuellen RAM-Adresse
201D   23              INC HL       ⎭ in DE
201E   56              LD D, M
201F   1B              DEC DE
```

```
2020   CD B7 04 *     CALL ADRSDP
2023   1A             LD A, (DE)
2024   CD 23 04 *     CALL DADP
2027   DD 21 F2 23    LD IX, 23F2    Anzeigezeigespeicicher des
                                     Betriebssystems
202B   CD 73 20       CALL DAK 3
202E   FE 01          CP 01          ST ?
2030   28 CE          JRZ CE         E1
2032   18 E5          JR E5          E2
2034   51                            Startadresse des Inter-
2035   20                            ruptprogrammes ISR
2036   BE CE C2                      1 Texttabelle
2039   08 BE D2                      "L.S.-LES."
```

2.4.2. Unterprogramm INITIAL

```
203C   F3             DI             Sperrung Interrupt
203D   ED 5E          IM 2           Interrupt-Mode 2
203F   3E 20          LD A, 20       Interruptvektor High-
2041   ED 47          LD I, A        Teil in das I-Register
2043   3E 34          LD A, 34       Interruptvektor Low-Teil
2045   D3 FA          OUT FA         Steuerwort für PIO-Port A
2047   3E 4F          LD A, 4F       PIO-Port A, Mode 1
2049   D3 FA          OUT FA         (Byte-Eingabe)
204B   3E 83          LD A, 83       Freigabe Interrupt
204D   D3 FA          OUT FA         für PIO
204F   FB             EI             Freigabe Interrupt
2050   C9             RET
```

2.4.3. Interruptprogramm ISR

```
2051   F5            PUSH AF
2052   DB F8         IN F8
2054   E5            PUSH HL
2055   D5            PUSH DE
2056   B7            OR A          A = 0? (Anfangs-00
                                         ignorieren)
2057   21 FD 23      LD HL, 23FD   Merkbit 6, ob Anfangs-00
205A   20 04         JRNZ 04       I1
2050   CB 76         BIT 6, M      bisher nur "0"?
205E   28 0D         JRZ 0D        I2
2060   CB F6      I1:SET 6, M      Merkbit 6 setzen, 1.Byte # 0
2062   21 ED 23      LD HL, 23ED
2065   5E            LD E, M       Eingelesenes Datenwort
2066   23            INC HL        in 1. RAM-Adresse ein-
2067   56            LD D, M       lesen (steht auf 23ED
2068   12            LD (DE), A    und 23EE), RAM-Adresse
2069   13            INC DE        erhöhen und diese wie-
206A   72            LD M, D       der in 23ED und 23EE
206B   2B            DEC HL        abspeichern
206C   73            LD M, E
206D   D1         I2:POP DE
206E   E1            POP HL
206F   F1            POP AF
2070   FB            EI
2071   ED 4D         RETI
```

2.4.4. Unterprogramm DAK 3

```
2073   37            SCF
2074   08            EX AF
2075   D9            EX X
2076   0E 00         LD C, 00
2078   1E FB         LD E, FB
207A   DD 7E 00   D2: LD A, (IX + 00)
207D   2F            CPL
207E   D3 F4         OUT F4
2080   7B            LD A, E
2081   D3 F5         OUT F5
2083   06 18         LD B, 18       Verkürzte Zeitschleife
2085   10 FE         DJNZ FE        in DAK 2 ist B = 64
2087   06 04         LD B, 04
2089   DB F9         IN F9
208B   57            LD D, A
2080   CB 12      D1: RL D
208E   38 02         JRC 02
2090   79            LD A, C
2091   08            EX AF
2092   0C            INC C
2093   10 F7         DJNZ F7        D1
2095   DD 23         INC IX
2097   CB 03         RLC E
2099   3E FF         LD A, FF
209B   D3 F5         OUT F5
209D   38 DB         JRC DB         D2
209F   11 FA FF      LD DE, FFFA
20A2   DD 19         ADD IX, DE
20A4   D9            EX X
20A5   08            EX AF
20A6   C9            RET
```

2.4.5. Unterprogramm EINGABE

Da dieses Unterprogramm sehr vielseitig einsetzbar ist, sollen an dieser Stelle einige Hinweise zur Modifikation gegeben werden. Das Unterprogramm ermöglicht das Einschreiben von Ziffern in einen durch HL festgelegten RAM-Bereich und die Darstellung auf dem Display. Der Eingabemodus ist ähnlich wie bei Taschenrechnern und entspricht etwa dem Eintragen einer Adresse beim LC-80 mittels Tastatur.
Dabei können folgende Parameter gewählt werden:

- Der Inhalt von B legt die Stellenzahl der "rotierenden" Zeichen fest, im folgenden Unterprogramm z. B. 04, d. h. 4 Stellen werden angezeigt.
- Der Inhalt von C bestimmt den Zeichenvorrat, hier 10, d. h. alle Zeichen von 0 ... F sind erlaubt. Wird nur dezimale Eingabe erwünscht, muß C = 0A sein. Sollen nur die Ziffern 0 ... 6 erlaubt sein, wie das für eine Uhrzeiteingabe sinnvoll sein kann, wird C = 07.
- Der Inhalt von E gibt die Anzahl der belegbaren RAM-Zellen an. Im folgenden Anwendungsfall ist E = 02. Damit kommen wir aus, denn 2 Zellen speichern 2 Bytes und das sind 4 Hexa-Ziffern.
  In anderen Anwendungsfällen des Unterprogrammes könnte es aber nützlich sein, ganze Zahlenkolonnen eintragen zu können, wovon dann natürlich nur die letzten n (n ist in B festgelegt) angezeigt werden.
- IX zeigt hier auf den Anzeigebereich des Betriebssystems, kann aber auch anderweitig benutzt werden.

Die Festlegung obiger Parameter erfolgt am Anfang des Unterprogrammes EINGABE. Werden andere Parameter benötigt, erfolgt der Einsprung erst auf Adresse 20B0.

```
20A7   01 10 04       LD B0, 0410
20AA   1E 02          LD E, 02
20AC   DD 21 F2 23    LD IX, 23F2
20B0   DD E5       T5:PUSH IX
20B2   E5             PUSH HL
20B3   C5             PUSH BC
20B4   7E             LD A, M
20B5   F5             PUSH AF
20B6   CD CA 04 * T2: CALL ONESEG
20B9   DD 77 00       LD (IX + 00), A
20BC   DD 23          INC IX
20BE   F1             POP AF
20BF   05             DEC B
20C0   28 0F          JRZ 0F               T1
20C2   0F             RRC A
20C3   0F             RRC A
20C4   0F             RRC A
20C5   0F             RRC A
20C6   CD CA 04 *     CALL ONESEG
20C9   DD 77 00       LD (IX + 00), A
20CC   DD 23          INC IX
20CE   23             INC HL
20CF   10 E3          DJNZ E3              T2
20D1   C1          T1:POP BC
20D2   El             POP HL
20D3   DD El          POP IX
20D5   C5          T3:PUSH BC
20D6   E5             PUSH HL
20D7   CD 5A 04 *     CALL DAK 1
20DA   E1             POP HL
20DB   C1             POP BC
20DC   FE 12          CP 12                EX
20DE   C8             RET Z
20DF   B9             CP C
20E0   F3             JRNC F3              T3
20E2   E5             PUSH HL
```

```
20E3   C5            PUSH BC
20E4   43            LD B, E
20E5   ED 6F      T4:RLD
20E7   23            INC HL
20E8   10 FB         DJNZ FB       T4
20EA   C1            POP BC
20EB   E1            POP HL
20EC   18 C2         JR C2         T5
20EE   FF
```

Das war es schon! Sicherheitshalber retten wir das ganze Programm auf Kassette (Anfangsadresse 2000, Endadresse 20EE), um uns das nochmalige Eintippen bei Bedienfehlern zu ersparen.

Nun müssen Elektronik, Mechanik und Programmsystem zum Leben erweckt werden. Grundsätzliche Funktionskontrollen sind schon erfolgt:

- Die Leiterplatte wurde auf richtige Schaltung und Anschlußbelegung geprüft. Die Funktion aller 9 Lichtschranken ist gewährleistet, an den entsprechenden Ausgängen der drei V 4093 D konnten die beiden Zustände Loch/Nichtloch elektrisch nachgewiesen werden.
- Der Lochstreifenantrieb funktioniert, optisch stimmt alles, nichts klemmt oder hat zuviel Spiel.
- Das Programm wurde gründlich auf Eingabefehler kontrolliert.

Dann wird die Leiterplatte in den abgeschalteten LC-80 gesteckt und eingeschaltet. Dann starten wir das Programm nach dem Einlesen der Kassette mit EX auf Adresse 2000. Wenn alles funktioniert, erscheint nun

L.S.- L E S.

Danach wird ST betätigt, es erfolgt die Anzeige:

r A F F F F ,

was soviel heißt wie RAM-Adresse, in die unser erstes Lochstreifen-Byte geladen werden soll. Diese RAM-Adresse- geben wir nun durch die Hexa-Tastatur ein, bis z. B.:

r A 2 4 0 0 (RAM-Erweiterung vorausgesetzt!)

angezeigt wird. Wiederum durch EX wird das Eingabeprogramm für Lochstreifen aktiviert, es erscheint in der Anzeige:

2 3 F F F F .

Das ist genau die Adresse vor 2400 mit den Daten FF.
Wenn wir jetzt (erst einmal ohne Motorantrieb) einen Lochstreifen einführen, dann den Schalter S 1 betätigen, springt die Adressenanzeige auf 2400, wenn das erste Zeichen auf dem Lochstreifen erscheint. Eine Besonderheit ist in unser Programm eingebaut; der Leser ignoriert den nicht gelochten Anfang des Lochstreifens (00). Erst das erste Zeichen mit mindestens einem Loch (zusätzlich zur Transportspur) wird "ernstgenommen" und auf Adresse 2400 abgespeichert. Ab jetzt ist auch das Zeichen 00 erlaubt - es wird genauso als Byte behandelt wie alle anderen.
Wir überprüfen nun die richtige Übernahme der Zeichen in den RAM-Speicher ab 2400. Sollte etwas nicht funktionieren, liegt es meist an dem nicht präzise genug aufgebauten optischen System. Auch der zeitliche Durchgang der Lochränder (Flanken am Trigger) kann Fehler verursachen - besonders kritisch ist hier die zeitliche Lage des Transportspur-Impulses. Gegebenenfalls muß hier der Kondensator 1 nF am Eingang 5 des V 4093 D (3) vergrößert werden. Damit wird dieser etwas verzögert und die Datenspuren haben inzwischen ihre logischen Zustände sicher erreicht. Diese Optimierung muß natürlich auf die Transportgeschwindigkeit des Lochstreifens abgestimmt werden (mit Motorantrieb).

Wenn dann alles richtig funktioniert, nehmen wir uns jetzt ein etwa 2 m langes Stück Lochstreifen (das entspricht etwa 1 kByte) mit einem ca. 10 cm langen Anfang ohne Datenlöcher. Dieser Lochstreifen soll zur Kontrolle der zuverlässigen Datenübernahme benutzt werden - sein Inhalt ist gleichgültig. Diesen Lochstreifen schieben wir so in das optische System ein, daß alle 8 Datenspuren abgedeckt sind - richtige Lage der Transportspur beachten!
Das Programm wird mit:
RES, ADR, EX, ST, 2, 4, 0, 0 und EX
aktiviert, S 1 und danach der Antrieb (oder gleichzeitig mit S 1) eingeschaltet. Wir beobachten dabei auf dem Display das Hochzählen der Adressen von 2400 bis etwa 2800. Danach legen wir den Lochstreifen noch einmal ein und wiederholen des Ganze mit:
RES, ADR, EX, ST, 2, 8, 0, 0 und EX
Danach werden 31 und der Motor eingeschaltet und der Lochstreifeninhalt nun auf die Adressen ab 2800 geladen. Wir können jetzt die Inhalte der Speicherzellen

2400 mit 2800,
2401 mit 2801 usw.

vergleichen, was "zu Fuß" aber sehr mühsam ist.
Folgendes Programm nimmt uns das ab:

```
2200   21 00 24       LD HL, 2400
2203   11 00 28       LD DE, 2800
2206   1A          V1: LD A, (DE)
2207   BE              CP M
2208   20 04           JRNZ 04        V2
220A   23              INC HL
220B   13              INC DE
220C   18 F8           JR F8          V1
220E   CD B7 04 * V2: CALL ADRSDP
```

```
2211   CD 5A 04 *     CALL DAK 1
2214   76             HALT
```

Wenn wir es starten, wird die erste im Inhalt nicht übereinstimmende Adresse angezeigt.
Das wird dann überprüft - wahrscheinlich ist es das Ende des Lochstreifens oder, wenn dieser länger als 1 kByte war, die Adresse 2000, da deren Inhalt ja der im 2. Durchlauf überschriebenen Adresse 2800 nicht mehr entsprechen kann.
Wenn alles seine Richtigkeit hat, werden wir das Programm in unserem EPROM speichern.

2.5. Lochstreifenprogramm im EPROM

Unser erster selbstprogrammierter EPROM enthält zur Zeit nur das EPROM-Ladeprogramm (siehe Heft 2). Zusätzlich soll dort das Lochstreifenprogramm untergebracht werden - eine gute Übung zum EPROM-Programmieren.
Da wir EPROMs nur mit dem auf dem EPROM selbst stehenden Ladeprogramm laden können, ist (wenigstens "leihweise") ein 2. leerer EPROM U 2716 C erforderlich.
Der LC-80 wird ausgeschaltet (das Lochstreifenprogramm war ja auf Kassette geladen) und das EPROM-Programmierboard in den USER-Bus gesteckt. Danach nehmen wir den LC-80 wieder in Betrieb und schalten die Programmierspannung zu.
Zuerst übernehmen wir jetzt das EPROM-Ladeprogramm:

RES, ST, 2, 0, 0, +, 1, 0, 0,
0, +, 0, RES, ADR, 1, 0, 0, B,
EX, ST

Das ist die Eingabeanweisung für die EPROM-Teilprogrammierung von Adresse 1000 bis 1200 auf die Adressen 0000 bis 0200 des leeren 2. EPROMs.
Jetzt laden wir das Lochstreifenprogramm von der Kassette auf die Adressen von 2000 an. Ist das erfolgt, müssen die Inhalte folgender RAM-Zellen geändert werden:

Adresse	alte Daten	neue Daten
2004	20	12
2015	20	12
2018	20	12
202D	20	12
2035	20	12
2040	20	12

Danach wird das gesamte Programm nochmals überprüft - bis auf die geänderten Daten muß es identisch sein mit dem Programm von Abschnitt 2.4.
Jetzt kommt auch das in den EPROM:

RES, ST, 1, 0, 0, +, 2, 0, 0,
0, +, 2, 0, 0, RES, ADR, 1, 0,
0, B, EX, ST

Wenn alles richtig übernommen wurde, haben wir jetzt einen EPROM mit beiden Programmen, den wir dann (natürlich bei abgeschaltetem LC-80) wieder in die ROM-Position 3 stecken können. Nach dem Einschalten erreichen wir auf der Adresse

1000 das EPROM-Ladeprogramm und auf
1200 das Lochstreifen-Leseprogramm.

Die Bedienung ist denkbar einfach. Nach Aufruf der Adresse 1200 und EX wird durch

L.S.- L E S.

die Bereitschaft angezeigt, durch ST wird die erste RAM-Adresse, auf die die Lochstreifendaten geladen werden sollen, ausgewählt. Nach erneuter Betätigung von EX kann die Übertragung beginnen. Dazu ist S 1 (und der Antrieb) auf dem Lochstreifenleser einzuschalten.

Abschließend noch ein Wort zur praktischen Auswertung von Lochstreifendaten. Zwar sind die Abmessungen der Lochstreifen international genormt, die gelochten Daten sind aber leider nicht einheitlich kodiert. So gibt es z. B. den R 300-Code, den ASCII-Code oder den ISO-7bit-Code. Im Gegensatz zur LC-80-Sprache, bei der ein Byte zwei Zeichen charakterisiert (z. B. CD) stellt in den o. g. Codes ein Byte auch nur ein (allerdings alphanumerisches) Zeichen dar. Meist reichen zur Zeichendarstellung 7 bit aus, das 8. Bit wird zur Paritätskontrolle benutzt. Damit wird eine einfache Fehlererkennung möglich. Wenn wir solche Lochstreifen interpretieren wollen, müssen wir also ihre Kodierung kennen. Dann lassen sich die eingelesenen Daten durch geeignete Umformung auch für den LC-80 auswerten.
Direkt nutzbar sind solche Lochstreifen, die unmittelbar hexadezimal kodierte Informationen enthalten. Viele EDV-Anlagen sind z. B. in der Lage, eingelesene ROM-Inhalte o. ä. im Hexa-Code auszugeben.

3. Motoren gehorchen Befehlen

Alle bisherigen Experimente spielten sich fast ausschließlich auf elektronischem Gebiet ab. Das wird jetzt anders - wir wollen per Software mechanische Wirkungen durch unseren Computer erreichen. Das ist übrigens eines der Hauptanwendungsgebiete der Mikrorechentechnik - denken wir nur an die Industrieroboter, numerisch gesteuerte Werkzeugmaschinen, aber auch an die moderne Militärtechnik. Überall werden Antriebe durch Mikrorechner gesteuert. Wir werden einige Prinzipien im Experiment kennenlernen - wenn auch in bescheidenem Maße.
Unser Antrieb wird ein Spielzeugmotor für 3 ... 5 V und 1700 ... 2800 Umdrehungen pro Minute sein - etwa der Typ 3041 von PIKO, der mit Getriebe für ein paar Mark in allen Geschäften für Bastlerbedarf zu haben ist. Als Spannungsquelle kommen Batterien (+4,5 V) oder ein separates Netzteil mit 5 V Ausgangsgleichspannung in Frage.
Unsere vergleichsweise bescheidenen Leistungen lassen noch eine Steuerung mit Transistoren zu, die "größeren Brüder" der Industrie verlangen Thyristoren oder komplette Leistungsanlagen wie Stromrichter usw.
Wir wenden uns im folgenden zunächst der Steuerschaltung zu.

3.1. Eine universelle Steuerschaltung

Das Kernstück der Motoransteuerung ist eine Brückenschaltung, gebildet aus den Leistungstransistoren T 1 ... T 4. Diese beiden Komplementärstufen werden von CMOS-Treibern V 4050 D und diese wiederum vom PIO-Port B angesteuert. Dabei wird durch die gewählte Eingangsverkopplung erreicht, daß pro Komplementärstufe

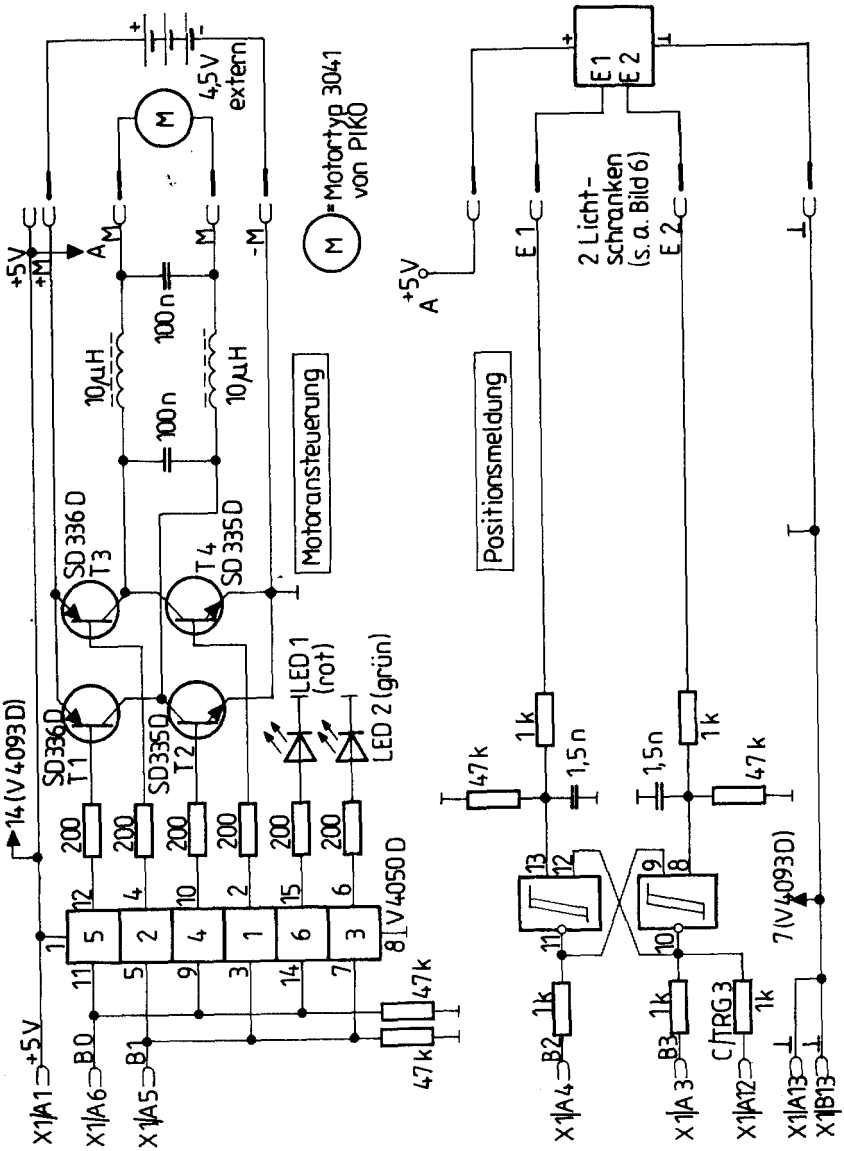


Bild 4: Motoransteuerung mit Positionsrückmeldung

maximal ein Transistor leitend ist. Bei gegensätzlicher Ansteuerung von B 0 und B 1 ergibt sich eine Spannungsdifferenz am Motor und er dreht sich. Die logischen Signale an B 0 und B 1 können durch die beiden LEDs optisch dargestellt werden. Zusätzlich zur Motoransteuerung befindet sich noch ein CMOS-Trigger V 4093 D auf der Leiterplatte, den wir später noch brauchen werden.

Beim Aufbau der Leiterplatte ist unbedingt darauf zu achten, daß der Minuspol der Motorstromversorgung mit Masse des LC-80 (Steckkontakte X 1/A 13 und X 1/B 13) verbunden wird.
Bild 4 zeigt die Gesamtschaltung der Motorsteuerung.

3.2. Er dreht sich!

Wir stecken die Steuerschaltung bei abgeschaltetem LC-80 in den USER-Bus, schalten dann den Rechner ein und schließen zuletzt die externe Spannungsquelle +4,5 ... +5 V an. Beim Abschalten gehen wir umgekehrt vor, jetzt muß die externe Spannungsquelle zuerst abgeklemmt werden, dann wird der LC-80 ausgeschaltet und zuletzt die Leiterplatte abgezogen.
Das folgende kurze Programm läßt den Motor mit voller Kraft anlaufen:

```
2000   3E FF        LD A, FF     } PIO-Port B, Mode 3
2002   D3 FB        OUT FB       }
2004   3E FC        LD A, FC     } E/A-Definition: B 0, B 1 =
2006   D3 FB        OUT FB       } Ausgänge, B 2,B 3 = Eingänge
2008   3E 01        LD A, 01     } Ausgabe von "1" an B 0
200A   D3 F9        OUT F9       } Ausgabe von "0" an B 1
200C   76           HALT
```

Wird auf 2009 statt 01 nun 02 eingegeben, ändert sich die Drehrichtung, bei 00 oder 03 steht der Motor, weil beide Endstufen in gleicher Richtung durchgeschaltet werden.
Das war noch recht einfach, aber immerhin können wir Links- und Rechtslauf sowie Stillstand des Motors durch Programmierung schalten.
Richtig interessant wird es aber erst, wenn alle Funktionen mit definierten Tasten aufgerufen werden können. Das folgende Programm realisiert das und zwar wollen wir:

bei [+] Vorwärtslauf,
bei [-] Rückwärtslauf und
bei [0] Stop des Motors erreichen.

```
2000   3E FF          LD A, FF      } PIO-Port B (Mode 3)
2002   D3 FB          OUT FB        }
2004   3E FC          LD A, FC      } E/A-Definition: B 0, B 1 =
2006   D3 FB          OUT FB        } Ausgänge, B 2, B 3 = Eing.
2008   CD 83 04 * M1: CALL DAK 2
200B   FE 0A          CP 0A         [+] ?
200D   28 0A          JRZ 0A        M2
200F   FE 17          CP 17         [-] ?
2011   28 0C          JRZ 0C        M3
2013   FE 04          CP 04         [0] ?
2015   28 0E          JRZ 0E        M4
2017   18 EF          JR EF         M1
2019   3E 01       M2:LD A, 01      } Ausgabe: B 0 = 1 } vorwärts
201B   D3 F9          OUT F9        }          B 1 = 0 }
201D   18 E9          JR E9         M1
201F   3E 02       M3:LD A, 02      } Ausgabe: B 0 = 0 } rückwärts
2021   D3 F9          OUT F9        }          B 1 = 1 }
2023   18 E3          JR E3         M1
2025   3E 00       M4:LD A, 00      } Ausgabe: B 0 = 0 } Halt des
2027   D3 F9          OUT F9        }          B 1 = 0 } Motors
2029   18 DD          JR DD         M1
```

Beim Starten des Programms tut sich zunächst nichts, erst das Drücken der [+] - oder [-] -Taste veranlaßt den Anlauf des Motors in der gewünschten Richtung, [0] stoppt ihn sofort.
Wir bemerken, daß der durch Tastendruck gewählte Zustand des Motors bis zum nächsten Druck einer anderen Taste erhalten bleibt. Diese "Selbsthalteschaltung" ist für manche Anwendung ungünstig, oft wird verlangt, daß der Motor nur solange läuft, wie die entsprechende Taste gedrückt ist. Das ist bei geforderter Dosierung (z. B. Kranmotoren) erwünscht. Auch dafür ein Beispiel:

```
2000   3E FF          LD A, FF      } PIO-Port B (Mode 3)
2002   D3 FB          OUT FB        }
2004   3E FC          LD A, FC      } E/A-Definition: B 0, B 1 =
2006   D3 FB          OUT FB        } Ausgänge, B 2, B 3 = Eing.
2008   3E 00      M1: LD A, 00      } Motor aus
200A   D3 F9          OUT F9        }
2000   CD 83 04 *     CALL DAK 2
200F   FE 0A          CP 0A           [ + ] ?
2011   28 06          JRZ 06          M2
2013   FE 17          CP 17           [ - ] ?
2015   28 0F          JRZ 0F          M3
2017   18 EF          JR EF           M1
2019   3E 01      M2: LD A, 01      } Motor rechts ein
201B   D3 F9          OUT F9        }
201D   06 10          LD B, 10      } Zeitschleife für
201F   CD 83 04 *     CALL DAK 2    } Rechtslauf
2022   10 FB          DJNZ FB       }
2024   18 E2          JR E2           M1
2026   3E 02      M3: LD A, 02      } Motor links ein
2028   D3 F9          OUT F9        }
202A   06 10          LD B, 10      } Zeitschleife für
2020   CD 83 04 *     CALL DAK 2    } Linkslauf
202F   10 FB          DJNZ FB       }
2031   18 D5          JR D5           M1
```

Grundsätzlich steht der Motor jetzt, nur solange eine Taste ([+] oder [-]) gedrückt wird, läuft er in der entsprechenden Richtung. Die beiden Zeitschleifen sind notwendig, da der Motor bei jedem Programmdurchlauf ab M 1 zyklisch ausgeschaltet wird. Dieser Zustand wird durch DAK 2 für 10 ms erhalten. Die Zeitschleifen bewirken nun, daß der Einschaltzustand 16 x solange gehalten wird. Damit läuft der Motor nahezu mit voller Kraft. Wenn wir probeweise auf den Adressen 201E und 202B den Wert 01 eingeben, läuft der Motor sehr langsam.
Das bringt uns auf eine tolle Idee!!!
Wir können Motoren nicht nur ein- und ausschalten, sondern auch die Drehzahl wählen - nämlich über das Tastverhältnis von Ein- und Ausschaltdauer, wenn alles sehr schnell geht. Das wird gleich ausprobiert, indem wir auf den Adressen 201E und 202B mit den Werten von 01 bis 20 verschiedene Testverhältnisse realisieren. Es ist sogar möglich, für Rechte- und Linkslauf unterschiedliche Drehzahlen einzustellen. Das könnte für ein Automodell interessant werden, wo im "Rückwärtsgang" geringere Geschwindigkeiten erwünscht sind.
Erstaunlich, welche Möglichkeiten sich da bieten. Aber es geht noch weiter.
Oft steht das Problem, einen Motor automatisch "weich" anlaufen zu lassen - nichts ist wirklichkeitsfremder als ein in weniger als 1 Sekunde auf Höchstgeschwindigkeit beschleunigter Zug auf der Modellbahnanlage. Aber genau das ist bei vielen Anlagen, die im Automatikbetrieb laufen, der Fall. Die Schaltglieder (z. B. Signalrelais) lassen eben nur Fahren oder Stehen des Zuges zu. Das nachfolgende Programm löst diese Aufgabe.
Im Startmoment steht der Motor. Beim Drücken der [+]-Taste (die hier als "Gaspedal" benutzt wird) beginnt er, langsam anzulaufen. Wird die Taste losgelassen, läuft er mit der gerade erreichten Drehzahl weiter. Die [-]-Taste (Bremse) bewirkt ein "gefühlvolles" Abtouren, auch hier wird beim Loslassen die erreichte Geschwindigkeit beibehalten. Genau das

Richtige für alle Fahrmodelle.
Eine Bemerkung noch für die Modellbahnfreunde.
Unsere Experimente führen wir mit einem 4,5 V-Motor durch. Diese Betriebsspannung reicht für die Modelleisenbahn nicht aus. Eine einfache Erhöhung der extern zugeführten Motorspannung ist nicht möglich, da die Schaltpegel der CMOS-Bausteine nur 0 V bzw. +5 V erreichen. Ein Ausweg wäre die gleichzeitige Versorgung der CMOS-Treiber mit der Fahrspannung von +12 V, dann reicht aber der Signalpegel unserer PIO nicht mehr. In jedem Fall ist dann zwischen PIO und V 4050 D eine Pegelwandlung vorzunehmen.
Hier nun das Programm:

```
2000   3E FF          LD A, FF      PIO-Port B (Mode 3)
2002   D3 FB          OUT FB
2004   3E FC          LD A, FC      E/A-Definition: B 0, B 1 =
2006   D3 FB          OUT FB        Ausg., B 2, B 3 = Eing.
2008   16 01          LD D, 01      Grundzustand des Motors (V in D)
200A   0E 10          LD C, 10      Beschleunigungs- bzw. Bremswert
200C   7A         M1: LD A, D       Zeit für vorwärts in D, dann
200D   ED 44          NEG A         in A, dort negieren - die
                                     Differenz zu FF ist die
200F   5F              LD E, A       Haltzeit, in E laden
2010   42              LD B, D       Vorbereitung Zeitschleife vorw.
2011   3E 01       M2: LD A, 01      Ausgabe Vorwärtslauf
2013   D3 F9          OUT F9
2015   10 FA          DJNZ FA       M2  Zeitschleife vorwärts
2017   43              LD B, E       Vorbereitung Zeitschleife Halt
2018   3E 00       M3: LD A, 00      Ausgabe Halt
201A   D3 F9          OUT F9
201C   10 FA          DJNZ FA       M3  Zeitschleife Halt
201E   0D              DEC C
201F   20 EB          JRNZ EB       Zeitschleife für V-Änderung
                                     M1
2021   3E 00          LD A, 00      Ausgabe Halt
2023   D3 F9          OUT F9
2025   CD 83 04       CALL DAK 2
```

```
2028   FE 0A          CP 0A          [+] ?
202A   28 06          JRZ 06         M4
202C   FE 17          CP 17          [-] ?
202E   28 0A          JRZ 0A         M5
2030   18 D8          JR D8          M1
2032   7A         M4: LD A, D
2033   FE FE          CP FE          Maximaldrehzahl erreicht?
2035   28 D3          JRZ D3         M1
2037   14             INC D          Drehzahl erhöhen!
2038   18 D0          JR D0          M1
203A   7A         M5: LD A, D
203B   FE 01          CP 01          Minimaldrehzahl erreicht?
203D   28 CB          JRZ CB         M1
203F   15             DEC D          Drehzahl erniedrigen!
2040   18 C8          JR C8          M1
```

Ein schnelleres Anfahren und Abbremsen erreichen wir durch Änderung der Information auf 200B (z. B. 05).
Soll der Motor in der anderen Richtung laufen, wird auf 2012 statt 01 die Information 02 eingetragen.

Und noch eine erstaunliche Variante.
Wird in das ursprüngliche Programm auf Adresse 2019 statt 00 die neue Information 02 eingetragen, ist ein stufenloser Betrieb von Linkslauf über Halt bis Rechtslauf und umgekehrt möglich.

In den gezeigten Programmvarianten wird mittels Veränderung des Tastverhältnisses die Drehzahl eines Kleinmotors eingestellt.
Im letzten Beispiel wird der Motor nicht ein- und ausgeschaltet, sondern ständig umgepolt. Je nach Zeitverhältnis beider "Halbwellen" ist jede Drehzahl und Drehrichtung stufenlos einstellbar.
Mit diesem Prinzip können Antriebe für Demonstrationsmodelle in beliebiger Weise an das gestellte Problem angepaßt werden. Dabei werden oft komplizierte Getriebe entbehrlich.

Außer den gezeigten Beispielen sind weitere Anwendungsmöglichkeiten gegeben, wie z. B.:

- Ansteuerung eines Motors mit Drehzahl und Drehrichtung je nach gedrückter Taste (z. B. [0] = Stand; [1] = sehr langsam vorwärts,... [F] = Maximaldrehzahl vorwärts)
- Ansteuerung mehrerer Motoren
- Benutzung des PIO-Ports A zur Abfrage externer Bedienungsorgane (z. B. Steuerknüppel oder Zusatztastatur)
- Zeitablaufsteuerung

Im folgenden Abschnitt werden wir nicht nur Aufträge an unseren Motor geben, sondern auch Informationen über Zustände des Antriebs einholen. Mittels einfacher Lichtschranken ist es möglich, Daten über Drehzahl des Motors oder Positionsmeldungen des Antriebssystems (z. B. für ein Aufzugsmodell) in den Rechner einzugeben, dort auszuwerten und dann Kommandos an den Motor zu geben.
Das ist der Grund, weshalb wir den Schaltkreis V 4093 D auf der Leiterplatte haben, er dient als Signalaufbereitung für die Lichtschranken.

3.3. Positionsmeldung

Es ist immer wieder faszinierend, Industrieroboter bei der Arbeit zu beobachten. Mit erstaunlicher Präzision werden Werkstücke von den festgelegten Stellen abgeholt, in bestimmte Positionen gebracht und bearbeitet. Wie funktioniert so etwas und haben wir eine Chance, Bewegungen ähnlich präzise mit dem LC-80 zu steuern?

Zur ersten Frage:
Das Prinzip derartiger Steuerungsabläufe ist die Zerlegung einer Bewegung in winzig kleine Schritte - die dazu erforderlichen Motoren heißen demzufolge Schrittmotoren. Diese sind in der Lage, impulsgesteuert Drehwinkelschritte von wenigen Grad auszuführen. Entsprechende Getriebe sorgen für die entsprechende Untersetzung und damit die Präzision des Antriebes.
Zur zweiten Frage:
Der LC-80 kann so etwas, er ist ja ein vollwertiger Mikrorechner. Nur unser Motor und seine Mechanik sind nicht vergleichbar mit Schrittmotoren - dafür aber wesentlich billiger. Für die Demonstration von Prinzipien der Bewegungssteuerung und den Betrieb einfacher mechanischer Modelle wird es trotzdem reichen.
Die wichtigste Voraussetzung für unser Vorhaben ist eine Positionsrückmeldung. Dabei beschränken wir uns auf die Meldung jeder vollen Umdrehung der langsamen Getriebeachse. Dort bauen wir zwei Lichtschranken auf, die über eine Kreisscheibe mit einem Loch das aus zwei Gattern eines V 4093 D gebildete Flip-Flop setzen und rücksetzen können.
Dieser etwas aufwendigere Weg mit zwei Lichtschranken wurde gewählt, um allen denkbaren Prell- und Flankenproblemen aus dem Wege zu gehen.
Das mechanische Prinzip zeigt Bild 5.

Elektrisch wird die Schaltung gemäß Bild 6 aufgebaut und an die Motorsteuerung von Seite 27 angeschlossen.

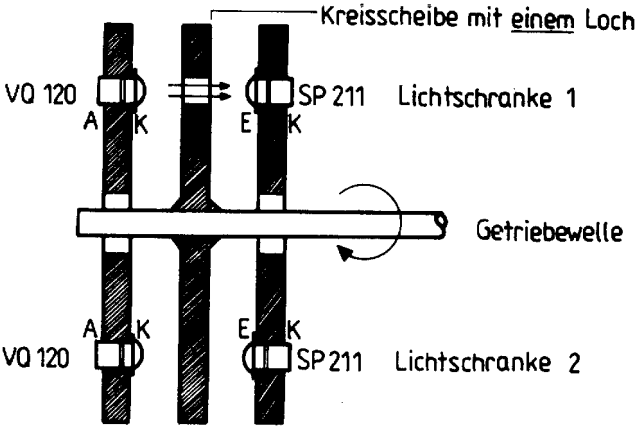


Bild 5: Mechanik der Lichtschranken

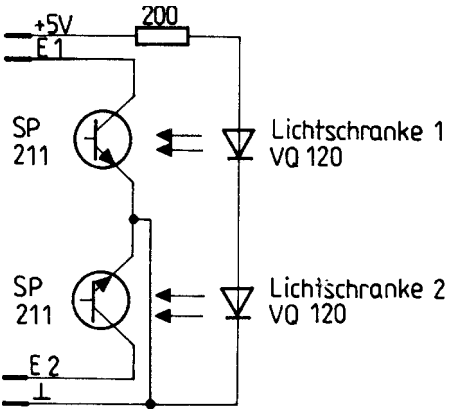


Bild 6: Schaltung der Lichtschranken

Zur Erprobung lassen wir den Motor laufen und prüfen am Ausgang (10) des Flip-Flops, ob ein sauberes Umschalten des Pegels mit etwa gleichem Tastverhältnis im Takt der Wellendrehzahl erfolgt. Der Flip-Flop-Ausgang 10 liegt auf der Leitung B 3 des USER-Bus, der negierte Ausgang 11 auf B 2. Für die folgende Programme sind B 2 und B 3 als Eingänge programmiert und der Zustand von B 2 wird ausgewertet.
Für ein erstes Experiment mit der Rückmeldeschaltung erzeugen wir mittels Interruptprogramm einen kurzen Piepton bei jeder Umdrehung.
Die gesamte PIO-Initialisierung schreiben wir als Unterprogramm ab Adresse 2100, da wir es für später folgende Experimente immer wieder benötigen werden.
Das Hauptprogramm, hier nur sehr kurz, beginnt auf der Adresse 2000, das Interruptprogramm auf 2200.

3.3.1. Hauptprogramm

```
2000   CD 00 21      CALL INI       PI0-Initialisierung
2003   3E 01         LD A, 01       Motor vorwärts an
2005   D3 F9         OUT F9
2007   18 FE         JR FE          dynamischer Halt
```

3.3.2. Unterprogramm INI

```
2100   F3            DI             Interrupt gesperrt
2101   ED 5E         IM 2           Interrupt-Mode 2
2103   3E FF         LD A, FF       PIO-Port B (Mode 3)
2105   D3 FB         OUT FB
2107   3E FC         LD A, FC       E/A-Definition: B 0, B 1 =
2109   D3 FB         OUT FB         Ausg., B 2, B 3 = Eing.
```

```
210B   3E 21        LD A, 21      } Interrupt-Vektor, High-Teil
210D   ED 47        LD I, A       } in I-Register
210F   3E 50        LD A, 50      } Interrupt-Vektor, Low-Teil
2111   D3 FB        OUT FB        } für PIO-Port B
2113   3E 97        LD A, 97      } Interrupt-Steuerwort
2115   D3 FB        OUT FB 1      }
2117   3E FB        LD A, FB      } Maske (Interrupt nur erlaubt
2119   D3 FB        OUT FB        } für Eingang B 2)
211B   FB           EI              Interrupt-Freigabe
211C   C9           RET
```

3.3.3. Interrupt-Startadresse

```
2150   00                         } Startadresse des Interrupt-
2151   22                         } programmes auf 2200
```

3.3.4. Interrupt-Programm

```
2200   CD 5F 03     CALL TON
2203   FB           EI            Wiederfreigabe nach Interrupt
2204   ED 4D        RETI
```

Wenn beim Programmstart der Motor läuft und es bei jeder Umdrehung der langsamen Getriebewelle piept, ist alles in Ordnung.

Na gut - aber was hat das mit einem Schrittmotor zu tun, unserem eigentlichen Ziel?
Eine ganze Menge, wie wir gleich sehen werden. Wenn wir nämlich das Hauptprogramm und das Interruptprogramm entsprechend verändern, wird die Rückmeldung nicht zur Er-

zeugung eines Tones, sondern zum Stoppen des Motors genutzt. Wir gestalten das Programm so, daß ein Tastendruck auf [+] eine Umdrehung vorwärts, beim Betätigen von [-] eine Umdrehung rückwärts ausgeführt wird.
Das Unterprogramm INI und die Interrupt-Startadresse bleiben unverändert wie beim vorigen Experiment. Sind beide nicht mehr vorhanden, müssen wir die Programmteile auf den Adressen 2150 und 2200 wieder eintragen.

3.3.5. Hauptprogramm "Schrittmotor"

```
2000   CD 00 21       CALL INI
2003   CD 5A 04 * H3: CALL DAK 1
2006   FE 10          CP,10          [+] ?
2008   28 06          JRZ 06         H1
200A   FE 11          CP 11          [-] ?
200C   28 08          JRZ 08         H2
200E   20 F3          JRNZ F3        H3
2010   3E 01      H1: LD A, 01       } Motor vorwärts an
2012   D3 F9          OUT F9         }
2014   18 ED          JR ED          H3
2016   3E 02      H2: LD A, 02       } Motor rückwärts an
2018   D3 F9          OUT F9         }
201A   18 E7          JR E7          H3
```

3.3.6. Interruptprogramm "Schrittmotor"

```
2200   3E 00          LD A, 00       } Motor Halt
2202   D3 F9          OUT F9         }
2204   FB             EI
2205   ED 4D          RETI
```

So ein Steuerprogramm kann schon recht nützlich sein, z. B. zum Positionieren eines Aufzugantriebes oder zur Winkeleinstellung einer Antenne.
Alle Anwendungen dieser Art setzen natürlich ein nachgeschaltetes Untersetzungsgetriebe voraus. Als Linearantrieb für Modelle (Kranbahn, Aufzug usw.) eignet sich hervorragend ein langer Gewindestift mit ca. 300 Gewindegängen, der mit einer flexiblen Kupplung (Spiralfeder, Gummischlauch) von unserer langsamen Getriebewelle bewegt wird. Darauf läuft eine gegen Verdrehen gesicherte Mutter als Mitnehmer für das zu bewegende Objekt hin und her, wenn sich der Gewindestift dreht. Als erstes Anwendungsbeispiel für solch einen Linearantrieb wollen wir ein Positioniersystem, wie es bsw. für Plotter oder Modellobjekte verwendet werden kann, bauen und programmieren.
Wir wollen erreichen, daß über die Tastatur ein Koordinatenwert eingegeben werden kann, der dann durch unseren Antrieb exakt eingestellt wird. Dabei bedeutet der Wert 01, daß die Spindel eine Umdrehung ausführt - FF entspricht dann 255 Umdrehungen. Wir können jede Position zwischen 00 und PF vorgeben, der Antrieb wird es ausführen.
Zuerst geben wir das folgende Programm ein.

Das Programmsystem "Linearantrieb" besteht aua folgenden Teilen:

- Hauptprogramm "Linearantrieb" (Adresse 2000)
- Unterprogramm INI (Adresse 2100)
- Unterprogramm ANZVOR (Adresse 2120)
- Interruptprogramm "Linearantrieb" (Adresse 2200)

Die zur Abspeicherung von Daten benutzten Adressen zeigt folgende Aufstellung:

2300 Zielposition in Hexa-Darstellung
2301 augenblickliche Position des Antriebes in Hexa-Darstellung

```
2302 }                                          / rechtes
2303 } Zielposition in 7-Segment-Form           \ Display
2304   dunkel                                     mittleres
2305   dunkel                                     Display
2306 }                                          / linkes
2307 } Ist-Position in 7-Segment-Form           \ Display
2308   Differenz als Betrag zwischen Ziel- und Ist-
       position (Hex)
2309   Richtungsspeicher (vorwärts 01, rückwärts 02)
```

3.3.7. Hauptprogramm "Linearantrieb"

```
2000   21 00  00      LD HL, 0000     } Ist- und Zielspeicher
                                      } auf 00 setzen, mitt-
2003   22 00 23       LD (2300),HL    } leres Display löschen
2006   22 04 23       LD (2304),HL    }
2009   CD 00 21       CALL INI        PIO initialisieren
200C   3E 00       H6: LD A, 00        } Motor aus
200E   D3 F9          OUT F9          }
2010   CD 20 21    H2: CALL ANZVOR     } Anzeige von Ist- und
2013   CD 5A 04*      CALL DAK 1       } Zielposition
2016   FE 12          CP 12            EX ?
2018   28 07          JRZ 07           H1
201A   21 00 23       LD HL, 2300
201D   ED 6F          RLD              Linksverschiebung der Ziffer
201F   18 EF          JR EF            H2
2021   21 01 23    H1: LD HL, 2301     Ist-Position
2024   3A 00 23       LD A, (2300)     Zielposition
2027   BE             CPM              Vergleich Ist-/Zielpos.
2028   28 E6          JRZ E6           wenn gleich    H2
202A   96             SUB M            Ziel minus Ist-Position
202B   38 09          JRC 09           wenn negativ   H3
202D   32 08 23       LD (2308), A     positive Differenz in 2308
2030   3E 01          LD A, 01         } Motor vorwärts an
2032   D3 F9          OUT F9           }
```

```
2034   18 09          JR 09          H4
2036   ED 44      H3: NEG            aus negativer Differenz
2038   32 08 23       LD (2308), A   Betrag bilden und in 2308 laden
203B   3E 02          LD A, 02       Motor rückwärts an
203D   D3 F9          OUT F9
203F   32 09 23   H4: LD (2309), A   Richtung (01, 02) in 2309
2042   3A 08 23   H5: LD A, (2308)   Differenzbetrag in A
2045   FE 00          CP 00          Differenz mit 00 vergleichen
2047   28 C3          JRZ C3         wenn 00, Motor aus  H6
2049   CD 20  21      CALL ANZVOR    Anzeige des augenblicklichen
204C   CD 83 04*      CALL DAK 2     Standes (Ist- und Zielposition)
204F   18 F1          JR F1          H5
```

3.3.8. Unterprogramm ANZVOR (Anzeigevorbereitung)

```
2120   21 02 23       LD HL, 2302    Beginn Anzeigebereich Ziel
2123   3A 00 23       LD A, (2300)   Zielposition
2126   CD D9 04*      CALL TWOSEG    Zielposition in 7-Segment
                                     umformen
2129   21 06 23       LD HL, 2306    Anzeigebereich für Ist-
                                     Position
2120   3A 01 23       LD A, (2301)   Ist-Position
212F   CD D9 04 *     CALL TWOSEG    Ist-Position in 7-Segment
                                     umformen
2132   DD 21 02 23    LD IX, 2302    IX für DAK 1/2 vorbereiten
2136   C9             RET
```

3.3.9. Interruptprogramm "Linearantrieb"

```
2200   E5             PUSH HL
2201   21 08 23       LD HL, 2308    Differenzbetrag um 1
2204   35             DEC M          erniedrigen
```

```
2205   23             INC HL       } Richtungsspeicher auf Bit 0
2206   CB 46          BIT 0, M     } prüfen (b 0 negativ → F)
2208   21 01 23       LD HL, 2301    Ist-Position aufrufen
220B   28 03          JRZ 03         wenn Bit 0 = 0 → I1
220D   34             INC M          Ist-Position um 1 erhöhen
220E   18 01          JR 01          I2
2210   35          I1:DEC M          Ist-Position um 1 erniedrigen
2211   E1          I2:POP HL
2212   FB             EI             Interrupt wieder frei
2213   ED 4D          RETI

2150   00                          } Startadresse des Interrupt-
2151   22                          } programmes auf 2200
```

Damit das Programmsystem "Linearantrieb" funktioniert, muß auf Adresse 2100 das Unterprogramm INI stehen (von den vorherigen Experimenten) oder neu eingetragen werden.
Das auf Adresse 2000 stehende Programm wird mit [EX] gestartet, auf dem Display erscheint:

[0 0 0 0]

Mit den Zifferntasten [0] bis [F] kann rechts der Zielwert eingegeben werden. Vorsichtshalber tragen wir dort erst einmal 01 ein und betätigen wieder [EX]. Der Antrieb müßte jetzt eine Umdrehung ausführen und danach anhalten. Wenn die Antriebsmutter am Ende (00) der Spindel war und die eben ausgeführte Bewegung in die richtige Richtung ging, können jetzt auch höhere Positionen (z. B. 20) eingegeben werden, ansonsten wird erst der Motor umgepolt. Eine erneute Eingabe kann immer beim Stillstand des Motors erfolgen, [RES] zerstört alle gespeicherten Informationen und sollte deshalb nur bei "Katastrophen" betätigt werden. Am Ende der Experimente sollte man den Antrieb immer in Position 00 fahren, das erleichert den Neubeginn bei nachfolgenden Versuchen.

Ein typischer Anwendungsfall für unser Antriebssystem ist ein Aufzugsmodell. Nur - es ist nicht sehr vorbildgetreu, wenn die "Etagen" nur etwa 0,5 mm Abstand haben. Das ist nämlich die Gewindesteigung unserer M 4-Spindel. Es müßte also einen Trick geben, mit dem wir die Anzahl der Umdrehungen pro "Stockwerk" festlegen und damit einen echten Aufzugsbetrieb ermöglichen können. Den Trick gibt es - der fast vergessene CTC-Schaltkreis ist für diese Aufgabenstellung wie geschaffen.

3.4. Ein vergessener Schaltkreis hilft

Den CTC-Schaltkreis U 857 D auf unserem LC-80 haben wir bisher noch nicht verwendet. Wir erinnern uns daran, daß dies ein Zähler-/Zeitgeber-Baustein ist, den wir jetzt als Zähler benutzen wollen.
Die Hardware für die gestellte Aufgabe beschränkt sich auf einen Widerstand. Der Flip-Flop-Ausgang 10 ist über 1 kOhm auch an den Triggereingang C/TRG 3 des CTC-Schaltkreises U 857 D (Steckverbinderanschluß X 1/A 12) angeschlossen (siehe auch Bild 4).
Ähnlich wie bei der PIO erfolgt auch beim CTC eine Initialisierung. Im Rahmen dieser Initialisierung erfolgt eine Festlegung des Zählumfanges. Dieser Wert ist zwischen 00 und PF frei wählbar. Jeder Eingangsimpuls führt zu einer Erniedrigung des Zählers. Wenn 00 erreicht wird, erfolgt eine Interruptauslösung für den entsprechenden Kanal. Diesen Interrupt lassen wir in unserem Beispiel genauso wirken wie im Programmsystem "Linearantrieb".
In Anlehnung an das Programmierbeispiel auf S. 164 des LC-80-Handbuches wählen wir den Kanal 3 zum Zählen der Impulse aus. Da wir außer der PIO nun noch den CTC initialisieren müssen, ist ein neues Initialisierungsprogramm INICTC erforderlich.

3.4.1. Unterprogramm INICTC

```
2100   F3          DI          Interrupt gesperrt
2101   ED 5E       IM 2        Interrupt-Mode 2
2103   3E FF       LD A, FF    } PIO-Port B (Mode 3)
2105   D3 FB       OUT FB      }
2107   3E FC       LD A, FC    } E/A-Definition: B 0, B 1 =
2109   D3 FB       OUT FB      } Ausgänge, B 2,B 3 = Eingänge
210B   3E 21       LD A, 21    Interrupt-Vektor (High-Teil)
210D   ED 47       LD I, A     in das Interrupt-Register
210F   3E 50       LD A, 50    } Interrupt-Vektor (Low-Teil)
2111   D3 EC       OUT EC      } in CTC-Kanal 0 (Adresse im
                               } LC-80: EC), Interrupt-Vektor
                               } 56 für Kanal 3
2113   3E D5       LD A, D5    } Kanalsteuerwort für CTC-
2115   D3 EF       OUT EF      } Kanal 3
2117   3E 01       LD A, 01    } Zählervoreinstellung für Kanal 3
2119   D3 EF       OUT EF      } (hier Wert 01, d. h. Interrupt
                               } nach jeder Umdrehung)
211B   FB          EI          Interrupt freigegeben
211C   C9          RET
```

1) Wegen einer besonderen Programmiervorschrift des CTC muß der Interruptvektor immer für den Kanal 0 programmiert werden, in unserem Beispiel hat der Kanal 0 die Adresse EC und der Low-Teil des Interrupt-Vektors den Wert 50.
   Die anderen Kanäle ergeben sich dann so:

Kanal 0:	50
	51
Kanal 1:	52
	53
Kanal 2:	54
	55
Kanal 3:	56
	57

   Deshalb müssen wir auch die Interrupt-Startadresse neu festlegen:

3.4.2. Interrupt-Startadresse

2156	00	Startadresse des Interrupt-
2157	22	programmes auf 2200

Das war's schon, mehr brauchen wir gegenüber dem Programm "Linearantrieb" nicht zu ändern.
Folgende Programmteile werden unverändert übernommen:

- Hauptprogramm "Linearantrieb"
- Unterprogramm ANZVOR
- Interruptprogramm "Linearantrieb".

Wenn alles stimmt, nehmen wir den "Aufzug" in Betrieb - genauso wie beim Programm Linearantrieb beschrieben. Es-funktioniert auch genauso! Warum das?
Weil wir auf Adresse 2118 den Wert 01 eingetragen haben, d. h. der CTC-Kanal ist auf den Wert 01 voreingestellt - damit reicht ein Impuls (eine Umdrehung) aus, um den Wert 00 und so ein Interrupt zu erreichen.
Versuchen wir es einmal mit 02 auf der Adresse 2118, nachdem wir den Antrieb vorher wieder auf 00 gefahren haben. Wird jetzt z. B. 10 eingetippt, fährt der Antrieb auf die ehemalige Marke 20, also doppelt so weit. Ganz einfach, er multipliziert unseren Eingabewert mit 2, indem er nur bei jeder 2. Umdrehung einen Interrupt auslöst.
Damit sind wir nun in der Lage, jede beliebige "Etagenhöhe" vorzuprogrammieren. Wird z. B. die Etagenhöhe von 20 (Hexazahl = 32 Umdrehungen) auf Adresse 2118 eingetragen, erscheint bei Eingabe von 01 als "Zieletage" die 01 nach 32 Umdrehungen im linken Display - die erste Etage wurde erreicht. Genauso geht es mit den übrigen Stockwerken. Bei unseren Eingaben müssen wir die begrenzte Länge der Spindel beachten, wesentlich mehr als 256 Umdrehungen haben wir nicht zur Verfügung. Es ist also wichtig, überschlägig die Zieletage mit der Anzahl

der Umdrehungen pro Etage zu multiplizieren, sonst gibt es einen "Unfall".
Natürlich kann das auch softwaremäßig verhindert werden, ein Weg dazu ist die Begrenzung der Eingabe auf einstellige (Hexa-) Ziffern. Dazu schreiben wir beispielsweise auf Adresse 201D den Wert 77 und auf Adresse 201E den Wert 00. Damit begrenzen wir die Anzahl der Stockwerke auf dezimal 16. Die maximale Anzahl der Umdrehungen pro Stockwerk ist damit 0F, also ebenfalls 16...
Wen die hexadezimale Eingabe, Anzeige und Zählweise stört, der kann sein Programm mit dem DAA-Befehl auch für "Outsider" überschaubar machen.

noch ein Tip zum Schluß!
Es wird beim Basteln, Programmieren und Spielen öfter vorkommen, daß der Antrieb durch falsche Vorgaben "über das Ziel hinausschießt". Dann ist er nur durch RES zu stoppen. Wie bekommen wir ihn dann wieder in die Ausgangsstellung 00? Durch das folgende kurze Programm:

```
2270   CD 00 21       CALL INICTC
2273   3E 01 (02)     LD A, 01 (02)    }    Motorlauf vorwärts
2275   D3 F9          OUT F9           }    (rückwärts) an
2277   76             HALT
```

Ist der Antrieb in der Ausgangslage 00, wird mit RES gestoppt.

So, als Einführung in Techniken zur Motorsteuerung mit Mikrorechnern mag das genügen. Erweiterungen in allen Richtungen sind dem Bastler jederzeit möglich, hier einige Anregungen:

- Mehrmotorenansteuerung (Roboter- und Kranmodelle usw.)
- automatische Koordinatenerzeugung im Rechner selbst (Folgesteuerung, Fütterungsautomaten für mehrere Aquarien oder Käfige, Steuerung für mechanische Großanzeigen, Drehscheibensteuerung auf Modellbahnen, Koor-

natenantrieb für Hobbymaschinen u. v. a.)

- Antennendrehanlage mit automatischer "Zielzuweisung" durch Kopplung mit dem Sender-Tastwahl-Aggregat (auf galvanische Netztrennung achten!)
- Steuerung von Großmotoren (das Prinzip der Positionsrückmeldung ist nicht an die direkte Ansteuerung durch den LC-80 gebunden!)
- Weiterführung der Feinmechanik bis hin zum Plotter oder Zeichenautomaten oder gar Bohrautomaten für die Leiterkartenherstellung

4. Spielpause

Die vorangegangenen Kapitel mit ihren zum Teil erheblichen Hardware-Problemen geben uns das Recht, jetzt einmal zu spielen, ohne daß etwas gebaut werden muß.

4.1. Logikspiel (Master-Mind)

Dieses Spiel gibt es in den verschiedensten Varianten mit und ohne Computer (Superhirn, Master-Mind, Logiktrainer usw.). Immer geht es darum, eine vom gegnerischen Spieler (oder vom Computer) vorgegebene unbekannte Farb- oder Ziffernkombination zu ermitteln. In unserem Falle gibt der LC-80 eine vierstellige Zahl vor, gebildet aus den Ziffern 0 ... 7, die mit möglichst wenigen Versuchen zu ermitteln ist. Dabei sind beliebige Wiederholungen einzelner Ziffern möglich (z. B. 1137).
Das Spiel ist auf der Adresse 2000 zu starten und meldet sich dann mit einer Laufschrift "LOGIKSPIEL". Nach Drücken der [+] - Taste erscheint dann auf dem Display

[0 0 0 0.0 1] ,

das heißt, es handelt sich um den ersten Versuch. Durch Eingeben von Ziffern der eigenen Wahl (0 ... 7) können die ersten 4 Stellen des Displays beschrieben werden, wie wir es vom Adresseneingeben her gewohnt sind. Nach Betätigen der [+]-Taste wird uns das Ergebnis präsentiert, etwa so:

[1 P 2 F] .

Damit sagt der Computer, daß eine der 4 eingegebenen Ziffern im Wert und in ihrer Position (P) mit "seiner" Kombination übereinstimmt und außerdem zwei weitere Ziffern in "seiner" Zahl vorkommen, sich aber nicht auf den richtigen Positionen befinden (F). Beim "handbetriebenen" Master-Mind wären

P = schwarze Stifte,

F = weiße Stifte,

und die Ziffern entsprächen den verschiedenen Farbstiften. Wieder durch Betätigen von [+] wird der zweite Versuch vorbereitet, wobei sich zur Kontrolle zunächst noch die vor-angegangene Kombination im Display befindet. Nach Eingabe der 2. Zahl erfolgt mit [+] eine erneute Bewertung usw. Es ist sinnvoll, die einzelnen Versuche und die jeweilige Reaktion des Computers zu notieren und mit Logik hinter seinen "Geheimcode" zu kommen.
Nun zum Programmsystem.

4.1.1. Hauptprogramm

```
2000   06 1F          LD B, 1F                    \
2002   21 00 23       LD HL, 2300                  | Speicher-
2005   3E 00          LD A, 00                     | bereiche
2007   77          M1: LD M, A                      | löschen
2008   23              INC HL                       |
2009   10 FC          DJNZ FC          M1          /
200B   21 08 23       LD HL, 2308                  \ 2 Zufalls-
200E   CD A6 20       CALL ZUFAZ                   / ziffern eintr.
2011   0E 0F       M2: LD C, 0F                     \
2013   DD 21 A0 20     LD IX, 20A0                  |
2017   06 10       M3: LD B, 10                     |
2019   CD 83 04*   M4: CALL DAK 2                  | Laufschrift
201C   FE 0A          CP 0A            [+]         | LOGICSPIEL
201E   28 09          JRZ 09           M5          |
2020   10 F7          DJNZ F7          M4          /
```

```
2022   DD 2B          DEC IX                          
2024   0D             DEC C                           Laufschrift
2025   20 F0          JRNZ F0        M3               "LOGICSPIEL"
2027   18 E8          JR E8          M2               
2029   21 0A 23   M5: LD HL, 230A                     2 Zufalls-
202C   CD A6 20       CALL ZUFAZ                      ziffern eintr.
202F   3E 01          LD A, 01                        Versuchsnr. auf
2031   32 0C 23       LD (230C), A                    auf 01 setzen
2034   21 12 23   M6: LD HL, 2312                     
2037   DD 21 06 23    LD IX, 2306                     
203B   CD B4 20       CALL ANZ                        Darstellen des
203E   CD 5A 04 *     CALL DAK 1                      Speicher-
2041   FE 10          CP 10          [+] ?           bereiches
2043   28 05          JRZ 05         M7               Versuch und
2045   CD C8 20       CALL VERPRO                     Versuchsnummer
2048   18 EA          JR EA          M6               
204A   CD DD 20   M7: CALL VERGL                      
204D   3A 0C 23       LD A, (230C)                    aktuelle Ver-
2050   C6 01          ADD 01                          suchsnummer
2052   27             DAA                             aus 230C holen,
2053   32 0C 23       LD (230C), A                    um 1 erhöhen,
2056   E6 F0          AND F0                          Ergebnis BCD-
2058   0F             RRCA                            korr.
2059   0F             RRCA                            untere 4 bit
205A   0F             RRCA                            auf 230C,
205B   0F             RRCA                            obere 4 bit
205C   32 0D 23       LD (230D), A                    auf 230D ablegen
205E   3E 4F          LD A, 4F                        "P" in 2316
2061   32 16 23       LD (2316), A                    eintragen
2064   3E 4E          LD A, 4E                        "F" in 2312
2066   32 12 23       LD (2312), A                    eintragen
2069   3A 06 23       LD A, (2306)                    F-Zahl in
206C   CD CA 04 *     CALL ONESEG                     7-Segment-Form
206F   32 13 23       LD (2313), A                    in 2313 laden
2072   3A 07 23       LD A, (2307)                    P-Zahl in
2075   CD CA 04       CALL ONESEG                     7-Segment-Form
2078   32 17 23       LD (2317), A                    in 2317 laden
```

```
207B   FE 2B          CP 2B                  vergl. mit "4"
207D   28 0A          JRZ 0A          M8
207F   DD 21 12 23 M9:LD IX, 2312      }     Anzeigen des Ver-
2083   CD 5A 04 *     CALL DAK 1       }     gleichsergebnisses
2086   C3 34 20       JMP 2034
2089   FD 21 36 21 M8:LD IY, 2136      }     Noten für Tusch
208D   CD EE 04 *     CALL MUSIK       }
2090   18 ED          JR ED           M9
```

4.1.2. Text "LOGICSPIEL" als Laufschrift

```
2092   00
2093   00
2094   00
2095   00
2096   C2             "L"
2097   CE             "E"
2098   21             "I"
2099   4F             "P"
209A   AE             "S"
209B   C6             "C"
209C   21             "I"
209D   E6             "G"
209E   E7             "O"
209F   C2             "L"
20A0   00
20A1   00
20A2   00
20A3   00
20A4   00
20A5   00
```

4.1.3. Unterprogramm ZUFAZ

```
20A6   ED 5F          LD A, R          ; A mit 2 Hexa-
20A8   E6 77          AND 77           ; Ziffern aus dem
20AA   ED 6F          RLD              ; Refresh-Register
20AC   0F             RRCA             ; laden, Bit 3 und
20AD   0F             RRCA             ; 7 ausblenden und
20AE   0F             RRCA             ; so umformen, daß
20AF   0F             RRCA             ; je eine Ziffer
20B0   23             INC HL           ; (0 ... 7) in HL
20B1   ED 6F          RLD              ; und HL + 1 ab-
20B3   C9             RET              ; 7 gelegt wird
```

4.1.4. Unterprogramm ANZ

```
20B4   06 06          LD B, 06         ; 6maliges Umfor-
20B6   2B        M10: DEC HL           ; men der Tipzah-
20B7   DD 2B          DEC IX           ; len und der Ver-
20B9   7E             LD A, M          ; suchsnummer in
20BA   CD CA 04 *     CALL ONESEG      ; 7-Segment-Form,
20BE   DD 77 00       LD (IX + 00),A   ; Vorbereiten der
20C0   10 F4          DJNZ F4     M10  ; Anzeige
20C2   21 02 23       LD HL, 2302      ; Setzen des Punktes
20C5   CB E6          SET 4, M         ; in Digit 4
20C7   C9             RET
```

4.1.5. Unterprogramm VERPRO

```
20C8   06 04          LD B, 04         ; Verschieben von
20CA   21 0D 23       LD HL, 230D      ; 4 Tippziffern um
20CD   23        M11: INC HL           ; eine Stelle
```

```
20CE   E6 07          AND 07              letzte Ziffer
20D0   ED 67          RRD                 in T 4
20D2   CB 1E          RR M                T 4 → T 3
20D4   CB 1E          RR M                T 3 → T 2
20D6   CB 1E          RR M                T 2 → T 1
20D8   CB 1E          RR M                T 1 geht ver-
20DA   10 F1          DJNZ F1       M1    loren
20DC   C9             RET
```

4.1.6. Unterprogramm VERGL

```
20DD   CD 5F 03       CALL PIEP           Piepton
20E0   3E 00          LD A, 00
20E2   32 06 23       LD (2306), A        P und F
20E5   32 07 23       LD (2307), A        löschen
20E8   06 04          LD B, 04            Positionsvergl.
20ER   11 OD 23       LD DE, 230D              ↓
20ED   21 07 23       LD HL, 2307
20F0   13        M12: INC DE
20F1   23             INC HL              (Z 4)
20F2   1A             LD A, (DE)          (T 4)
20F3   BE             CPM                 T4 mit Z 4 vergl.
20F4   20 07          JRNZ 07       M13
20F6   3A 07 23       LD A, (2307)
20F9   30             INC A               P um 1
20FA   32 07 23       LD (2307), A        erhöhen
20FD   10 F1     M13: DJNZ F1       M12   nächstes Z und T
20FF   21 08 23       LD HL, 2308         Doppel der Zu-
2102   11 1A 23       LD DE, 231A         fallszahl Z 1 bis
2105   01 04 00       LD BC, 0004         Z 4 auf 231A bis
2108   ED B0          LDIR                231D ablegen
                                           (Z 1' bis Z 4')
210A   06 04          LD B, 04            Farbvergleich
210C   21 0E 23       LD HL, 230E              ↓
```

```
210F   11 1A 23  M14: LD DE, 231A
2112   0E 04         LD C, 04
2114   1A         M15: LD A, (DE)
2115   BE              CPM                    T 4 mit Z 4' vergl.
2116   20 0C           JRNZ 0C        M16
2118   3A 06 23        LD A, (2306)          ┐
211B   3C              INC A                 ├ F um 1
211C   32 06 23        LD (2306), A          ┘ erhöhen
211F   3E FF           LD A, FF              ┐ Z 4' mit FF
2121   12              LD (DE), A            ┘ überschreiben
2122   18 04           JR 04          M17
2124   13         M16: INC DE                ┐ T 4 mit Z 3'
2125   0D              DEC C                 ├ vergleichen
2126   20 EC           JRNZ EC        M15    ┘ usw.
2128   23         M17: INC HL                ┐ alles mit T 3
2129   10 E4           DJNZ E4        M14    ┘ wiederholen usw.
212B   21 07 23        LD HL, 2307            P in (HL)
212E   3A 06 23        LD A, (2306)           F in A laden
2131   96              SUB M                 ┐ P_neu = F - P
2132   32 06 23        LD (2306), A          ┘ in 2306 laden
2135   09              RET

2136   0B 04
2138   0F 04                                 ┐ Noten für
213A   12 04                                 ├ Tusch bei
2130   17 10                                 ┘ richtigem Tip
213E   80
```

4.1.7. Speicherorganisation

2305	2304	2303	2302	2301	2300
T 1	T 2	T 3	T 4	V(10)	V(1)

Anzeigebereich 1
Versuch und Versuchsnr.
im 7-Segment-Code

230B	230A	2309	2308	2307	2306
Z 1	Z 2	Z 3	Z 4	P	F

Z 1 ... Z 4: Zufalls-
ziffern
P: Anzahl der richtigen
Positionen
F: Anzahl der richtigen
"Farben"

2311	2310	230F	230E	230D	230C
T 1	T 2	T 3	T 4	V(10)	V(1)

T 1 ... T 4: aktueller
Tip
V(10): Nr. des Versuches
(10)
V(1): Nr. des Versuches
(1)

2317	2316	2315	2314	2313	2312
p	"P"			F	"F"

Anzeigebereich 2
P = Zahl im 7-Segment-
Code
F = Zahl im 7-Segment-
Code
"P" = Buchstabe P
"F" = Buchstabe F

231D	231C	231B	231A	2319	2318
Z 1'	Z 2'	Z 3'	Z 4'		

Z 1' ... Z 4':
Doppel von Z 1 ... Z 4
zum Vergleich

Soll dieses Spiel in andere Speicherbereiche übernommen werden (z. B. in einen EPROM), sind folgende Adressen zu modifizieren:

200F, 2010, 2015, 2016, 202D, 202E, 203C,
203D, 2046, 2047, 204B, 204C, 2087, 2088,
208B, 208C.

So, und nun viel Spaß!

4.2. Spielautomat

Sicher sind noch hin und wieder in alten Gaststätten oder auf dem Rummel Spielautomaten anzutreffen, die im Volksmund "Groschengräber" oder "einarmige Banditen" heißen. Wir wollen hier nicht zum Aufleben dieser zum Ausplündern der Gäste dienenden Geräte beitragen. Technisch ist ein solcher Spielautomat aber allemal interessant, besonders wenn man so etwas in modernster Technik und vor allem zu Hause besitzt. Das Prinzip ist einfach zu überschauen.

Nach Einwurf der Münze wird mit einem Hebel der Mechanismus in Gang gesetzt. Drei Rollen mit verschiedenen Symbolen (Pflaumen, Birnen und anderem "Obst") beginnen zu rotieren. Nach einer gewissen Zeit stoppt die erste Rolle, dann die zweite und die dritte. Dieses Anhalten kann auch in begrenztem Maße vom Spieler beeinflußt werden. Ziel ist es, bestimmte Dreierkombinationen (z. B. 3 gleiche Symbole) zu erreichen, bei denen ein Vielfaches der Einsatzsumme vom Automaten ausgezahlt wird.

Natürlich kann unser LC-80 nicht mit Pflaumen oder Birnen und schon gar nicht mit Geld dienen, aber daran ist der Spielbetrieb auch nicht gebunden. Wir wollen dennoch versuchen, das Original so nachzubilden, daß dieselbe Anzahl von Symbolen, die gleichen Gewinnchancen und sogar die Buchführung über gewonnene "Groschen" oder Spielverluste möglich ist. Wer will, kann damit seine eigene Spielbank aufmachen und ganz "naturgetreu" Freunde und Bekannte um ihr Geld bringen. Zusätzlich bietet sich die Möglichkeit, einem solchen Spielautomaten "auf die Finger zu sehen" und den Wert eines derartigen Gerätes für den Besitzer zu erkennen.

Die Einsätze der Spieler, die Chancen für Gewinnkombinationen sowie die ausgeschütteten Beträge sind nämlich so aufeinander abgestimmt, daß nur der Besitzer an einem Spielautomaten verdienen kann.

Der LC-80 bietet als Symbole wieder einmal Ziffern. Diese lassen wir über ein einfaches Zählprogramm auf den ersten drei Stellen des Displays "rotieren". Durch Drücken der [EX]-Taste wird das Programm auf Adresse 2000 gestartet. Die rollenden drei Ziffern erscheinen. Beim Drücken von [1] wird die erste Stelle, bei [2] die zweite und bei [3] die dritte gestoppt. Gewinne werden rechts angezeigt, möglich sind 3 oder 9 Spielmarken, die einen Wert von je 0,10 M symbolisieren.
Das Gewinnschema ist einfach:

bei Gleichheit der 1. und 2. Ziffer 3 (0,30 M),
bei Gleichheit der 1., 2. und 3. Ziffer 9 (0,90 M).

Mit der Wahl der möglichen Ziffern (1 ... 6), dem Gewinnschema und der bei jedem Spiel symbolisch einzuzahlenden Spielmarke (0,10 M) ergibt sich statistisch, daß 2/3 der eingezahlten Beträge als Gewinn wieder ausgezahlt werden. Der Rest bleibt im "Automaten". Zur Demonstration dieses Verhaltens wurde ein kleiner "Buchhalter" mit einprogrammiert, der bei stehenden Ziffern über [+] abgefragt werden kann.
Der Automat beginnt mit einem Startkapital von 10,00 M. Bei jedem Spiel kommen 0,10 M hinzu, jeder Gewinn wird automatisch abgezogen.
Mit [0] beginnt ein neues Spiel - nicht vergessen, einen Groschen einzuzahlen!
Viel Glück!

Programm "Spielautomat"

4.2.1. Hauptprogramm

```
2000   21 00 10      LD HL, 1000            } 10,00 M in
2003   22 16 22      LD (2216), HL          } Bank laden
2006   3E 03      M1:LD A, 03               } 03 in 2211
2008   32 11 22      LD (2211), A           } eintragen
```

```
200B   06 0C          LD B, 0C
200D   21 00 22       LD HL, 2200          Speicherbereich
2010   3E 00          LD A, 00             von 2200 bis
2012   77          M2:LD M, A              220C löschen
2013   23             INC HL
2014   10 FC          DJNZ FC        M2
2016   DD 21 00 22    LD IX, 2200          Anzeigebereich 1
201A   CD 7A 20    M3:CALL ROTANZ
201D   FE 05          CP 05          [1]?  1. Ziffer
201F   20 F9          JRNZ F9        M3    Stop?
2021   CD A1 20       CALL STOP
2024   CD 7A 20    M4:CALL ROTANZ
2027   FE 06          CP 06          [2]?  2. Ziffer
2029   20 F9          JRNZ F9        M4    Stop?
202B   CD A1 20       CALL STOP
202E   CD 7A 20    M5:CALL ROTANZ
2031   FE 07          CP 07          [3]?  3. Ziffer
2033   20 F9          JRNZ F9        M5    Stop?
2035   CD 5F 03       CALL PIEP 1
2038   CD AC 20       CALL VERGLEICH
203B   3A 06 22       LD A, (2206)         Wenn Gewinn = 0,
203E   FE 00          CP 00                dann B mit Ge-
2040   28 0B          JRZ 0B         M7    winnzahl (3 oder
2042   47             LD B, A              9) laden und
2043   C5          M6:PUSH BC              B-mal Wechsel-
2044   CD 5F 03       CALL PIEP 1          ton erzeugen
2047   CD 67 03       CALL PIEP 2
204A   C1             POP BC
204B   10 F6          DJNZ F6        M6
204D   CD D0 20    M7:CALL BILANZ
2050   CD 8C 20    M8:CALL ANZ
2053   FE 04          CP 04          [0]?  neues Spiel?
2055   28 AF          JRZ AF         M1
2057   FE 0A          CP 0A          [+]?  Anzeige des
2059   20 F5          JRNZ F5        M8    Bankguthabens?
```

```
205B   ED 5B 16 22    LD DE, (2216)          Eintragen des
205F   CD B7 04 *     CALL ADRSDP            Bankguthabens
2062   21 00 00       LD HL, 0000            in Anzeigebe-
2065   22 F2 23       LD (23F2), HL          reich 2, Lö-
2068   21 F6 23       LD HL, 23F6            schen der letz-
206B   CB E6          SET 4, (HL)            ten Stellen,
206D   DD 21 F2 23    LD IX, 23F2            Setzen des De-
2071   CD 5A 04*  M9: CALL DAK 1             zimalpunktes, Anzeige
2074   FE 00          CP 00          [0]?
2076   20 F9          JRNZ F9        M9      neues Spiel?
2078   18 8C          JR 8C          M1
```

4.2.2. Unterprogramm ROTANZ

```
207A   DD 46 11       LD B, (IX + 11)
207D   21 09 22       LD HL, 2209
2080   7E         R1: LD A, M                Erzeugung der
2081   3C             INC A                  3 rollenden
2082   FE 07          CP 07 (Endzahl + 1)    Ziffern (An-
2084   20 02          JRNZ 02        R2      zahl der noch
2086   3E 01          LD A,01 (Anfangszahl)  rollenden Zif-
2088   77         R2: LD M, A                fern im Re-
2089   23             INC HL                 gister B)
208A   10 F4          DJNZ F4        R1
208C   06 03          LD B, 03               Übertragen der
208E   21 09 22       LD HL, 2209            3 rollenden
2091   11 03 22       LD DE, 2203            Ziffern in den
2094   7E         R3: LD A, M                7-Segment-Code
2095   CD CA 04 *     CALL ONESEG            und Ablage im
2098   12             LD (DE), A             Anzeigebe-
2099   23             INC HL                 reich 1
209A   13             INC DE
209B   10 F7          DJNZ F7        R3
```

```
209D   CD 83 04 *    CALL DAK 2            Anzeige
20A0   C9            RET
```

Das Unterprogramm ANZ ist ein Teil von ROTANZ (ab Adresse 208C).

4.2.3. Unterprogramm STOP

```
20A1   3A 11 22      LD A, (2211)          Herabsetzen
20A4   3D            DEC A                 der Anzahl der
20A5   32 11 22      LD (2211), A          rotierenden
20A8   CD 5F 03      CALL PIEP 1           Ziffern
20AB   C9            RET
```

4.2.4. Unterprogramm VERGLEICH

```
20AC   21 0B 22      LD HL, 220B           Vergleich der
20AF   7E            LD A, M               1. mit der 2.
20B0   2B            DEC HL                Ziffer
20B1   BE            CP M
20B2   20 1B         JRNZ 1B       V1
20B4   3E 03         LD A, 03              Gewinn 3, wenn
20B6   32 06 22      LD (2206), A          1. und 2. Ziffer
20B9   CD CA 04      CALL ONESEG           gleich, Umformen
20BC   32 00 22      LD (2200), A          in 7-Segment-Code
20BF   7E            LD A, M               Vergleich der
20C0   2B            DEC HL                2. mit der 3.
20C1   BE            CP M                  Ziffer
20C2   20 0B         JRNZ 0B       V1
```

```
20C4   3E 09          LD A, 09          ⎫ Gewinn 9, wenn
20C6   32 06 22       LD (2206), A      ⎪ 1., 2. und 3.
20C9   CD CA 04 *     CALL ONESEG       ⎪ Ziffer gleich,
20CC   32 00 22       LD (2200), A      ⎭ Umformung
20CF   C9          V1: RET
```

4.2.5. Unterprogramm BILANZ

```
20D0   3A 06 22       LD A, (2206)      ⎫ Gewinnziffer
20D3   07             RLCA              ⎪ 4x links ver-
20D4   07             RLCA              ⎪ schieben (aua
20D5   07             RLCA              ⎪ (09 wird 90),
20D6   07             RLCA              ⎪ Ergebnis in
20D7   E6 F0          AND F0            ⎪ Register D
20D9   57             LD D, A           ⎭ laden
20DA   3A 16 22       LD A, (2216)      ⎫ Gewinn vom
20DD   92             SUB D             ⎪ Guthaben (Gro-
20DE   27             DAA               ⎪ schen) abzie-
2ODF   32 16 22       LD (2216), A      ⎪ hen, bei Über-
20E2   30 09          JRNC 09        B1 ⎪ trag, Markbe-
20E4   3A 17 22       LD A, (2217)      ⎪ trag um 1 ver-
20E7   D6 01          SUB 01            ⎪ mindern, neue
20E9   27             DAA               ⎪ Guthaben ab-
20EA   32 17 22       LD (2217), A      ⎭ speichern
20ED   3A 16 22    B1: LD A, (2216)      ⎫ Spieleinsatz
20F0   C6 10          ADD 10            ⎪ (10) zu Gut-
20F2   27             DAA               ⎪ haben (Groschen)
20F3   32 16 22       LD (2216), A      ⎪ addieren, bei
20F6   30 09          JRNC 09        B2 ⎪ Übertrag Mark-
20F8   3A 17 22       LD A, (2217)      ⎪ betrag um 1 er-
20FB   C6 01          ADD 01            ⎪ höhen, Guthaben
20FD   27             DAA               ⎪ abspeichern
20FE   32 17 22       LD (2217), A      ⎭
2101   C9          B2: RET
```

Für die Umsetzung des Programmes in einen anderen Speicherbereich ist die Änderung folgender Adressen notwendig:

201B	2025	202F	204E
201C	2026	2030	204F
2022	2020	2039	2051
2023	202D	203A	2052

4.2.6. Speicherorganisation

Anzeigebereich

2205	2204	2203	2202	2201	2200
"Z 1"	"Z 2"	"Z 3"			"G"

220B	220A	2209	2208	2207	2206
Z 1	Z 2	Z 3			G

2217	2216
B 10	B 1

2211
D

Anzeigebereich 2

23F7	23F6	23F5	23F4	23F3	23F2
"B 10"	"B 10"	"B 1"	"B 1"		
"Mark"		"Groschen"			

Z 1, Z 2, Z 3	rollende Ziffern
"Z 1", "Z 2", "Z 3"	rollende Ziffern in 7-Segment-Form
G	Gewinn in Vielfachem des Spieleinsatzes
"G"	Gewinn in 7-Segment-Form
D	Anzahl der noch rollenden Ziffern
B 1, B 10	Groschen, Mark des Guthabens (bei Spielbeginn 10,00 M)
"B 1", "B 10"	Guthaben in 7-Segment-Form

4.3. Sterntaler

Dieses Spiel fordert unsere ganze Geschicklichkeit.
Das Display stellt das "Spielfeld" dar, auf dem es "Taler" regnet. Letztere werden durch zufällig nach unten fallende Leuchtbalken der 7-Segment-Anzeigen symbolisiert. Leider fallen die Taler nach unten durch und sind dann durch nichts mehr zu erwischen.
Glücklicherweise gibt es einen "Topf", durch einen Leuchtpunkt gebildet, der sich über die Tasten [0] und [3] hin und her bewegen läßt. Mit dem können wir die herabfallenden Taler auffangen, wenn wir ihn in die entsprechende Position rechts neben dem Leuchtbalken gebracht haben.
Das Ganze wird im Spielverlauf immer komplizierter und die "aufgefangenen Taler" werden mit einem Ton quittiert. Gezählt werden sie auch, so daß am Spielende jeder weiß, wie geschickt er gespielt hat.
Ein neues Spiel beginnt, wenn die Taste [+] gedrückt wird.

Also los!

4.3.1. Hauptprogramm

```
2000   06 10       H0: LD B, 10            ┐ Löschen des
2002   21 10 22        LD HL, 2210          │ Anzeige-
2005   36 00       H1: LD M, 0             ├ und
2007   2B              DEC HL               │ Speicher-
2008   10 FB           DJNZ FB        H1    ┘ bereiches
200A   DD 21 00 22    LD IX, 2200
200E   36 10           LD M, 10              Punkt setzen
2010   7D              LD A, L              ┐ Position des
2011   32 20 22        LD (2220), A         ┘ Punktes in 2220
2014   3E 1C           LD A, 1C             ┐ Startpunkt des
2016   32 22 22        LD (2222), A         ┘ Rückwärtszählers
2019   16 20       H2: LD D, 20              Länge der Stufen
201B   CD 85 20    H3: CALL STERNE
201E   3A 22 22        LD A, (2222)
2021   47              LD B, A              ┐ Schrittweise
2022   C5          H4: PUSH BC              │ Erhöhung der
2023   CD 50 20        CALL FÄNGER          │ Geschwindig-
2026   C1              POP BC               │ keit (Stufen-
2027   10 F9           DJNZ F9        H4    │ größe in
2029   15              DEC D                │ 2030 verän-
202A   20 EF           JRNZ EF        H3    │ derbar,
202C   3A 22 22        LD A, (2222)         │ Startwert
202F   D6 02           SUB 02 (04, 01)      │ in 2015)
2031   28 05           JRZ 05         H5    │ Stufen
2033   32 22 22        LD (2222), A         │ 01, 02, 04
2036   18 E1           JR E1          H2    ┘
2038   ED 5B 08 22 H5:LD DE, (2208)
203C   CD B7 04 *      CALL ADRSDP          ┐ Punktestand
203F   DD 21 F2 23     LD IX, 23F2          ├ im Anzeige-
2043   CD 5A 04 * H6: CALL DAK 1            ┘ bereich 2, Anzeige
```

```
2046   FE 10          CP 10          [+]?
2048   28 B6          JRZ B6         H0      neues Spiel
204A   18 F7          JR F7          H6      zurück zur Anzeige
```

4.3.2. Unterprogramm FÄNGER

```
2050   3A 20 22       LD A, (2220)           Pos. Fänger
2053   6F             LD L, A          }     als HL-Wert
2054   26 22          LD H, 22         }     aufbereiten
2056   06 08          LD B, 08               Geschwindigk.
2058   3E FF          LD A, FF
205A   CD 83 04*  F1: CALL DAK 2
205D   10 FB          DJNZ FB        F1
205F   FE 04          CP 04          [0]?    links?
2061   28 06          JRZ 06         F2
2063   FE 07          CP 07          [3]?    rechts?
2065   28 0E          JRZ 0E         F3
2067   18 16          JR 16          F4
2069   3E 05      F2: LD A, 05               linke Grenze
206B   BD             CP L                   erreicht?
206C   28 11          JRZ 11         F4      wenn nicht,
206E   CB A6          RES 4, M               Punkt nach
2070   23             INC HL                 links ver-
2071   CB E6          SET 4, M               schieben
2073   18 0A          JR 0A          F4
2075   3E 00      F3: LD A, 00               rechte Grenze
2077   BD             CP L                   erreicht?
2078   28 05          JRZ 05         F4      wenn nicht,
207A   CB A6          RES, M                 Punkt nach
207C   2B             DEC HL                 rechts ver-
207D   CB E6          SET 4, M               schieben
207F   7D         F4: LD A, L                neue Position
2080   32 20 22       LD (2220), A           des Fängers
2083   C9      RET                           eintragen
```

4.3.3. Unterprogramm STERNE

```
2085   06 07          LD B, 07              Prüfung, ob
2087   21 00 22       LD HL, 2200           im Anzeige-
208A   CB 7E      S1: BIT 7, M              bereich 1
208C   C4 BE 20       CANZ VERGL            irgendwo Bit 7
208F   CB BE           RES 7, M              (unterer Bal-
2091   23             INC HL                ken) gesetzt
2092   10 F6          DJNZ F6       S1      ist
2094   06 07          LD B, 07              Prüfung, ob im
2096   21 00 22       LD HL, 2200           Anzeigebereich
2099   CB 5E      S2: BIT 3, M              Bit 3 (mittle-
209B   28 04          JRZ 04        S3      rer Balken)
209D   CB FE          SET 7, M              gesetzt ist,
209F   CB 9E          RES 3, M              diesen löschen,
20A1   23          S3: INC HL                Bit 7 setzen
20A2   10 F5          DJNZ F5       S2      (Verschiebung
                                             nach unten)
20A4   06 07          LD B, 07              Prüfung auf
20A6   21 00 22       LD HL, 2200           Bit 2 (oberer
20A9   CB 56      S4: BIT 2, M              Balken), diesen
20AB   28 04          JRZ 04        S5      löschen, Bit 3
20AD   CB DE          SET 3, M              setzen
20AF   CB 96          RES 2, M              (Verschiebung
20B1   23          S5: INC HL                nach unten)
20B2   10 F5          DJNZ F5       S4
20B4   ED 5F          LD A, R               an zufälliger
20B6   E6 07          AND 07                Stelle neues
20B8   6F             LD L, A               Bit 2 (oberer
20B9   26 22          LD H, 22              Balken) setzen
20BB   CB D6          SET 2, M
20BD   C9             RET
```

4.3.4. Unterprogramm VERGL

```
20BE   3A 20 22      LD A, (2220)          Position Fänger
20C1   BD            CPL
20C2   20 25         JRNZ 25        V2
20C4   3A 08 22      LD A, (2208)          Punktzähler
20C7   C6 01         ADD 01                (Einer, Zehner)
20C9   27            DAA                   um 1 erhöhen
20CA   32 08 22      LD (2208), A
20CD   30 09         JRNC 09        V1
20CF   3A 09 22      LD A, (2209)          wenn Übertrag,
20D2   C6 01         ADD 01                Punktzähler
20D4   27            DAA                   (Hunderter, Tau-
20D5   32 09 22      LD (2209), A          sender) um 1 erhöhen
20D8   C5         V1: PUSH BC
20D9   E5             PUSH HL
20DA   D5             PUSH DE
20DB   CD 5F 03       CALL PIEP 1           Ton bei Treffer
20DE   D1             POP DE
2ODF   21 F2 23       LD HL, 23F2           die rechten
20E2   36 00          LD M, 00              beiden Stellen
20E4   23             INC HL                des Anzeigebe-
20E5   36 00          LD M, 00              reiches 2 löschen
20E7   El             POP HL
20E8   C1             POP BC
20E9   C9         V2: RET
```

4.3.5. Speicherorganisation

```
2200 ... 2205  Anzeigebereich 1 (Spielfeld)
2208           Punktzähler (Einer, Zehner)
2209           Punktzähler (Hunderter, Tausender)
2220           aktuelle Position des Fängers
2222           aktuelle Geschwindigkeit der fallenden Balken
```

2015 Festlegung der Anfangsgeschwindigkeit
2030 Stufengröße für den Geschwindigkeitszuwachs

Achtung!
Für Änderungen von Geschwindigkeitsstufen bzw. Anfangsgeschwindigkeit der fallenden Balken muß beachtet werden, daß das Ergebnis der Subtraktion (auf Adresse 202F) irgendwann einmal 0 ergibt, da dies auf Adresse 2031 das Kriterium für Spielende ist.

23F2 ... 23F7 Anzeigebereich 2(Trefferanzeige)
2057 Geschwindigkeit des Fängers

Diese drei Spiele mögen als Auswahl genügen. Der LC-80 hat bewiesen, daß er durchaus als Spielpartner akzeptiert werden kann. Zugleich verkörpern die gezeigten Beipiele jeweils die grundsätzlichen Spielbereiche:

- Spiele mit Logikcharakter
- Glücksspiele
- Geschickliohkeits- und Reaktionsspiele.

Sowohl die beiden letzten Gruppen als auch die erste sind grundsätzlich beliebig erweiterungsfähig, wie z. B. die Schachcomputer beweisen. So gibt es in der Hardware kaum wesentliche Unterschiede zwischen dem LC-80 und dem ersten Schachcomputer des VEB Mikroelektronik "Karl Marx" Erfurt, dem SC 2. Wer ist der erste, der mit seinem LC-80 Schach spielen kann??

5. Die Sinne eines Computers

Ohne Informationen aus der Umwelt über Sinnesorgane wäre jedes Leben unmöglich - dies gilt für den kleinsten Einzeller genauso wie für den Menschen.
Auch ein Computer benötigt Informationen aus der Umwelt, wenn er darauf reagieren soll. Dabei sind die zu verarbeitenden "Umweltreize" so vielfältig wie die Anwendungsbeipiele von Mikrorechnern selbst.
Beispiele für solche Informationen sind:

- Position (beliebige Koordinaten: X, Y, Z oder Winkel)
- Temperatur (absoluter Wert, Differenzen)
- Licht (optisch sichtbar, infrarot, ultraviolett, Kontraste, Farben, komplette Bilder)
- Druck (mechanisch, Gase)
- Menge (statisch oder dynamisch, z. B. Durchflußmenge/Zeit)
- Feldstärke (elektrisch, magnetisch, HF)
- Spannung, Strom, Widerstand, Kapazität, Induktivität, Frequenz
- Schall (hörbar, Ultraschall, Infraschall)
- Bewegung, Beschleunigung, Drehzahl, Schwingungen
- Material (Leiter oder Nichtleiter, durchsichtig oder nicht, Säure oder Lauge, weich oder hart usw.)

Wir sehen, daß es eine Menge von Umweltdaten gibt, die für eine Computeranwendung interessant sein können. Wir erkennen aber auch, daß viele dieser Informationen auch für den Menschen nicht direkt erfaßbar sind, sondern erst in eine
für ihn und seine Sinnesorgane "lesbare" Form umgewandelt werden müssen. So ist z. B. die magnetische Feldstärke vom Menschen nicht wahrnehmbar und erst die Umformung durch ein Feldstärkemeßgerät in einen sichtbaren Zeigerausschlag erlaubt ihm deren Bestimmung. Dasselbe Prinzip gilt für den

Computer.
Wenn wir unseren LC-80 "sensibel" für Umweltreize machen wollen, müssen wir ihm diese Informationen in einer für ihn auswertbaren Form anbieten. Das bedeutet, daß alle Daten in digitale, elektrische Informationen umgewandelt werden. Dies geschieht fast immer in zwei Schritten:

1. Schritt: Umwandlung der zu erfassenden Größe in elektrische Form (Spannung, Strom, .Widerstand, Kapazität, Frequenz, Impulslänge usw.)
2. Schritt: Digitalisierung der elektrischen Information (im einfachsten Fall eine Ja/Nein-Entscheidung oder eine Analog-Digital-Wandlung mit Zwischenwerten)

Sehr schön können wir das an zwei Beispielen erkennen, die wir in vorangegangenen Kapiteln des Buches schon realisiert haben:

- Digitalvoltmeter (Teil I, Kapitel 7)
  Der Temperaturfühler (Platin-Widerstandsthermometer Pt 100) setzt zunächst Temperaturen in Widerstandswerte um. Danach erfolgt die Digitalisierung mit einem A/D-Wandler C 520 D.
- Lochstreifenleser (Teil III, Kapitel 2)
  Hier wird die unterschiedliche Helligkeit (IR-Licht) am Fototransistor bei "Loch" und "kein Loch" zur Ja/Nein-Auswertung (H/L-Pegel am PIO-Eingang) benutzt.

5.1. Der erste Schritt - Sensoren

Obwohl auch international ein erheblicher Entwicklungsrückstand der Sensortechnik gegenüber dem Stand- der aktiven Bauelemente - insbesondere der Mikroprozessorentwicklung -

festzustellen ist, gibt es auch in der DDR eine ganze Reihe modernster Sensoren z. B. auf Halbleiter- oder Spezialkeramikbasis. Diese weisen zum Teil auch komplette Meßwertverarbeitungseinheiten auf. Beispiele sind CCD-Elemente als komplexe optische Sensoren oder Hall-Schaltkreise zum Nachweis magnetischer Felder.
Aber auch ohne diese (zum Teil sehr teuren!) Spezialbauelemente wollen wir jetzt einige für uns nutzbare Sensorprinzipien kennenlernen. Dabei legen wir weniger Wert auf große Genauigkeit oder Linearität der Sensoren, dafür aber mehr auf einfache Realisierbarkeit und geringen Aufwand für den Betrieb.
Ein wichtiger Grundsatz soll dabei sein, daß eine einheitliche Weiterverarbeitung der Meßwerte erfolgen kann. Das wird dadurch möglich, daß wir alle Informationen in Widerstands- bzw. Spannungswerte umwandeln. Zur Digitalisierung können wir dann die schon in Teil I, Kapitel 7 aufgebaute A/D-Wandlerschaltung mit dem 0 520 D nutzen.

5.1.1. Lichtsensor

Es gibt eine ganze Reihe von optischen Wandlern, die auf Lichtänderungen mit Veränderung ihrer elektrischen Eigenschaften reagieren:

- Fotozellen
- Fotodioden
- Fototransistoren
- Fotoelemente (Selen, Silizium)
- Bildaufnahmeröhren, CCD-Zeilen usw. zur Bildwandlung
- Fotowiderstände.

Für unsere Anwendungen sind Fotowiderstände interessant. Sie ermöglichen eine direkte Umwandlung des Helligkeitswertes in einen Widerstandswert, unabhängig von der Polarität der angelegten Spannung.

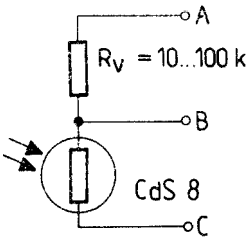


Bild 7: Lichtsensor

5.1.2. Temperatursensor

Im Gegensatz zu dem im Teil I, Kapitel 7 beschriebenen Temperaturmeßgerät mit Pt 100 wollen wir hier einen einfacheren (und sehr viel billigeren!) Sensor benutzen. Nachteile bezüglich Genauigkeit und Linearität nehmen wir dabei in Kauf. Auch hier gibt es eine Reihe weiterer Sensorprinzipien:

- Thermoelemente
- Siliziumdioden (deren Flußspannung ist linear abhängig von der Sperrschichttemperatur)
- Widerstandsmeßfühler (z. B. Pt 100)
- Kaltleiter (Halbleiter, deren Widerstand bei Erwärmung ansteigt)
- Heißleiter (Halbleiter, deren Widerstand bei Erhöhung der Temperatur sinkt).

Wir entscheiden uns für einen billigen Heißleitersensor.

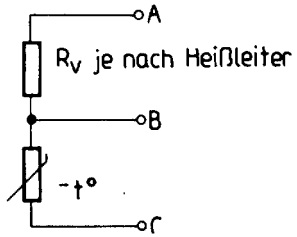


Bild 8: Heißleitersensor

5.1.3. Positionssensor

Wir benutzen einfache verstellbare Widerstände:

- Drehwiderstände (Potentiometer) für Winkelbewegungen
- Schieberegler für lineare Bewegungen

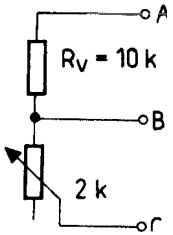


Bild 9: Positionssensor

5.1.4. Feuchtsensor, Flüssigkeitsmelder

Zwei Stäbe, möglichst aus nicht rostendem Stahl, die isoliert montiert in ein Gefäß hineinragen, bilden beim Eintauchen in eine leitfähige Flüssigkeit einen Widerstand, der von der Eintauchtiefe und der Leitfähigkeit abhängt. In einen Blumentopf gesteckt, ist der Widerstand ein Maß für die Erdfeuchtigkeit.

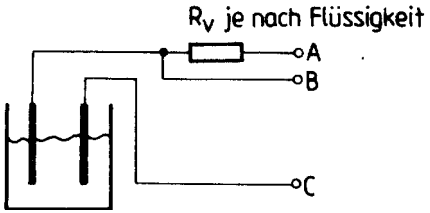


Bild 10: Flüssigkeitsmelder

5.1.5. Berührungssensor

Nach dem gleichen Prinzip arbeiten sog. Sensortasten an elektronischen Geräten. Hier wird der Widerstand der menschlichen Haut zum Schalten von Gerätefunktionen benutzt.

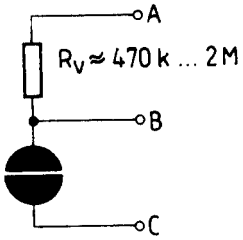


Bild 11: Berührungssensor

5.1.6. Beschleunigungssensor

Ein "Pendel" wird starr mit der Achse eines leichtgängigen Potentiometers verbunden. Beim Beschleunigen bzw. Abbremsen der Anordnung schlägt das Pendel aus und verändert damit den Widerstandswert des Potentiometers. Dessen natürliche Reibung dämpft die Schwingungen.

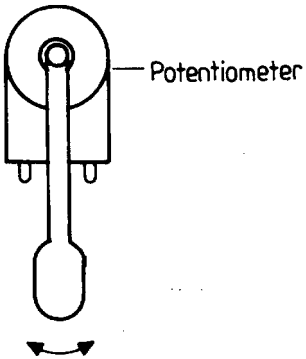


Bild 12: Beschleunigungssensor

5.1.7. Drucksensor

Es gibt Halbleiter-Drucksensoren, die aber dem Amateur kaum zugänglich sind. Zur Not kann man Dosenbarometer, die meist einen Übersetzungsmechanismus für den Zeigerantrieb besitzen, mit einem Potentiometer koppeln.

5.1.8. Massesensor

Als "elektronische Waage" eignet sich ein Potentiometer mit Feder.

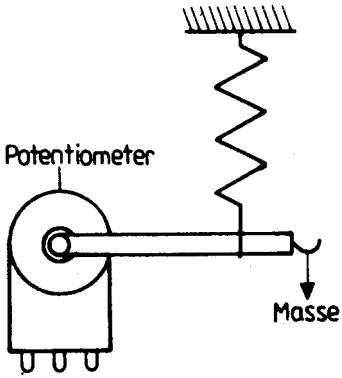


Bild 13: Massesensor

5.1.9. Andere Prinzipien

Diese Anregungen lassen sich beliebig fortsetzen, beim Austüfteln von Sensorprinzipien ist Phantasie gefragt. Oft werden recht "indirekte" Methoden benutzt, um zu Ergebnissen zu kommen, wie die zwei folgenden Beispiele zeigen.

5.1.9.1. Durchflußmengensensor

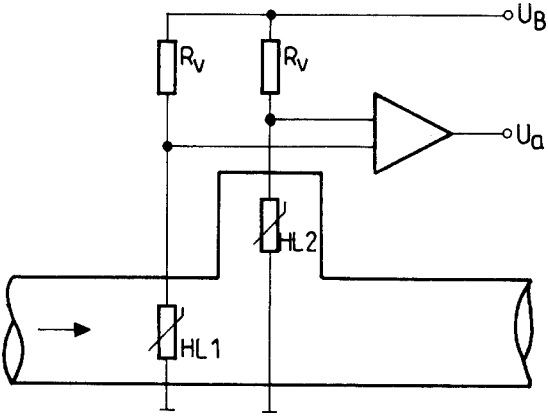


Bild 14: Durchflußmengensensor

Der durch beide Kombinationen R_V/HL (Heißleiter) fließende Strom erwärmt beide Heißleiter. Ist die Flüssigkeit in Ruhe, werden HL 1 und HL 2 in gleichem Maße aufgeheizt und es stellt es sich ein Temperaturgleichgewicht ein. Bei strömender Flüssigkeit erfolgt die Wärmeabfuhr an HL 1 wegen der direkten Umströmung stärker als an HL 2, sein Widerstand wird damit größer und das Potential am Spannungsteiler verschiebt sich. Der nachfolgende Differenzverstärker wertet das aus.

5.1.9.2. Gassensor für oxydierbare Gase

Oxydierbare Gase wie Flüssiggas, Leuchtgas und Kohlenmonoxid kommen im Haushalt vor, aber auch in der industriellen Praxis kann es die Aufgabe von Mikrorechnern werden, deren Vorhandensein oder die jeweilige Konzentration dieser Gase zu melden oder zu messen. Für den Nachweis von Gasen oder Gasanteilen gibt es eine Vielzahl von Sensoren, die aber dem Amateur kaum zugänglich sind. Für eigene Experimente kann man auf die in jedem Elektrogeschäft erhältlichen Gasanzünder-Glüheinsätze zurückgreifen. Diese glühlampenähnlichen Gebilde mit E 10-Sockel besitzen einen Draht, der beim Stromdurchfluß (ca. 100 mA) kaum sichtbar glüht. Beim Vorhandensein von Stadtgas, Flüssiggas oder Kohlenmonoxid leuchtet dieser Draht dann sehr hell auf und erhöht sprunghaft seinen Widerstand. Der dargestellte Effekt kann sofort mit einem Gasfeuerzeug überprüft werden (natürlich ohne das Gas vorher zu entzünden!). Wegen der Eigenschaft des Flüssiggases, schwerer als Luft zu sein, sollte der Sensor in eine Tasse gelegt werden und danach das Ventil des Feuerzeuges über dieser Tasse geöffnet werden.
Ein Nachteil dieses auf katalytischem Prinzip arbeitenden Sensors ist, daß vorhandenes Gas von ihm selbst entzündet wird - das ist aber der normale Verwendungszweck eines Gasanzünders. Wählt man den Grundstrom entsprechend klein und verwendet ein feinmaschiges Metallsieb zur Abschirmung des Sensorraumes, kann dieser Zündeffekt gut ausgeschlossen werden. Trotzdem ist zu extremer Vorsicht beim Umgang mit allen genannten Gasen zu raten, es besteht Vergiftungs-, Brand- und Explosionsgefahr!
Der zum Sensor umfunktionierte Gasanzündereinsatz ist sogar zur Kontrolle von Kfz-Abgasen auf Kohlenmonoxid- und Kohlenwasserstoffanteile einsetzbar, wie praktische Untersuchungen gezeigt haben.

Die beiden zuletzt beschriebenen Sensoren sind für Hobby-Experimente geeignet. Wir wollen für die weiteren Betrachtungen allerdings nur die unter 5.1.1. bis 5.1.8. beschriebenen Muster nutzen.

5.2. Meßwertumwandlung

Um den Bauaufwand in erträglichen Grenzen zu halten, benutzen wir den in Teil I, Kapitel 7 beschriebenen A/D-Wandler mit C 520 D zur Kopplung mit dem LC-80.
Die Beschaltung der Eingänge des C 520 D mit den Sensoren erfolgt dann so:

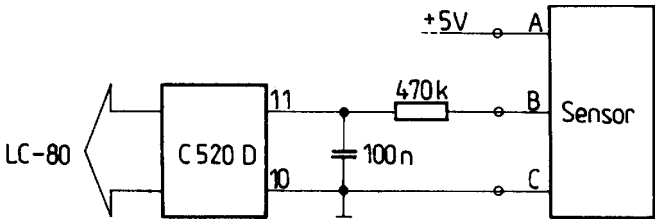


Bild 15: Beschaltung der Eingänge des C 520 D

Das zusätzliche RC-Glied dient zur Erhöhung des Eingangswiderstandes des A/D-Wandlers. Damit wird es möglich, das Widerstandsverhältnis beider Teilwiderstände der Sensoren optimal an den jeweiligen Anwendungsfall anzupassen.

5.3. Anwendungshinweise

Temperatur-, Licht-, Druck- und andere Meßgeräte sind dem Elektroniker geläufig und es gibt genügend Schaltungen für ihre Realisierung. Das grundsätzlich Neue bei der Anwendung von Sensoren in der Mikrorechnertechnik ist, daß sich nahezu beliebige Kombinationen verschiedener Meßgrößen, eine sehr große Anzahl von Meßstellen, eine beliebige Meßwertumformung durch Rechenoperationen und eine auf den Anwendungsfall zugeschnittene Reaktion des Computers realisieren lassen. Es würde den Rahmen dieses Buches sprengen und auch nur einen kleinen Teil der Leser interessieren, ein komplettes Anwendungsbeispiel mit Hard- und Software vorzustellen. Dagegen sollen einige allgemeingültige Hinweise zum Aufbau solcher Systeme gegeben und anschließend ein paar reizvolle Aufgabenstellungen genannt werden.

Zuerst das Problem, dem Rechner mehrere Meßgrößen zuzuführen. Es wird gelöst durch einen interessanten Schaltkreis des CMOS-Sortimentes des VEB Mikroelektronik "Karl Marx" Erfurt, den 8-Kanal-Analog-Multiplexer V 4051 D. Dieser Schaltkreis schaltet 8 analoge Eingangssignale (Z 0 ... Z 7) je nach Belegung der drei Steuereingänge A, B und C (3 bit) auf einen Ausgang Y. Die Schaltung in Bild 16 zeigt das Zusammenwirken von LC-80, C 520 D und V 4051 D. Die drei Steuerbits werden vom Port B (B 0, B 1 und B 2) zur Verfügung gestellt. Je nach Belegung dieser PIO-Anschlüsse wird ein Sensor über den V 4051 D und den C 520 D in den LC-80 eingelesen.
Die noch freien Leitungen A 7 und B 3 können abhängig von der PIO-Programmierung als Schaltaus- bzw. Eingänge verwendet werden. Der dargestellte Multiplexbetrieb ist jedoch nicht auf die analogen Eingangsgrößen beschränkt. Ebenso lassen sich die digitalisierten Daten am Ausgang des C 520 D multiplexen. So wäre es z. B. reizvoll, an gleicher Stelle eine externe

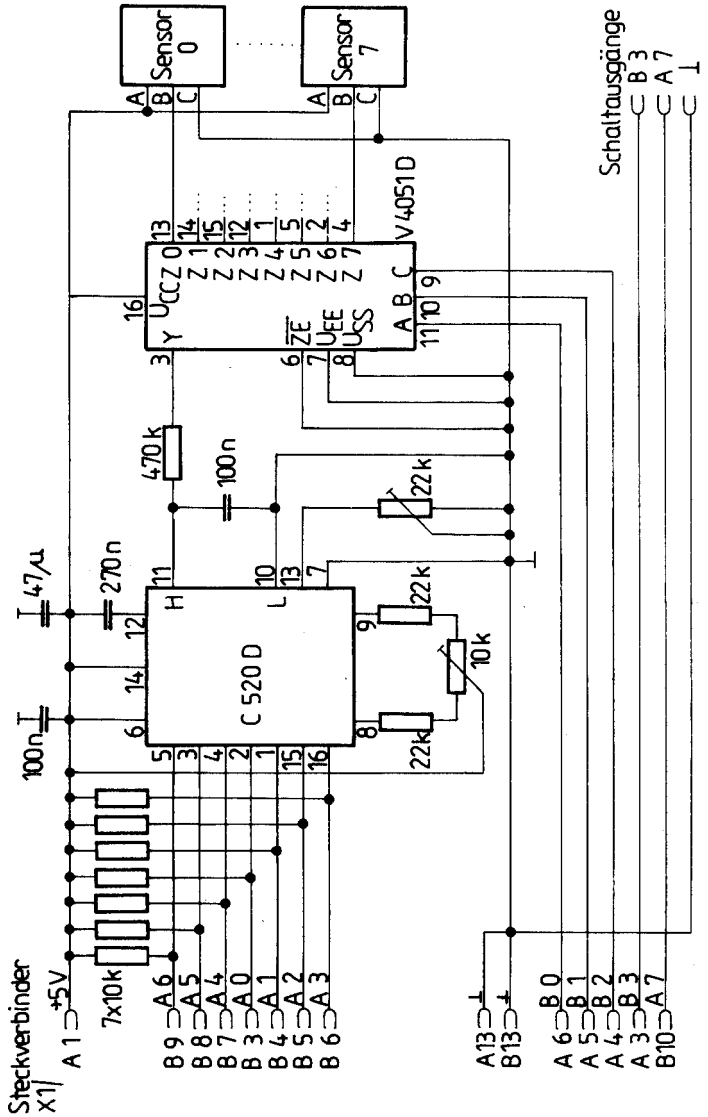


Bild 16: Multiplexbetrieb mit max. 8 Sensoren

Quarzuhr mit eigener Stromversorgung einzukoppeln. Man könnte zum Einlesen praktisch dasselbe Programm wie für den C 520 D verwenden und hätte dann die Möglichkeit, die Meßgröße in Verbindung mit der Zeit im Rechner zu verarbeiten bzw. auszugeben oder zu speichern.
Sollte für eine (fast schon professionelle) Anlage der USER-Bus des LC-80 nicht mehr ausreichen, hilft eine zusätzliche PIO, deren Anschluß an den LC-80 im Kapitel 7 beschrieben wird.

Zum Abschluß dieses Kapitels nun noch einige Vorschläge für die Verwendung der Sensortechnik in Verbindung mit dem LC-80. Die Aufgabenstellungen sollen natürlich nur Anregungen sein und können beliebig erweitert oder eingeschränkt werden:

- Temperaturerfassungssystem für die Wohnung (oder im Betrieb) für mehrere Zimmer. Dabei könnte eine (programmierte oder zusätzlich angeschlossene) Uhr festlegen, zu welcher Zeit in welchem Zimmer welche Temperatur eingestellt werden soll. Die jeweiligen Werte werden mit Sollwerten verglichen und entsprechend nachgeregelt. Die Stellglieder könnten kleine Motoren sein, die z. B. Heizungsventile gemäß Kapitel 3 in die entsprechende Lage fahren.
- Strichcode-Leser mit optischem Sensor zur Entschlüsselung von Strichcode-Informationen
- komplexes Steuersystem für die Dunkelkammer
  Wohl jeder Fotoamateur kennt das Problem, die richtige Blendeneinstellung und Belichtungszeit beim Vergrößern zu ermitteln. Ein oder mehrere Fotowiderstände können in der Papierebene so angebracht werden, daß sie eine mittlere Beleuchtungsstärke über das gesamte Papierfeld feststellen. Dieser Wert kann mit Hilfe verschiedener Tabellen im LC-80 (Papiersorte usw.) so umgesetzt

werden, daß gleich die Belichtungszeit in s. ausgegeben wird. Noch eleganter wäre die unmittelbare Zu- bzw. Abschaltung der Lampe des Vergrößerungsgerätes durch den Computer. Hier sollten aber nur erfahrene "Elektriker" das Werkzeug zur Hand nehmen, sonst wird's gefährlich!

- Ein "Lichtmengenschalter" für das Blumenfenster könnte ähnlich aufgebaut werden. Hier wird z. B. in jeder Minute die Helligkeit gemessen und die Werte ständig addiert. Sinkt das Tageslicht unter eine festgelegte Schwelle (am Abend), wird die Differenz der "Lichtsumme" zu einem vorgegebenen Sollwert ermittelt und eine künstliche Lichtquelle solange eingeschaltet bis dieser Sollwert erreicht ist.
  Vielleicht auch keine schlechte Anregung für landwirtschaftliche Spezialkulturen...
- Mit einem Feuchtsensor können die Blumentöpfe während des Sommerurlaubs kontrolliert und mit einer kleinen Spielzeugwasserpumpe nachgefüllt werden. Aber Vorsicht! Jeder Stromausfall kann eine Überschwemmung verursachen...

Es zeigt sich, daß ein Mikrorechner ein dienstbarer Geist sein kann. Leider stellen wir aber auch fest, daß mit jeder weiteren Anwendung des LC-80 dieser immer wieder blockiert ist. Man müßte mehrere haben ...
Ein Ausweg aus dieser Misere ist der im Kapitel 6 gezeigte Weg - der Selbstbau eines Computers für einen einzigen Zweck. Hier wird auch gezeigt, wie man zu einer solchen Entwicklung den LC-80 zu Hilfe nehmen kann und auch, wie man das Betriebssystem eines solchen Spezialcomputers vor Problemen im Zusammenhang mit Störungen und Netzausfällen schützen muß. Gerade das letzte Beispiel eben ist typisch für eine solche Herangehensweise. Ein solches System muß beim Wiedereinschalten der Netzspannung von allein anlau-

fen, seine Programme abarbeiten und danach wieder in einen Wartezustand gehen, der z. B. durch einen Interrupt einer äußeren Uhr aufgehoben wird. Das Ganze setzt feste Programme in EPROMs voraus und erfordert viel Systemdenken - lohnt aber den Aufwand.

6. Wenn Computer Kinder kriegen ...

Der LC-80 ist, wie wir inzwischen festgestellt haben, ein recht universelles Gerät. Je nach Hard- und Softwareausrüstung kann er Uhr, Spielautomat, Steuerzentrale oder Miniorgel sein. Einen Haken hat das Ganze - man hat nie ein "fertiges" Gerät, wie das beim traditionellen Basteln üblich war. Universalität hat eben auch ihren Preis. Aber - beim Basteln und Programmieren sind wir auf den Geschmack gekommen, wir kennen die Vorzüge unseres Mikroprozessors U 880 D und die Anpassungsfähigkeit des gesamten Computersystems. Wenn wir aus naheliegenden Gründen den LC-80 nicht als Uhr an die Wand hängen oder als Fernthermometer entfremden wollen, müssen wir uns etwas einfallen lassen, um das Mikrorechnerkonzept auch im Hobbybereich zu nutzen.
Wir bauen einfach einen "zweckgebundenen" Computer, der erheblich einfacher ausfallen kann als der LC-80.
Wie das funktioniert und wie man den LC-80 zur Hard- und Softwareentwicklung (quasi als "Geburtshelfer") nutzen kann, wird in diesem Kapitel anhand eines einfachen Beispieles demonstriert.
Es ist nun nicht ganz leicht, ein geeignetes Objekt zu finden, an dem einerseits die Strategie bei der Entwicklung singulärer Lösungen mit Mikrorechner und deren Simulation auf dem LC-80 beispielhaft durchgespielt werden können und andererseits der Hardwareaufwand (und damit der Preis!) in vertretbaren Grenzen bleibt. Außerdem sollte eine Beispiellösung auch noch einen erstrebenswerten Nutzen für den Bastler bringen - es muß sich also lohnen.
Vorgeschlagen wird ein Melodiegenerator, der auf Knopfdruck Volkslieder, Nationalhymnen oder Weihnachtslieder spielt, ganz nach Wunsch. Die Melodien können mit dem LC-80 individuell programmiert werden - vorausgesetzt, man besitzt einen EPROM U 2716 C und das im Heft 2 vorgestellte

Programmiermodul sowie die ebenfalls dort erstellte RAM- und ROM-Erweiterung mit dem Ladeprogramm.
Und wie kann man einen solchen Melodiegenerator verwenden? Zum Beispiel als eindrucksvolle Haustür-"Klingel" oder als elektronisches Innenleben einer Spieldose...

6.1. Pflichtenheft

Wir nehmen uns vor, soft- und hardwareseitig einen Melodiegenerator zu entwickeln, der bei der jeder Auslösung eine von mehreren möglichen Melodien spielt. Als Auslösungsprinzip wählen wir die Aufhebung des RESET-Zustandes.
Da wir mit dem U 2716 C über den reichlichen Speicherplatz von 2 kByte verfügen, können 7 verschiedene Lieder nacheinander aufgerufen werden, bis sich der Zyklus mit dem ersten Lied wiederholt. Mit einem einfachen Umschalter lassen sich 7 weitere Melodien aktivieren, so daß unsere Superklingel z. B. in der Weihnachtszeit stimmungsvolle Weihnachtslieder intoniert. Die Software soll so gestaltet sein, wie wir es schon vom LC-80 her kennen. Wir benutzen mit geringfügigen Änderungen sein "Musikkonzept", das in Teil I, Kapitel 2 bereits ausführlich beschrieben wurde.
Die Hardware wird so minimiert, daß wir mit möglichst wenigen Bauelementen auskommen. Dabei muß trotzdem berücksichtigt werden, daß alle im "rauhen Alltag" einer normalen Haustürklingel vorkommenden Störungen (Netzeinbrüche, Störimpulse, Klingelpartien) weitgehend verkraftet werden und aus unserer Klingel keine "Nervensäge" wird.

6.2. Software

Im Gegensatz zu bisherigen Experimenten fangen wir mit der Softwareentwicklung an. Kernstück unseres Musikprogrammes war bisher das im Betriebssystem des LC-80 untergebrachte Unterprogramm MUSIK (04EE). Unsere Melodieklingel wird dieses Betriebssystem nicht besitzen, sondern benötigt ein eigenes. Dennoch wollen wir uns die Regeln beim Musikmachen mit dem LC-80 noch einmal ins Gedächtnis zurückrufen. Soll ein Musikstück gespielt werden, sind zunächst einmal die Noten in einen beliebigen Speicherbereich einzutragen. Danach wird IY mit dem Speicherplatz der ersten Note geladen und dann das Unterprogramm MUSIK aufgerufen.
Genauso werden wir es auch tun. Unser Musikprogramm muß aber bei der Adresse 0000 beginnen, da der Mikroprozessor nach jedem RESET grundsätzlich dort mit der Programmabarbeitung beginnt. Die Noten der verschiedenen Lieder legen wir zweckmäßigerweise auf markante Speicherbereiche. Die Zuordnung der einzelnen Programmteile erfolgt so:

```
0000   MUSIK-Hauptprogramm
0080   1. Lied        ab 00B4 Tonhöhen- und
0100   2. Lied                -längentabelle
0180   3. Lied
0200   4. Lied
0280   5. Lied
0300   6. Lied
0380   7. Lied
```

Das erste kByte endet mit 03FF, es lassen sich also einschließlich Grundprogramm 7 Lieder zu je 80 (Hexa) Bytes unterbringen. Das sind jeweils 64 Töne einschließlich programmierter Pausen pro Lied - es müßte also für normale Ansprüche reichen.
Eine Einschränkung gibt es beim ersten Lied. In dessen Speicherbereich ist noch zusätzlich die Tonhöhen- und -längentabelle, die beim LC-80 im Bereich von 0534 bis 0573 liegt, untergebracht

Damit wird das erste Lied auf die Länge von 32 (Hexa) Bytes reduziert.
So, nun als erstes zum Musikhauptprogramm. Es wurde so überarbeitet, daß überflüssige Funktionen weggelassen wurden, wie z. B. die Möglichkeit der Melodiewiederholung. Andererseits mußten neue Funktionen eingebaut werden wie der Sprung auf Adresse 0038 zum HALT am Programmende. Damit nutzen wir eine RESTART-Funktion des U 880 D aus - findet die CPU nämlich einen Befehl FF (das ist z. B. immer in einem leeren Speicherbereich der Fall), führt sie automatisch einen Sprung auf die Adresse 0038 und von da aus auf 006B aus, wo sie einen HALT-Befehl findet. Dieser Trick sichert ab, daß der definierte HALT-Zustand auch dann erreicht wird, wenn sich die CPU z. B. infolge einer Netzstörung in nicht vorhandenen Speicherbereichen "tummelt". Ein weiterer zusätzlicher Programmteil am Anfang dient der Auswahl der verschiedenen Notenbereiche. Wir wissen, daß der erste bei 0080 beginnt und alle nachfolgenden in einem Abstand von 80 (Hexa) Bytes folgen. Jedes Lied wird mit einem HALT-Befehl abgeschlossen. Der dabei auf L-gehende /HALT-Ausgang bewirkt hardwaremäßig ein L am /RESET-Eingang. Im RESET-Zustand wartet die CPU auf ein "Klingelzeichen", welches /RESET auf H setzt und damit die Programmabarbeitung ab 0000 startet. Da wir keinen externen RAM besitzen und sich nach RESET alle Normalregister in einem undefinierten Zustand befinden, gibt nur der Zustand des statischen IX-Registers (welches die letzte Note des vorher gespielten Liedes enthält) einen Hinweis auf das nächste zu spielende Lied. Der Anfangsteil des Hauptprogrammes inkrementiert IX nun solange bis die 7 niederwertigsten Bits gleich 0 sind. Das ist immer genau dann der Fall, wenn der Wert xx80 oder xx00 erreicht ist (also ein Liedanfang). Außerdem muß dann noch überprüft werden, daß mindestens eines der drei Bits 7, 8 oder 9 gleich 1 ist (wenn nicht, zeigt IX auf den Beginn des Hauptprogrammes, da auch hier die Bedingung xx00 zutrifft). Anschließend werden die höchsten Bits 10 bis 15 auf 0 gesetzt.
Warum benutzen wir eigentlich das IX-Register und nicht, wie der LC-80 das IY-Register zur Musikerzeugung? Ganz einfach,

intern tut's auch der LC-80 mit IX. Da im Betriebssystem aber IX für die Anzeigeprogramme DAK 1 und DAK 2 schon fest vergeben war, wurde das noch freie IY-Register für das Unterprogramm MUSIK benutzt. Die beiden Befehle PUSH IY und POP IX auf den Adressen 04EE und 04F0 bewirken einfach, daß der Inhalt von IY auf IX übertragen wird. In unserem Programm können wir uns das ersparen.

Das soll als Erläuterung zunächst genügen. Die praktische "Herstellung" unseres Klingelprogrammes soll nun in einzelnen Schritten erfolgen:

1. LC-80 ausschalten
   EPROM-Board anstecken
2. LC-80 einschalten (Programmierspannung nicht vergessen)
3. Hauptprogramm ab Adresse 2400 eintragen
4. Tonhöhen- und -längentabelle ab 24B4 eingeben
5. 7 Lieder eintippen
6. alles gründlich kontrollieren
7. EPROM-Ladeprogramm auf Adresse 1000 starten
   EPROM auf Programmierfassung stecken
   ST drücken

Und nun zu den Programmen gemäß den Punkten 3., 4. und 5., die im folgenden aufgelistet sind:

6.2.1. Hauptprogramm

```
2400   DD 23       M1: INC IX              ) IX solange er-
2402   DD F9           LD SP, IX           ) höhen, bis
2404   21 00 00        LD HL, 0000         ) niederwertigste
2407   39              ADD HL, SP          ) 7 Bit = 0
2408   7D              LD A, L             ) (Umweg über
2409   E6 7F           AND 7F              ) Stackpointer
240B   FE 00           CP 00               ) und HL-Register)
240D   20 F1           JRNZ F1        M1   )
```

```
240F   CB 7D          BIT 7, L              Ist Bit 7 des
2411   20 0A          JRNZ 0A       M2      L-Registers
2413   CB 44          BIT 0, H              oder eines der
2415   20 06          JRNZ 06       M2      Bits 0 oder 1
2417   CB 4C          BIT 1, H              des H-Registers
2419   20 02          JRNZ 02       M2      = 1?, wenn nicht,
241B   18 E3          JR E 3        M1      IX weiter erhöhen
241D   DD F9      M2: LD SP, IX             Bits 10 ... 15
241F   21 00 00       LD HL, 0000           des Indexregi-
2422   39             ADD HL, SP            sters IX werden
2423   EB             EX DE, HL             0 gesetzt (Umweg
2424   7A             LD A, D               über SP, HL und
2425   E6 03          AND 03                DE)
2427   57             LD D, A               Jetzt ist ein
2428   DD 21 00 00    LD IX, 0000           Liedanfang im
242C   DD 19          ADD IX, DE            Register IX.
242E   3E 0F          LD A, 0F              PIO (D 206) in
2430   D3 F7          OUT F7                Mode Ausgabe (PB)
2432   DD 7E 00   M3: LD A, (IX + 00)       Tonerzeugungapro-
2435   87             ADD A                 gramm wie LC-80
2436   18 02          JR 02         M4              ↓
2438   18 31          JR 31         M9      RESTART 38→HALT
243A   00         M4: NOP
243B   00             NOP
243C   38 2D          JRC 2D        M9
243E   0E 00          LD C, 00
2440   CB 77          BIT 6, A
2442   20 02          JRNZ 02       M5
2444   CB C9          SET 1, C
2446   E6 3F      M5: AND 3F
2448   21 B4 00       LD HL, 00B4           Tontabelle 00B4
244B   5F             LD E, A
244C   16 00          LD D, 00
244E   19             ADD HL, DE
244F   5E             LD E, M
2450   23             INC HL
2451   56             LD D, M
```

```
2452   DD 23          INC IX
2454   DD 66 00       LD H, (IX + 00)
2457   3E FF          LD A, FF
2459   6A          M6:LD L, D
245A   D3 F5       M7:OUT F5                    PIO-Ausgabe
245C   43             LD B, E
245D   00          M8:NOP
245E   10 FD          DJNZ FD              M8
2460   A9             XOR C
2461   2D             DEC L
2462   20 F6          JRNZ F6              M7
2464   25             DEC H
2465   20 F2          JRNZ F2              M6
2467   DD 23          INC IX
2469   18 C7          JR C7                M3
246B   76          M9:HÄLT
```

6.2.2. Tonhöhen- und -längentabelle

```
24B4   70 18          24CA   3B 2E          24E0   1F 57
24B6   6A 1A          2400   38 31          24E2   1D 5C
24B8   64 1B          24CE   35 33          24E4   1B 62
24BA   5E 1D          24D0   32 37          24E6   1A 67
24BC   59 1E          24D2   2F 3A          24E8   18 6E
24BE   54 20          24D4   2C 3D          24EA   17 74
24C0   4F 22          24D6   2A 41          24EC   16 7B
24C2   4B 24          24D8   27 45          24EE   14 82
24C4   46 26          24DA   25 49          24F0   13 8A
24C6   42 29          24DC   23 4D          24F2   12 92
24C8   3F 2B          24DE   21 52
```

6.2.3. Melodien

Wir wollen folgende Lieder im RAM-Bereich ab 2480 abspeichern:

1. Horch, was kommt von draußen rein ...
2. Auf, auf zum fröhlichen Jagen ...
3. Das Lieben bringt groß' Freud ...
4. Wenn alle Brünnlein fließen ...
5. Hoch auf dem gelben Wagen ...
6. Muß i' denn ...
7. Es blies ein Jäger wohl in sein Horn ...

Die folgende Tabelle enthält die Zusammenstellung der Noten und Pausen mit der jeweiligen Startadresse des Liedes:

Lied Nr	1	2	3	4	5	6	7
Start-adr.	2480	2500	2580	2600	2680	2700	2780
	0B 04	0B 08	0B 08	00 08	0B 08	0B 04	0D 08
	0D 04	10 0B	10 0B	12 07	10 04	0D 04	12 07
	0F 04	0B 04	12 04	20 01	12 04	0F 07	20 01
	10 04	10 08	14 08	12 08	14 08	20 01	12 07
	12 04	14 03	15 04	14 07	17 08	0F 04	20 01
	14 04	20 01	19 04	20 01	14 08	12 04	12 07
	12 08	14 04	17 10	14 08	12 08	10 07	20 01
	10 04	10 10	20 08	16 0B	10 10	20 01	11 04
	0D 04	0B 08	10 04	14 04	0B 08	10 04	0F 04
	16 08	10 04	14 04	12 08	10 04	14 04	0D 10
	12 04	14 04	17 07	16 08	12 04	12 06	0F 10
	0F 04	12 08	20 01	17 08	14 08	14 02	0D 08
	17 08	0B 07	17 08	16 08	12 08	12 04	12 08
	80	20 01	15 07	14 08	10 10	10 04	16 08
		0B 08	20 01	12 08	80	0F 10	19 07
		0D 04	15 08	14 10		12 06	20 01

Nr.	1	2	3	4	5	6	7
		0F 04 10 10 80	14 10 80	80		14 02 12 04 10 04 0F 07 20 01 01 04 12 04 10 07 20 01 10 08 0D 08 12 08 01 10 80	19 10 17 08 19 08 16 12 80

Das war's! Nach einer nochmaligen Kontrolle starten wir das EPROM-Ladeprogramm auf Adresse 1000 (Teil II noch einmal anschauen!) und es erscheint

```
U  2 7 1 6
```

Jetzt kann ein EPROM dieses Typs eingesteckt und die Taste ST betätigt werden. Bei

```
   P A S S
```

können wir auf ein gelungenes Melodieprogramm mit 7 Volksliedern hoffen.

Nur ausprobieren können wir es leider nicht, noch nicht ...

6.3. Das neue Betriebssystem

Keine Angst - wir lassen den LC-80 im Prinzip so wie er ist. Aber wir werden jetzt eine neue und, wie sich zeigen wird, sehr nützliche Eigenschaft an ihm kennenlernen.
Wie wir aus dem Schaltplan, der Bedienungsanleitung und insbesondere dem Handbuch LC-80 wissen, hat der LC-80 insgesamt 5 ROM-Plätze, von denen wir im Grundzustand die oberen 3 nutzen können. Die dazugehörigen Adresse sind:

```
ROM 1   0000 ... 07FF
ROM 2   0800 ... 0FFF
ROM 3   1000 ... 17FF
```

ROM 4 und 5 sind nicht direkt ansprechbar - das gelingt uns erst, wenn an der Stelle B 1 im Stromlaufplan mittels Kleinschalter KSD 11 oder Drahtbrücke L-Pegel an D 209/C geschaltet wird. Damit wir nicht erst lange suchen müssen, sind beim LC-80 diese Anschlüsse bereits herausgeführt worden (oben rechts neben der Betriebsspannungsbuchse). Dort sind 4 Bohrungen für einen solchen DIL-Switch-Schalter angebracht, von denen eine mit U und eine mit M bezeichnet ist. Wird U (USER) mit M (Monitor) verbunden, werden ROM 1 und 2 abgeschaltet und ROM 4 und 5 an deren Stelle aktiviert, also:

```
ROM 4   0000 ... 07FF
ROM 5   0800 ... 0FFF
ROM 3   1000 ... 17FF
```

Genau das haben wir gewollt. Wenn wir auf Position 4 (4. ROM von oben) eine 24polige Fassung sorgfältig einlöten, einen Schalter oder eine Drahtbrücke von U nach M legen und unseren eben programmierten EPROM U 2716 C in diese Fassung stecken, wird beim Einschalten bereits eine Melodie erklingen. Wenn nicht, schalten wir ab und suchen die Fehlerursache, die

hard- oder softwareseitig liegen kann.
Klappt alles, muß am Ende der Melodie die rote HALT-LED leuchten. Beim Betätigen von RES ertönt das nächste Lied; nach der 7. Melodie geht es mit der ersten wieder los.

Das ist schon eine tolle Sache - der Computer (eigentlich ist es ja gar nicht mehr unser LC-80) fängt von ganz allein an, sein Programm abzuarbeiten, stoppt an der richtigen Stelle und wartet auf ein neues Startkommando.

Wem die eben gezeigten Praktiken noch nicht so ganz geheuer vorkommen, kann jetzt seinen EPROM wieder herausziehen (Spannung vorher abschalten!), den Schalter zwischen U und M wieder öffnen, das EPROM-Board anstecken, den LC-80 einschalten und dann den EPROM-Inhalt in den RAM-Speicher ab 2400 einlesen. Nach der Kontrolle, ob alles richtig in den RAM geladen wurde, können dort auf den 7 festgelegten Adressen neue Lieder eingetragen werden. Wie die Melodien entwickelt werden, steht im Teil I, Kapitel 2. Wer will, kann zuvor den ursprünglichen EPROM-Inhalt auf Kassette retten (FILE-Name beliebig, Startadresse 2400, Endadresse 27FF). Wenn neue Melodien im RAM eingetragen wurden, empfiehlt es sich, diese vorher einzeln auszuprobieren, ohne jedesmal einen EPROM laden zu müssen. Auch das ist ganz einfach, wir geben dazu folgendes Kurzprogramm auf Adresse 2000 ein:

```
2000   FD 21 00 25    LD IY, 2500
2004   CD EE 04 *     CALL MUSIK
2007   76             HALT,
```

wenn unser neues Lied auf Adresse 2500 beginnt. Analog gilt das für die anderen festgelegten Liedadressen. Zu beachten ist lediglich, daß die Melodien nicht länger als 80 (Hexa) Bytes werden und insbesondere das Lied ab 2480 nicht länger wird als bis zur Adresse 24B0. Für den Fall, daß uns alle Melodien gefallen, können wir damit einen neuen EPROM laden. Ist nur einer vorhanden, muß er natürlich vorher gelöscht

werden (Band II, Kapitel 8).
Damit haben wir einen neuen Weg beschritten - die Benutzung des LC-80 als (bescheidenes) Entwicklungssystem für eigenständige Funktionseinheiten auf Mikroprozessorbasis. Die Tastatur, das Display und das Monitorprogramm des LC-80 sowie das EPROM-Programmiermodul gestatten die komplette Fertigstellung von Mikrorechnersoftware zur direkten Verwendung in singulären Mikroprozessoranwendungen.

Doch der letzte Schritt, die Hardware für unsere Klingel, muß noch getan werden.

6.4. Hardware

Die komplette Schaltung für eine Mikrorechnerklingel zeigt Bild 17. Dazu noch einige Erläuterungen.
Wie im LC-80 ist der Mikroprozessor U 880 D (es genügt eine Bastlervariante) das Herz des Systems - gesteuert von einem EPROM U 2716 C, der das unter 6.2. beschriebene Programm enthält. Die tonerzeugende PIO des LC-80 wird durch ein D-Flip-Flop V 4013 D des VEB Mikroelektronik "Karl Marx" Erfurt nachgebildet. Dessen Funktion ist dabei mit der der PIO identisch (natürlich nur 1 bit und nur für Ausgabe geeignet!). Das zweite Flip-Flop des V 4013 D erzeugt das /RESET-Signal für die CPU bzw. hebt es auf. Das Letztere wird durch das Schließen des Relaiskontaktes bewirkt, der über ein im Ansprechen um ca. 0,3 s verzögertes Gatter des V 4093 D (4fach HAND mit Schmitt-Trigger-Verhalten) mit nach-folgendem Differenzierglied C 2/R 4 einen Setzimpuls auf den V 4013 D gibt (Q geht auf H, /RESET wird aufgehoben). Damit beginnt die CPU mit der Abarbeitung des Programmes ab Adresse 0000, es erklingt ein Lied. Nach dessen Ende geht der CPU-Ausgang /HALT auf L und damit Ausgang 4 des V 4093 D auf H.

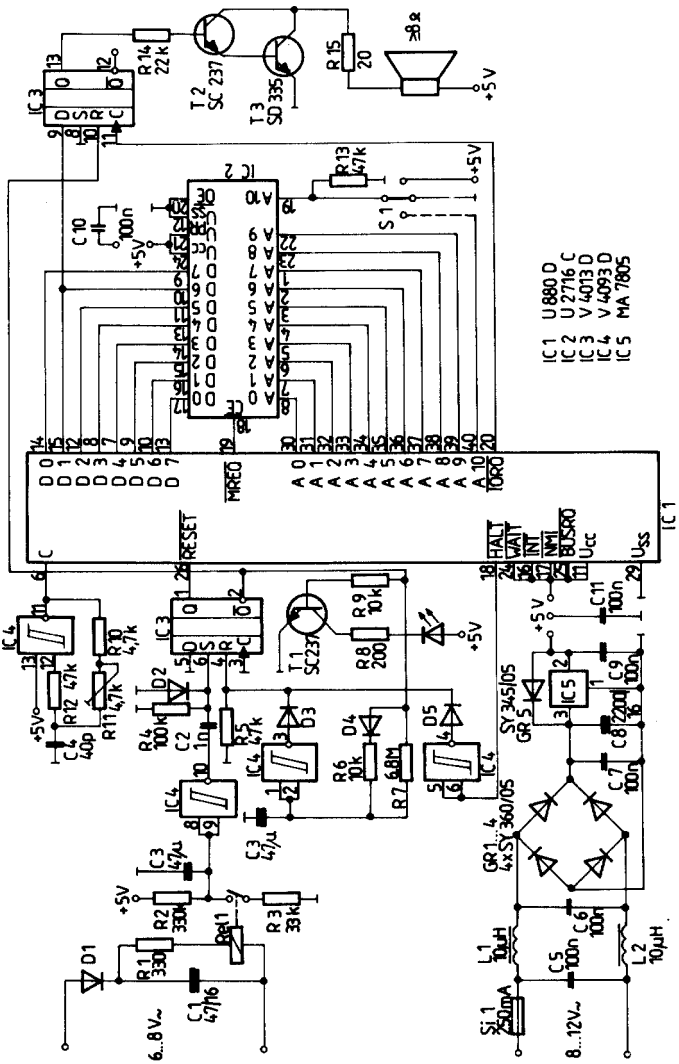


Bild 17: Schaltung des Melodiegenerators

Dies bewirkt über die Diode D 5 ein Rücksetzen des Flip-Flops und damit ein L am /RESET-Eingang des U 880 D, der bis zum erneuten Auslösen des Relais im RESET-Zustand verharrt. Zusätzlich zum "normalen" Rücksetzvorgang der CPU beim Liedende bewirkt das RC-Glied R 7/C 3 am Eingang 1/2 des V 4093 D einen verzögerten RESET-Impuls für den V 4013 D. Dies ist dann notwendig, wenn einmal kein HALT-Signale von der CPU kommen sollte. Der langsame Pegelwechsel an dem RC-Glied R 7/ C 3 kann nur von einem Schaltkreis mit Schmitt-Trigger-Verhalten verarbeitet werden, deshalb auch hier der Einsatz des V 4093 D.

Ein drittes Gatter dieses Schaltkreises bildet einen eleganten Taktgenerator mit nur einem Gatter, der über einen Regelwiderstand auf eine Frequenz von etwa 1 MHz eingestellt wird. Damit sind Tonhöhen und -längen der. Melodien identisch mit den Werten im LC-80.

Anstelle einer üblichen Kontroll-LED für die Betriebsspannung wird in der vorliegenden Schaltung angezeigt, wenn sich die CPU ordnungsgemäß im RESET-Zustand befindet. Sie verlischt nur dann, wenn ein Lied gespielt wird.

Das Netzteil entspricht dem des LC-80, auf die Siebungs- und Entstörmaßnahmen sollte keinesfalls verzichtet werden. Ein integrierter Festspannungsregler ist nicht unbedingt nötig, jedoch sollten die +5V ± 0,25 V genau eingehalten werden. Die Stromaufnahme der Gesamtschaltung beträgt etwa 200 mA. Als Netztrafo sollte nach Möglichkeit ein extern angebrachter Klingeltransformator (vielleicht schon vorhanden?) 8 V/ 1A verwendet werden. Das macht die sonst recht aufwendigen Schutzmaßnahmen (Schutzisolierung, Schutzkontaktausführung o. ä.) überflüssig, da diese Trafos kurzschluß- und berührungssicher ausgeführt und für Dauerbetrieb geeignet sind.

Das Relais wird über eine Gleichrichterdiode und einen Siebelko mit der Wechselspannung (ca. 6 ... 8 V) betrieben, die auch schon die bisherige Klingel versorgt hat. Relais und Vorwiderstand sind so aufeinander abzustimmen, daß das Betätigen des Klingelknopfes sicher zum Auslösen der Computerklingel führt.

6.5. Wenn's Weihnachten wird ...

Im Schaltbild bemerken wir einen Umschalter, der die Adreßleitung A 10 des U 2716 C auf +5 V legen kann.
A 10 ist das höchste Adreßbit des EPROMs - es muß in der Erprobungsphase unserer Klingel auf L (Masse) liegen, da wir ja nur das erste kByte des U 2716 C programmiert haben. Funktioniert die Schaltung sicher, können wir uns an die zweite Hälfte des Speichers wagen und dort ein alternatives Liederrepertoire (z. B. 7 Weihnachtslieder) entstehen lassen. Programmtechnisch geht das so vor sich, daß genauso wie beim ersten Teil nach Anschluß des Programmierboards

- Hauptprogramm (ab 2400)
- Tonhöhen- und –längentabelle (ab 24B4) sowie
- 7 Melodien (2480, 2500, 2580, 2600, 2680, 2700, 2780)

eingetragen werden. Unmusikalische Zeitgenossen finden am Schluß dieses Abschnittes 7 komplette Weihnachtslieder. Natürlich können hier auch eigene Kompositionen geladen werden.
Ist das erfolgt und geprüft worden, wird dieser Teil in den EPROM-Bereich ab 0400 geladen. Wie in Teil II, Kapitel 8 erläutert, müssen wir dabei folgendermaßen vorgehen:

RES , ST	Grundzustand, Startbedingungen
0 , 4 , 0 , 0	Byte-Anzahl 0400
+	
2 , 4 , 0 , 0	RAM-Adresse 2400
+	
0 , 4 , 0 , 0	EPROM-Adresse 0400
RES , ADR	
1 , 0 , 0 , B	EPROM-Ladeprogramm 100B
EX	Anzeige "U 2716"
ST	Programmierung des 2. EPROM-Teils

Na, hat alles geklappt?
Wenn ja, besitzen wir damit eine Klingel mit 2 x 7 Melodiefolgen, die alternativ benutzt werden können. Ganz nach Wunsch können Lieder geändert oder gegen andere ausgetauscht werden.
Eine Besonderheit gibt es bei dem Liedsatz, der im zweiten kByte untergebracht ist. Das komplette Programm läßt sich nicht, wie unter 6.3. beschrieben, als neues Betriebssystem im LC-80 testen. Das liegt daran, daß das Adreßbit A 10 nicht hardwaremäßig auf H gelegt werden kann. Wenn aber der Computer nach dem RESET seine Arbeit beginnt, beginnt er auf Adresse 0000 und damit ist auch A 10 auf L.

Je nach Anwendung des Melodiegenerators (Klingel oder Spieldose) kann es erforderlich werden, die Lautstärke zu verändern. Dies kann entweder durch einen Festwiderstand in Reihe zum Lautsprecher oder durch ein Potentiometer von 50 oder 100 Ohm erreicht werden.

Zum Schluß wie versprochen noch ein Sortiment Weihnachtslieder. Im Gegensatz zu den .Volksliedern werden sie in ganzer Länge angespielt. Wer dies nicht möchte, kann an geeigneter Stelle den Wert 80 (Liedende) eingeben:

1. Alle Jahre wieder ...
2. Laßt uns froh und munter sein ...
3. Leise rieselt der Schnee ...
4. Kling, Glöckchen kling ...
5. Morgen, Kinder, wird's was geben ...
6. Ihr Kinderlein kommet ...
7. Vom Himmel hoch ...

Lied-Nr.	1	2	3	4	5	6	7
Startadr.	2480	2500	2580	2600	2680	2700	2780
	12 0B	12 07	14 0F	17 10	10 08	12 07	17 10
	14 04	20 01	20 01	14 08	0B 08	20 01	16 10
	12 08	12 07	14 08	15 08	0D 08	12 10	14 10
	10 08	20 01	12 0B	17 04	0B 08	0F 08	16 10
	0F 10	12 04	14 04	19 04	0D 04	12 07	12 10
	0D 10	14 04	12 08	17 04	10 04	20 01	14 10
	0B 08	12 04	10 23	19 04	0F 04	12 10	16 10
	0D 04	10 04	20 01	17 10	12 04	0F 08	17 10
	0F 04	0F 07	10 10	15 10	10 08	12 08	20 08
	10 08	20 01	0D 08	12 08	0B 08	10 10	17 07
	0F 08	0F 07	10 0B	17 08	14 07	0D 08	20 01
	0D 20	20 01	0F 04	14 18	20 01	10 08	17 10
	0F 08	0F 10	0D 08	20 08	14 04	0F 18	12 0F
	12 08	10 07	0B 24	12 07	15 04	12 07	20 01
	14 08	20 01	12 0B	20 01	17 08	20 01	12 10
	12 08	10 07	11 04	12 08	14 08	12 10	0F 10
	17 10	20 01	12 08	14 08	15 08	0F 08	12 10
	16 08	10 04	15 08	10 08	14 08	12 07	10 10
	14 08	12 04	14 08	14 10	12 10	20 01	0F 10
	12 08	10 04	12 08	12 10	10 08	12 10	20 08
	10 04	0F 04	10 24	15 07	0B 08	0F 08	0F 08
	0F 04	0D 07	12 0B	20 01	0D 08	12 08	14 0F
	10 08	20 01	0D 03	15 08	0B 08	10 10	20 01
	12 08	0D 07	20 01	17 08	0D 04	0D 08	14 10
	0F 18	20 01	0D 08	12 08	10 04	10 08	12 10
	80	0D 10	0F 08	15 10	0F 04	0F 17	16 10
		0B 08	0D 08	14 10	12 04	20 01	17 10
		0D 08	0F 08	12 07	10 08	0F 08	14 10
		0F 08	10 24	20 01	0B 08	0D 0F	12 10
		10 08	80	12 08	14 07	20 01	20 08
		12 04		14 08	20 01	0D 07	17 08

Lied-Nr.	1	2	3	4	5	6	7
		14 04 12 04 14 04 12 10 17 08 12 07 20 01 12 04 14 04 12 04 10 04 0F 08 05 08 12 10 17 08 12 07 20 01 12 04 14 04 12 04 10 04 0F 08 0D 08 0B 10 80		16 08 17 10 12 10 14 08 19 08 17 08 16 08 19 10 17 0F 20 01 17 10 14 08 15 08 17 04 19 04 17 04 19 04 17 10 15 10 12 08 17 08 14 18 80	14 04 15 04 17 08 14 08 15 08 14 08 12 10 15 07 20 01 15 08 19 07 20 01 19 08 12 07 20 01 12 08 17 10 10 07 20 01 10 08 15 07 20 01 15 08 14 04 12 04 10 04 0F 04 10 10 80	20 01 0D 08 10 0F 20 01 10 07 20 01 10 08 0F 0F 20 01 0F 07 20 01 0F 08 14 17 20 01 14 08 12 0F 20 01 12 07 20 01 12 08 17 10 12 08 0F 08 10 10 0D 08 0A 08 0B 18 80	16 10 14 10 12 10 14 10 10 08 0F 08 0D 10 0B 10 80

6.6. Aussichten

An einem einfachen Beispiel haben wir gelernt, mit Hilfe des LC-80 eigenständige Mikrorechnerlösungen zu entwickeln. Dabei lassen sich Software und Hardware schon vorab so optimieren, daß am fertigen Gerät kaum noch böse Überraschungen zu erwarten sind. Unser Modell hat. schon um die Erfolgsaussichten beim Nachbau günstig zu gestalten, naturgemäß einen bescheidenen Umfang. Sollen für andere Geräte nur Ausgabefunktionen realisiert werden, kann das hier vorgestellte Prinzip analog weiterverfolgt werden. Für wenige Kanäle (bei der Klingel nur Datenbit 1) reichen weiterhin D-Flip-Flops, ansonsten können die schon bekannten Schaltkreise V 4042 D (4 D-Flip-Flops in einem IC) verwendet werden. Komplizierter wird es, wenn wir Eingabekanäle benötigen. Das läßt sich nur realisieren, wenn Bauelemente mit Tristate-Eigenschaften eingesetzt werden. Diese beteiligen sich nur am Datenverkehr, wenn die CPU es wünscht. Die komplette Schaltungstechnik zu diesem Problemkreis würde den Rahmen dieses Kapitels jedoch sprengen, so daß darauf verzichtet wird. Bei der Konzeption von kleineren mikroprozessorgesteuerten Geräten sollte in jedem Fall geprüft werden, ob nicht der Einsatz einer PIO lohnt. Deren frei Programmierbarkeit und die zur Verfügung stehenden 16 Kanäle für Ein- und Ausgabefunktionen lassen sie an nahezu alle Probleme mühelos anpassen. Bei vielen Mikrorechnerschaltungen kann nicht auf RAMs verzichtet werden. Diese sollten dann möglichst genauso wie im LC-80 adressiert werden, um letzteren zur Softwareerprobung nutzen zu können.

Und was können wir als nächstes entwickeln? Hier einige Vorschläge:

- Kfz-Bordcomputer mit Drehzahlmesser, Tankinhaltsanzeige, Temperaturfühler für Motor-, Innen- und

Außentemperatur, Bordspannungskontrolle, Borduhr ...

- Quarzgesteuerte Schaltuhr mit vielfältigen Schaltfunktionen (Wecken, Fische füttern, Musikaufnahmen usw.)
- Computer-Lichtorgel mit tollen Lichteffekten
- Computer-Funkuhr für die Auswertung von kodierten Zeitzeicheninformationen
- Code-Schloß mit personengebundener Einlaßkontrolle (Eingabe der Codenummer z. B. mit Tastatur oder Lochkarte gemäß Kapitel 2)
- Miniorgel mit entsprechender Tastatur
- Alarmanlage mit Meldung über den Ort des Alarmes

7. Ein "Tor zur Welt"

In all unseren bisherigen Experimenten genügte uns der USER-Bus zur Herstellung von Verbindungen zwischen LC-80 und Peripherie. Es kann jedoch der Fall eintreten, daß z. B. 12 PIO-Anschlüsse zu wenig sind, denken wir nur an unseren IC-Tester von Kapitel 9 im Teil I. Dort konnten wir aus diesem Grund nur 14polige Logikschaltkreise prüfen.
Die Konzeption des LC-80 berücksichtigt diese Erweiterungswünsche mit dem komplett herausgeführten CPU-Bus (Steckverbinder X 2). Das ist übrigens ein Merkmal, welches praktisch alle Home- und Personalcomputer des In- und Auslandes aufweisen - sie unterscheiden sich lediglich in der Steckverbinderform und -anschlußbelegung.

In diesem Kapitel werden wir im wesentlichen zwei Dinge tun:

1. eine zusätzliche PIO anschließen und
2. eine für spätere Erweiterungen unbedingt erforderliche Bustreiberstufe aufbauen.

7.1. Die dritte PIO

Die einfachste und für die meisten LC-80-Besitzer wahrscheinlich nützlichste Erweiterung ist eine dritte PIO mit insgesamt 16 zusätzlichen Ein- bzw. Ausgabekanälen.
Einfach wird die Sache deshalb, weil die zu erstellende Leiterplatte nur einen einzigen Schaltkreis enthält - eben diese PIO. Dieses Bauelement benötigt nicht allzuviel Strom aus dem +5 V-Versorgungssystem des LC-80 und auch keine großen Steuerleistungen (wegen der nMOS-Technologie der Mikrorechnerschaltkreise und der niedrigen Taktfrequenz des LC-80) aus dem CPU-

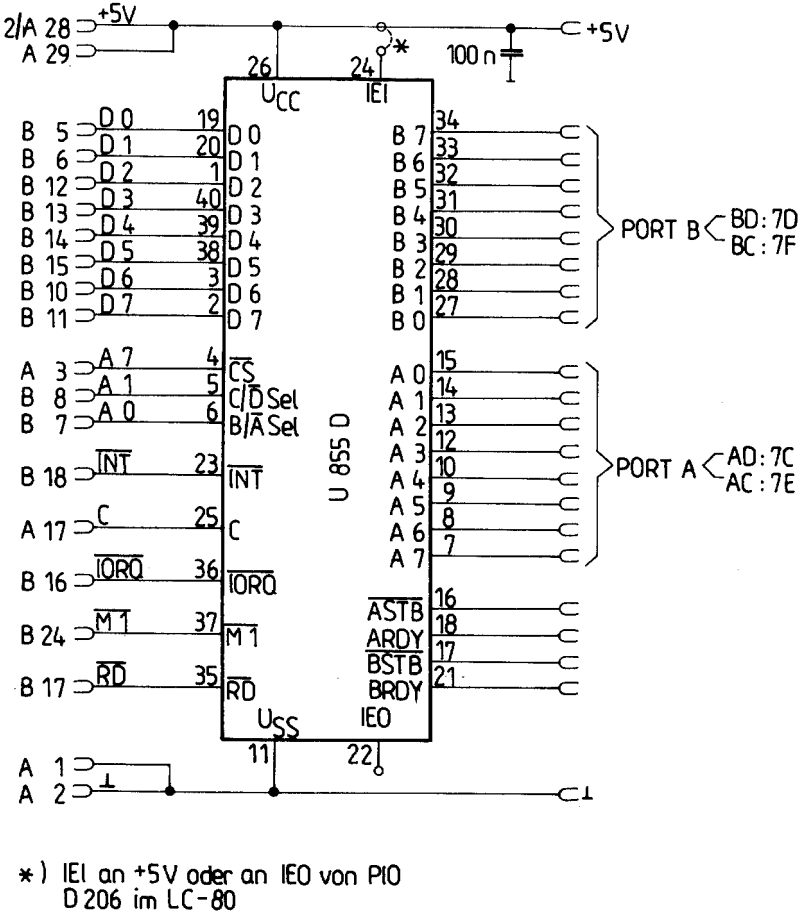
Bus, so daß man zunächst auf ein Bustreibersystem und eigene Stromversorgung verzichten kann. Unsere vorbereitenden Aktivitäten am LC-80 beschränken sich daher auf das Einlöten einer Drahtbrücke zwischen den Steckverbinderanschlüssen X 2/A 28/ A 29 und der oberhalb dieses Steckverbinders ankommenden +5 V-Leitung. Damit bekommt auch der CPU-Bus einen Betriebsspannungsanschluß.

Nun zur PIO-Erweiterungsleiterplatte. Wir bauen sie nach dem Schaltbild in Bild 18 auf.

Die Ankopplung des Datenbusses erfolgt ebenfalls ohne Besonderheiten. Auch ohne Probleme läßt sich die Adressierung der Zusatz-PIO bewerkstelligen. Bei der gewählten Beschaltung ergeben sich folgende Adressen:

Steuerkommandos an Port A (AC):	7E
Steuerkommandos an Port B (BC):	7F
Datenübergabe am Port A (AD):	7C
Datenübergabe am Port B (BD):	7D

Das RESET-Problem der PIOs (der--U 855 D hat ein Pin "zuwenig", so daß kein echter RESET-Anschluß existiert) wird im LC-80 so gelöst, daß mittels logischer Verknüpfungen ein modifiziertes /M1-Signal an die PIOs gelegt wird (Handbuch LC-80)
Dieses neue /M1-Signal wird glücklicherweise auf den CPU-Bus geführt und ermöglicht damit eine ordnungsgemäße RESET-Funktion auch der Zusatz-PIO.
Ein echtes Problem stellt die Interruptbeschaltung dar. Leider ist die Prioritätskette der drei Schaltkreise D 206, D 207 und D 208 nicht ohne weiteres zu verlängern, da IEO der PIO D 206 nicht auf den CPU-Bus geschaltet worden ist. Soll die neue PIO in diese Kette einbezogen werden, ist eine Verbindung zwischen IEO der PIO D 206 und IEI der externen PIO zu schalten (evtl. freie Steckkontakte von X 2 benutzen). Wem das alles zu kompliziert ist, sorgt softwaremäßig dafür, daß nicht gleichzeitig Interrupts von PIO 3 und den inneren Schaltkreisen des LC-80 erzeugt werden können.



*) IEI an +5V oder an IEO von PIO D 206 im LC-80

Bild 18: Die Anschaltung der PIO

Die beiden Ports A und B, die 4 Handshakesignale sowie Masse und Betriebsspannung von PIO 3 werden zweckmäßigerweise als Buchsenleiste nach außen geführt. Sehr gut geeignet dafür ist das Stecksystem, welches die Rundfunk- und Fernsehindustrie für ihre Steckmodule einsetzt. Dieses Konzept mit 5 mm-Rastermaß ist für die Leiterplattenkonstruktion sehr günstig, erlaubt auch die Teilbelegung dieser Anschlüsse (z. B. Port A und B getrennt) und gestattet auf einfache Weise auch Anschlüsse mit Krokodilklemmen und anderen Varianten.

Und was können wir an die PIO-Ports anschließen?

- 16 Leuchtdioden mit Treibern als Kontrolleinrichtung der PIO-Ports oder als 16bit-Analysator für die Darstellung von Registerinhalten

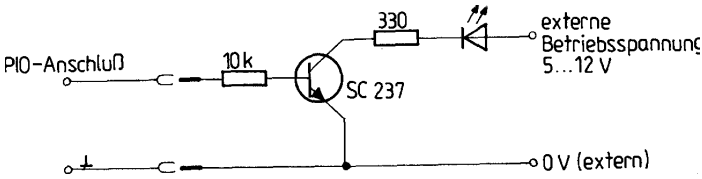


Bild 19: Schaltung für 1 bit

- zwei 8bit-D/A-Wandler gemäß Teil I, Kapitel 13 zur Ansteuerung eines Oszillographen oder zur Realisierung einstellbarer Spannungsquellen und Funktionsgeneratoren
- eine externe Tastatur zur Realisierung alphanumerischer Eingaben (z. B. mit 64 Tasten) oder als Manual für eine Computer-Orgel
- ein Steuersystem für Modellbahnanlagen
- einen erweiterten Schaltkreistester gemäß Teil I, Kapitel 9 (zusammen mit dem USER-Bus können dann 28 Pins beschaltet werden.)

7.2. Ein Bus mit "Nachbrenner"

Will man den LC-80 zu komplexeren Steueraufgaben heranziehen (z. B. in der Kleinrationalisierung) oder größere Speichererweiterungen realisieren, gibt es mehrere Probleme:

- die Stromversorgung des LC-80 ist nahezu "ausgereizt" und verträgt wenig Zusatzlast,
- die CPU-Signale werden gleichstrommäßig von allen nicht in MOS- bzw. CMOS-Technologie hergestellten Schaltkreisen belastet - weitere Verbraucher können leicht zu Fehlfunktionen führen und
- alle Leitungen und Bauelemente stellen über ihre Eingangskapazitäten eine dynamische Belastung dar, die ebenfalls zu Störungen führen kann, wenn es "zuviel" wird.

Mit all diesen Problemen werden wir fertig, wenn wir eine Zusatzeinheit aufbauen, die eine zweite Stromversorgung und ein Bustreibersystem mit einer entsprechenden Logik enthält. Dabei ist es zweckmäßig, zwei gleichgroße Leiterplatten "huckepack" aufeinanderzustapeln, von denen die untere das Bustreibersystem und die obere das Zusatznetzteil mit Kühlkörper aufnimmt. Wir beginnen mit dem Netzteil.

7.2.1. Zusatznetzteil

Diese Baugruppe wäre auch für andere Bereiche nutzbar und wird aus diesem Grund völlig separat aufgebaut.
Schaltungstechnisch sieht sie genauso aus wie die des LC-80, also Steckbuchse (für 10 ... 12 V Wechsel- oder Gleichspannung), 2 Drosseln 10 µH, C für HF-Entstörung 10 nF, Grätzgleichrichter, Ladekondensator 2200 µF/16 V und dann der integrierte Spannungsregler (entweder B 3170 V mit entsprechender Widerstandsbeschaltung oder MA 7805). Keinesfalls darf

auf die Schutzdiode (im LC-80 heißt sie V 217) verzichtet werden. Auch alle angegebenen Kondensatoren sind wichtig und sollten nicht eingespart werden. Der Kühlkörper ist großzügig zu bemessen, die Größe des im LC-80 benutzten kann als Minimum betrachtet werden. Ebenfalls wichtig ist auch, daß alle strombelasteten Leitungen genügend breit ausgelegt werden - dies ist z. B. auch für die Kühlung der Gleichrichterdioden günstig.
Die Wechselspannungsversorgung kann mit einem zweiten Netzmodul für den LC-80 aus dem VEB Mikroelektronik "Karl Marx" Erfurt realisiert werden, dabei umgeht man Gehäuse- und Schutzgüteprobleme.

7.2.2. Bustreibersystem

Die Signale des Bussystems kann man in drei Gruppen einteilen, die sich in ihrer Wirkrichtung unterscheiden:

- Ausgangssignale
  (z. B. Adressen: LC-80 → Peripherie)
- bidirektionale Signale
  (z. B. Daten: LC-80 ↔ Peripherie)
- Eingangssignale
  (z. B. /WAIT: LC-80 ← Peripherie)

Da nicht alle Signale eine Verstärkung benötigen, kommen wir mit vier modernen Treiberschaltkreisen der Schottky-Interface-Reihe des VEB Halbleiterwerk Frankfurt/Oder aus. Die Schaltkreise DS 8282 D und DS 8286 D haben genau die Eigenschaften, die wir brauchen. Sie beinhalten einmal gleich 8 Treiberstufen, lassen infolge ihrer quasi direkten inneren Verschaltung eine einfache Leiterplattenkonstruktion zu und liefern nichtinvertierte Ausgangssignale. Der DS 8286 D ermöglicht einen bidirektionalen Datenverkehr. Diese und weitere positive Eigenschaften haben natürlich auch einen Preis - die ICs benötigen jeder bis zu 160 mA Betriebsstrom. Zusätzlich ergeben

sich noch erhebliche Umschaltspitzen. Dies macht eine sorgfältige und ausreichend breite Auslegung der Masse- und Betriebsspannungsleiterbahnen auf der Platine erforderlich. Außerdem sollte jeder der 4 Schaltkreise mit einem 47 µF-Elko und einem 47 nF-Scheibenkondensator so kurz wie möglich abgeblockt werden.
Die Betriebsspannung (+5 V) wird aus unserem Zusatznetzteil (Abschnitt 7.2.1.) bereitgestellt. Damit lassen sich auch weitere externe Module betreiben, die dann über X 3/A 28/ A 29 ihre Betriebsspannung erhalten. Die Massen von LC-80 (X 2/A 1/A 2) und Zusatznetzteil müssen unbedingt auf der Bustreiberleiterplatte verbunden werden. X 3 ist der Steckverbinder, an den die Zusatzmodule angesteckt werden. Er ist genauso verdrahtet und hat konstruktiv dieselbe Ausführung wie der am LC-80 vorhandene X 2-Steckverbinder. Die eigentliche Bustreiberschaltung finden wir in Bild 20. Die drei Schaltkreise DS 8282 D (IC 1 ... 3) arbeiten generell nur in Richtung LC 80 → Peripherie. Dagegen sind die Daten über den DS 8286 D in ihrer Richtung durch das DIR-Signal (Pin 11) umschaltbar. Die Erzeugung des DIR-Signales erfordert eine Logikschaltung, die Bild 21 zeigt. Wann und wie wird die Richtung des Datenverkehrs umgeschaltet?
Würden wir den Bustreiber DS 8286 D direkt an der CPU unterbringen können, wäre alles ganz einfach: nur Leseoperationen (RD = L) würden die Datenrichtung B → A (Peripherie → CPU) erfordern. Im fertigen Gerät LC-80 ist es komplizierter: Datenrichtung B → A heißt jetzt, daß externe Speicher oder I/O-Glieder (außerhalb des Steckverbinders X 3) gelesen werden sollen. Dies ist infolge der unvollständigen Speicherdekodierung und der fehlenden I/O-Dekodierung nur über Umwege zu erreichen. Wir machen uns zwei Umstände zunutze:

- es gibt ein MBDI-Signal, das die komplette Abschaltung aller inneren Speicher des LC-80 ermöglicht (MEDI = L),
- die Adressen A 14 und A 15 werden nicht zur inneren Speicherauswahl herangezogen.

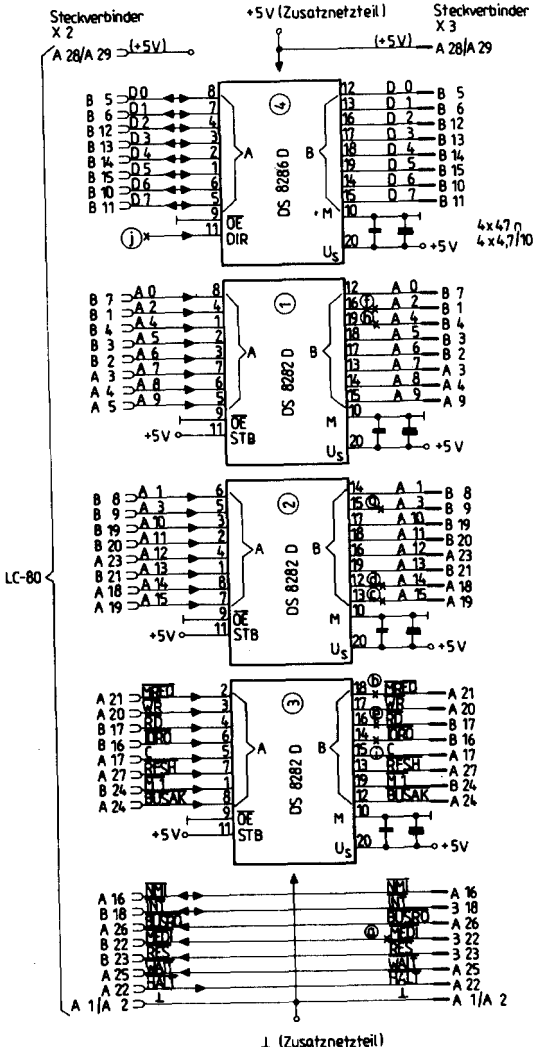
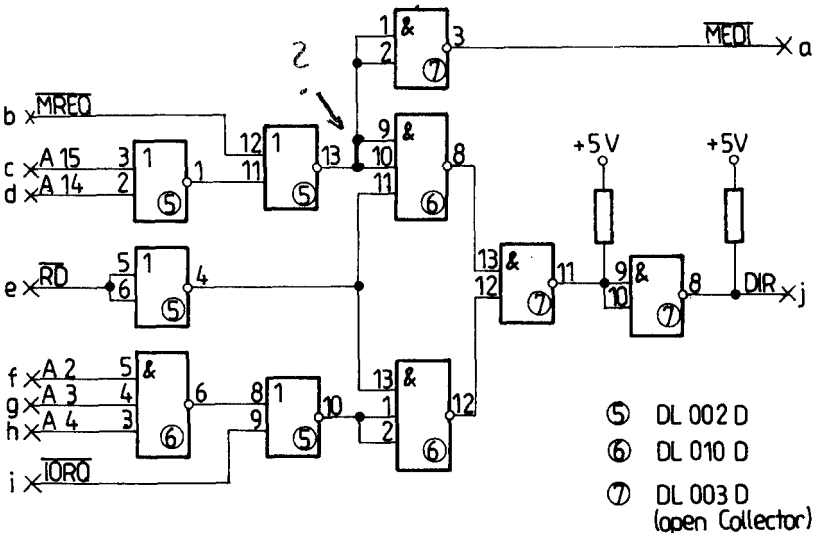


Bild 20: Bustreiber



Datenrichtung für DS 8286 D:

L an DIR ⇒	B → A, wenn	MREQ = L und	(externer
		RD = L und	Speicher-
	A 14 oder A 15 = H		zugriff)
	oder IORQ = L und		(externe
		RD = L und	I/O-
	A 2, A 3 und A 4 = H		Operationen)

Bild 21: Steuerung für Bustreiber (Logikbaugruppe)

Die Kombination beider Eigenschaften benutzen wir, um mittels einiger logischer Verknüpfungen den inneren und äußeren Speicherbereich zu trennen und damit ein Gegeneinanderarbeiten über den Datentreiber zu verhindern.
Der obere Teil dieser Logikbaugruppe hat dabei folgende Wirkungen:

- Die Signale /MREQ, R/D, A 14 und A 15 schalten die Datentransportrichtung des DS 8286 D (IC 4) auf B → A (LC-80 ← Peripherie), wenn:
  /MREQ = L (Speicherzugriff)
  u. /RD = L (Leseoperation)
  u. A 14 oder A 15 (oder beide) = H (externer Speicher angesprochen)
- /MREQ = L (Speicherzugriff) und A 14 oder A 15 (oder beide) = H (externer Speicher angesprochen) bewirken ein L an /MEDI und damit die komplette Abschaltung aller internen Speicher (ROM und RAM).
  Der Schaltkreis IC 7 (DL 003 D) hat Ausgangsstufen mit offenem Kollektor, daher ist es möglich, an seinem Ausgang Y 1 (Pin 3) das /MEDI-Signal auch über die Peripherieschaltung selbst auf L zu bringen und damit alle inneren Speicher abzuschalten.

Der untere Teil der Logikbaugruppe hat ähnliche Funktionen, hier wird zwischen inneren und äußeren I/O-Gliedern unterschieden:

- Die Signale /IORQ, /RD, A 2, A 3 und A 4 schalten die Datentransportrichtung des DS 8286 D auf B → A (LC-80 ← Peripherie), wenn:
  IORQ = L (I/O-Glied angesprochen)
  RD = L (Leseoperation)
  A 2, A 3 und A 4 = H (keiner der im LC-80 vorhandenen PIO- bzw. CTC-schaltkreise ist angesprochen)

Damit steht auch fest, welche Adressen externe Baugruppen, die wir über den Bustreiber anschließen wollen, haben dürfen:

- Speicher, gleichgültig, ob ROM oder RAM, müssen so adressiert und dekodiert werden, daß mindestens eins der beiden Adreßbits ,A 14 oder A 15 H-Pegel führt, also alle Adresse von 4000 bis FFFF.

- I/O-Baugruppen (PIO, CTC o. a.) müssen so adressiert werden, daß alle drei Adreßbits A 2, A 3 und A 4 H-Pegel führen. Dabei ist sowohl die direkte Adressierung (die Bits A 5, A 6 oder A 7 sind jeweils direkt mit CE einer externen I/O-Schaltung verbunden) als auch die Auswahl über einen 1 aus 8-Dekoder (DS 8205 D) erlaubt.

Wenn wir nun diese Logikbaugruppe und die vier Bustreiberschaltkreise auf einer Leiterplatte unterbringen, die Anschlüsse (a ... j) der Logik mit denen der gleichen Bezeichnung auf der Bustreiberschaltung verbinden und das Zusatznetzteil gemäß 7.2.1. anschließen, besitzen wir eine leistungsfähige Treibereinheit für unseren LC-80. Sie ermöglicht uns den nahezu unbegrenzten Ausbau des Mikrorechnersystems und die Anpassung an viele Aufgabenstellungen des Hobby-, aber auch des Rationalisierungsbereiches.
Mit dieser letzten Anstrengung im Rahmen des dritten Bandes haben wir uns alle Möglichkeiten geschaffen, den LC-80 über den ursprünglich vorgesehenen Zweck (Lerncomputer) hinaus nutzbringend anzuwenden.

8. Schlußkapitel

Damit sind wir am Ende unserer Exkursion in die Anwendungsbeispiele des LC-80 angelangt. Keinesfalls am Ende sind wir aber mit den Möglichkeiten, die unser Kleinstcomputer bietet. Wir haben versucht, typische Einsatzfälle mit einem Minimum an Hard- und Softwarekenntnissen zu realisieren und uns sogar mit einer eigenständigen Mikroprozessoranwendung vom LC-80 gelöst.
Natürlich sind unsere Kenntnisse der Programmierung in Maschinensprache noch sehr lückenhaft und auch über die Fähigkeiten der Bausteine unseres Mikrorechners (PIO, CTC, CPU) wissen wir noch lange nicht genug. Aber motiviert sind wir und haben Mut zum Experimentieren auf diesem für viele Hobbyelektroniker neuen Gebiet. Je nach Veranlagung werden wir in Maschinensprache weitere Programme erstellen, neue Zusatzbaugruppen basteln oder in höhere Programmiersprachen einsteigen. Glücklicherweise ist mit dem Computerbausatz Z 1013 vom VEB Robotron Elektronik Riesa ein neues Gerät auf dem Markt, das sowohl für Maschinen- bzw. Assemblerprogrammierung als auch für BASIC-Anwendungen in Verbindung mit einem Fernsehgerät als Display geeignet ist.
Die materiellen Voraussetzungen für ein weiteres Vertiefen der Kenntnisse auf dem Gebiet der Mikrorechner sind also da. Es bleibt zu hoffen, daß die in den vorliegenden Anwendungshinweisen für den LC-80 gemachten Erfahrungen den Amateuren Spaß, den Lernenden Wissen und den Rationalisatoren Gewinn gebracht haben. Damit wäre die Aufgabe dieser Broschürenreihe zum LC-80 erfüllt.

9. Literaturverzeichnis

/1/ Zielosko, G.:
Hinweise zur Anwendung des Lerncomputers LC-80
VEB Mikroelektronik "Karl Marx" Erfurt - Stammbetrieb

/2/ Zielosko, G.:
Anwendung des Lerncomputers LC-80 (2)
Programmierung und Löschung von EPROMs U 2716 C
VEB Mikroelektronik "Karl Marx" Erfurt - Stammbetrieb

/3/ Interface-IS, Dekoder-IS
Heft 19 der Reihe "Information und Applikation Mikroelektronik"
KdT Frankfurt/Oder, VEB Halbleiterwerk Frankfurt/Oder

/4/ Handbuch LC-80
VEB Mikroelektronik "Karl Marx" Erfurt - Stammbetrieb

/5/ Monitorprogramm LC-80
VEB Mikroelektronik "Karl Marx" Erfurt - Stammbetrieb

RFT

veb mikroelektronik
›karl marx‹ erfurt
stammbetrieb

DDR-5023 Erfurt, Rudolfstraße 47
Telefon: 5 80, Telex: 061306

elektronik
export·import

Volkseigener Außenhandelsbetrieb der
Deutschen Demokratischen Republik
DDR - 1026 Berlin, Alexanderplatz 6
Telex: BLN 114721 elei, Telefon: 2180