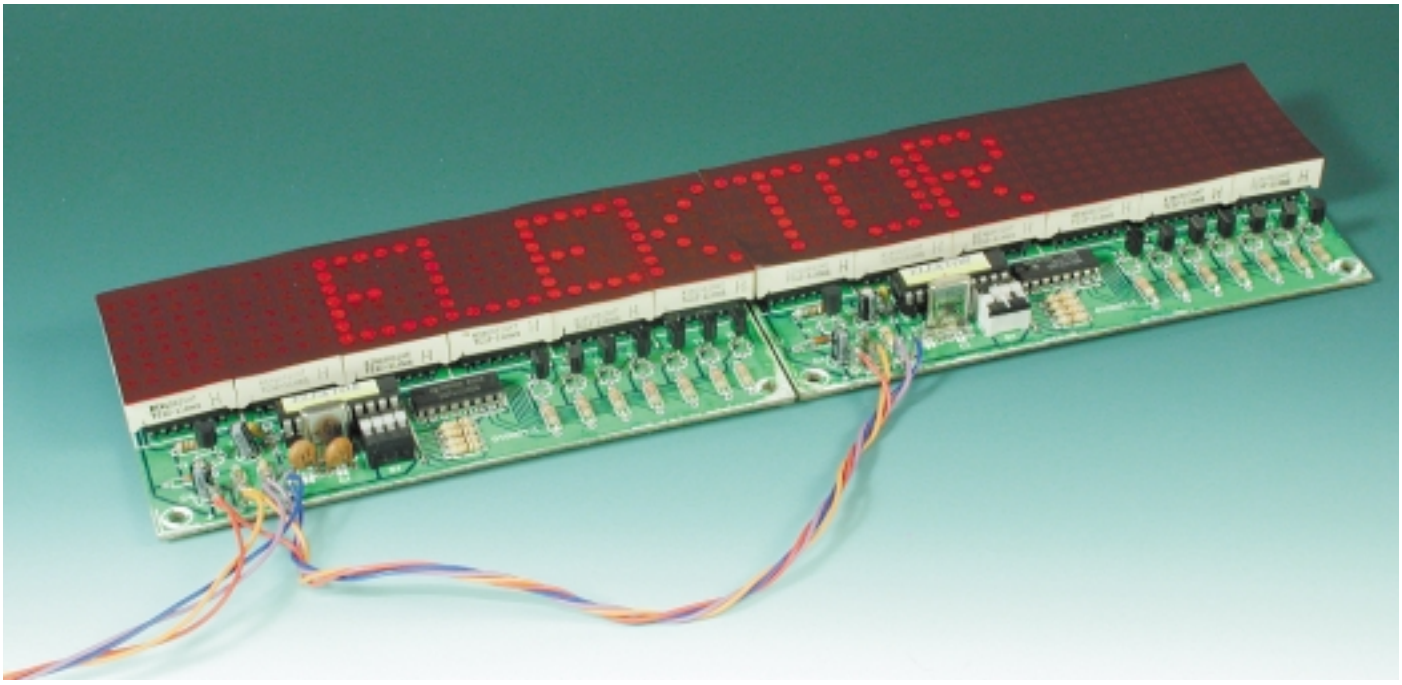
Modulare Punktmatrix-Anzeige

Ansteuerung über den COM-Port

Von A. Köhler

Industriell gefertigte Punktmatrix-Anzeigen sind in der Regel teuer und recht unflexibel, was die Anzahl der anzuzeigenden Stellen oder den verfügbaren Zeichensatz betrifft. Grund genug, zum Lötkolben zu greifen und eine auf die jeweilige Anwendung spezialisierte Punktmatrix-Anzeige selbst herzustellen.



Die hier vorgestellte modulare Punktmatrix-Anzeige besitzt sechs Anzeigestellen, es können ohne Änderung der Hardware bis zu 16 Module kaskadiert, die Anzeige also auf maximal 96 Stellen ausgebaut werden. Die Platine des Moduls ist für 5x7-Matrixdisplays mit einem Anschlussabstand von 33,0 mm (30 mm Zeichenhöhe) vorbereitet, über entsprechende Adapterplatinen können aber beliebig große (oder kleine) Matrixanzeigen oder sogar aus einzelnen LEDs bestehende Arrays betrieben werden, wenn diese elektrisch einigermaßen kompatibel sind.

Die Kosten der Anzeige sind gegenüber industriellen Produkten gering (hier lohnt sich der Selbstbau!). Die eingesetzten Logik-ICs sind preiswerte Standard-Produkte, die wenigen passiven Bauteile Pfennigartikel und selbst der eingesetzte Mikro-

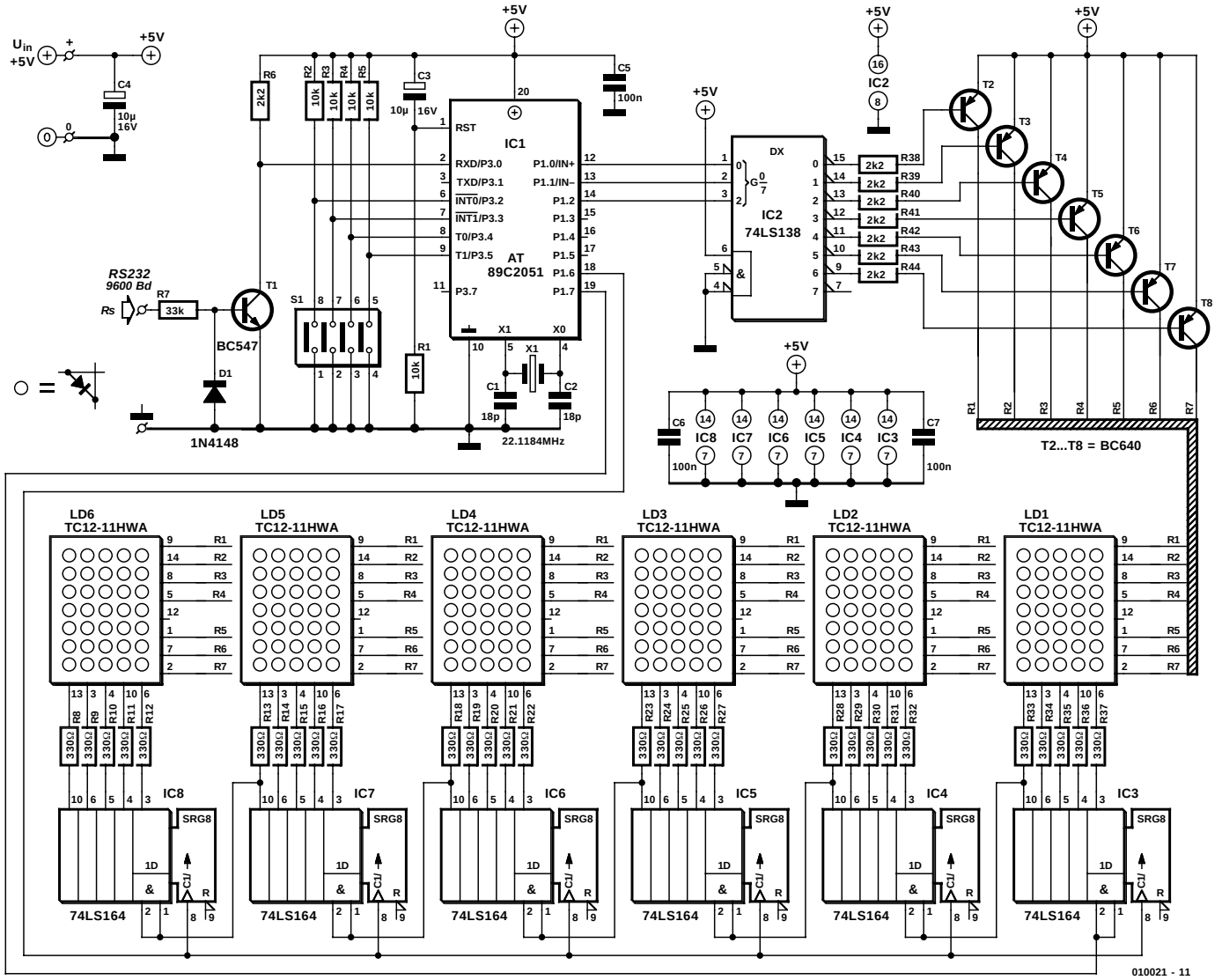


Bild 1. Die übersichtliche Schaltung eines Punktmatrix-Moduls.

controller, ein AT89C2051 mit Flash-Speicher von Atmel ist für ein paar DM zu haben. Ansonsten sind die Kosten weitgehend von Anzahl und Größe der Matrix-Displays abhängig. Der Controller, der die Anzeige des Moduls steuert, verfügt über einen Zeichensatz, der 95 Zeichen umfasst. Dazu gehören neben Ziffern und Buchstaben (in Klein- und Großschreibung) auch zahlreiche Sonderzeichen. Es ist aber - selbst für Assembler-Unkundige - leicht möglich, den Zeichensatz den eigenen Vorstellungen anzupassen und sogar einfache Graphiken darzustellen.

Ansteuerung in Multiplex

Eine Anzeigestelle wird üblicherweise aus einer 5x7- oder 8x8-Matrix aufgebaut. Rechnet man pro Leuchtdiode einen Strom von 10 mA, so benötigt eine Anzeigestelle maximal 350 mA respektive 640 mA. Schon mit ein paar Stellen würde die Matrix-Anzeige ein gewaltiges Netzteil erfordern, außerdem einen hohen Aufwand an Ansteuerleitungen und Vorwiderständen für die Leuchtdioden. Aus diesem Grund wird eine Multiplextechnik angewendet. Dabei lädt der Ansteuerschaltkreis den Inhalt einer Zeile der Matrix in mehrere Latchregister oder ein Schieberegister. Anschließend wird die gesamte Zeile über einen Treibertransistor für kurze zeit hellgesteuert. Danach stellt der Ansteuerschaltkreis die Informationen für die zweite Zeile in den Registern bereit und aktiviert diese. Dieser Vorgang setzt sich bis zur siebten oder achten Zeile fort und wird zyklisch wiederholt.

Erfolgt dieser Vorgang schnell genug, so entsteht für das Auge der Eindruck, als ob die Leuchtdioden gleichzeitig leuchten. Ein Problem ist dabei allerdings die Helligkeit. Je nach Anzahl der Zeilen leuchtet eine Zeile nur 1/7 oder 1/8 der gesamten Zeit, in der Praxis sogar noch kürzer. In der Zeit, in der die Informationen für die Zeile getauscht werden, müssen die LED nämlich auch dunkel bleiben. Damit die Anzeige dennoch in voller Pracht erstrahlt, wird jede Zeile mit einem Vielfachen des zulässigen Durchlass-Stroms angesteuert.

In der hier vorgestellten Schaltung in **Bild 1** werden die Spalteninformationen durch ein aus sechs Einzel-ICs des Typs 74LS164 bestehendes Schieberegister bereitgestellt. Von jedem Schieberegister werden dabei nur fünf Bit genutzt. Die sieben Zeilen selektiert der Con-

troller (P1.0 bis P1.2) über einen 1-aus-8-Dekoder 74LS138 (IC2). Für die Ansteuerung der Schieberegister werden nur zwei Portleitungen des Mikrocontrollers benötigt. Port 1.7 ist die Datenleitung, über P1.6 gelangt der Takt zum Schieberegister. Der Reseteingang des Schieberegisters wird nicht angesteuert, eine Nutzung entspräche der Aktivierung aller LED.
Der Anzeigevorgang beginnt mit der Aktivierung der nicht vorhandenen Zeile 8. Dadurch ist sichergestellt, dass der Betrachter den folgenden Aufbau der Anzeige nicht sieht. Ansonsten würden alle Dioden mit geringer Helligkeit leuchten. Der Mikrocontroller holt das darzustellende Zeichen aus seinem internen Speicher und adressiert ein Muster im Zeichengenerator. Von dort holt der Controller das erste Byte und zerlegt es in einzelne Bits. Je nachdem, ob die betreffende LED leuchten soll oder nicht, wird ein Low (LED leuchtet) oder ein High (LED ist aus) an den Dateneingang des Schieberegisters gelegt. Der Mikrocontroller aktiviert danach die Taktleitung, das Bit wird in das Schieberegister übernommen. Der Vorgang wird für das erste Zeichen fünfmal wiederholt, und all dies wiederum für die fünf anderen Zeichen. Nachdem das Muster für alle Zeichen so in das Schieberegister gelangt ist, wird der Treiber für die erste Zeile (R1) aktiviert. Alle LEDs der ersten Zeile, deren Spalteneingang Low ist, leuchten für eine kurze Zeit. Anschließend wird der Zeilentreiber wieder dunkelgetastet und die Spalteninformationen für die zweite Zeile behandelt. Dieser Vorgang setzt sich bis zur siebten Zeile fort und beginnt danach wieder mit der ersten Zeile. Nur wenn der Steuerrechner ein neues Zeichen überträgt, wird die Anzeigeroutine unterbrochen.

Schaltungsdetails

Nachdem über die grundsätzliche Funktion der Schaltung kein Zweifel mehr besteht, können wir uns einigen Schaltungsdetails widmen.
Da der Zeilendekoder 74LS138 low-aktive Ausgänge besitzt, die Zeilen jedoch die Anoden der Leuchtdioden bilden, sind Inverter/Puffer-Stufen unabdingbar. Diese Aufgabe übernehmen sieben PNP-Transistoren BC640. Die Spalten, also die Kathoden der Leuchtdioden, können durch das Schieberegister direkt über 330-Ω-Vorwiderstände angesteuert werden, da TTL-Schaltkreise je nach Ausführung 8...10 mA aufnehmen können.
Am bequemsten ist zweifelsohne der Einsatz von kompletten Punktmatrix-Displays. Zudem sind alle Leuchtdioden eines Displays auf geringe Helligkeitsunterschiede ausgemessen. Leider ist das Angebot an diesen Modulen nicht allzu groß. Geeignet sind beispielsweise die rot leuchtenden Kingbright-Displays TC12-11HWA (erhältlich bei Conrad). Neben der schlechten Auswahl kommt als weiterer Nachteil hinzu, dass diese Module für die Multiplexansteuerung nicht gerade prädestiniert sind. Ihre Leuchtkraft ist mit 560...1400 µcd recht gering, und selbst mit dem vierfachen Nennstrom (40 mA) wird nur ungefähr die doppelte Helligkeit erreicht. Wenn die Helligkeit nicht Ihren Vorstellungen entspricht, können Sie die Vorwiderstände auch problemlos auf 220 Ω verringern.
Aus der gleichen Serie sind (zum Beispiel bei Schuricht) auch andersfarbige und leuchtstärkere Module erhältlich, allerdings in kleinen Stückzahlen recht schwer:

TC12-11	*Leuchtkraft in µcd (I_F=10 mA) min...typ.*	*Farbe*
HWA	560...5600	Rot
YWA	2200...5600	Gelb
SGWA	3600...9000	Grün
EWA	3600...9000	Superrot
SRWA	5600...14000	Hyperrot

Natürlich kann man die Matrix-Anzeige auch aus Einzeldioden zusammenstellen. Hier sollte man, auch wenn es den Geldbeutel schmälert, Low-Current-LEDs verwenden, die bei Strömen von 2 mA schon die gleiche Leuchtkraft wie übliche Leuchtdioden bei 10 mA erreichen. Der Aufbau mit Einzel-Leuchtdioden hat jedoch noch einen weiteren Nachteil: Alle Leuchtdioden müssen mühselig und zeitintensiv einzeln ausgerichtet werden.
Die Stellung der vier DIP-Schalter, die mit den Porteingängen P3.2 bis P3.5 verbunden sind, ordnet dem Modul eine Adresse im Bereich 0 bis 15 zu. Die Adresse wird am DIP-Schalter binär eingestellt, wobei S1-8 das LSB ist. Der Schalter ist geschlossen, wenn der Schieber zum Controller-IC zeigt. Da die Module individuell adressiert werden, schließt man sie auch parallel (!) an die RS232-Leitung an. Das heißt, das alle Module dieselbe Information über die RS232-Leitung erhalten.
Das RS232-Interface ist ganz besonders einfach und ohne den üblichen MAX232 ausgeführt. Es besteht aus einem Transistor, zwei Widerständen und einer Diode. Der Vorwiderstand

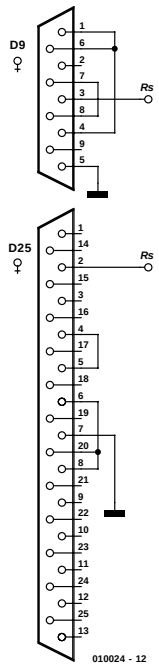


Bild 2. Die Verbindung zum COM-Port mit den notwendigen Modifikationen.

R7 begrenzt den Basisstrom für den Transistor, wenn die Schnittstelle des PCs ein High liefert. Der Transistor negiert diesen Pegel, so dass an Port 3.0 ein Low-Pegel anliegt. Liefert die Schnittstelle ein negatives Potential, so wird die Diode D1 leitend und begrenzt die negative Spannung an der Basis des Transistors auf etwa -0,7 V. Damit ist T1 sicher gesperrt, gegen zu hohe negative Spannung geschützt, und liefert ein High-Signal am P3.0.
Das Interface wird an die serielle PC-Schnittstelle angeschlossen. Da kein Handshaking stattfindet, müssen die entsprechenden Leitungen verbunden werden: bei einem 25-poligen Verbinder die Anschlüsse 4 mit 5 und 6 mit 8 und mit 20, bei einem 9-poligen Verbinder sind es 7 mit 8 und 1 mit 4 und mit 6 (**Bild 2**).
Die Übertragung findet mit 9600 Baud, keine Parität, acht Datenbits, ein Stoppbit statt. Als Taktgeber für den Mikrocontroller eignet sich daher ein Quarz mit einer Standardfrequenz von 22,1184 MHz. Durch Änderungen der Initialisierung des Mikrocontrollers lässt sich eine Anpassung an andere Quarzfrequenzen vornehmen.
Das Modul arbeitet mit einer

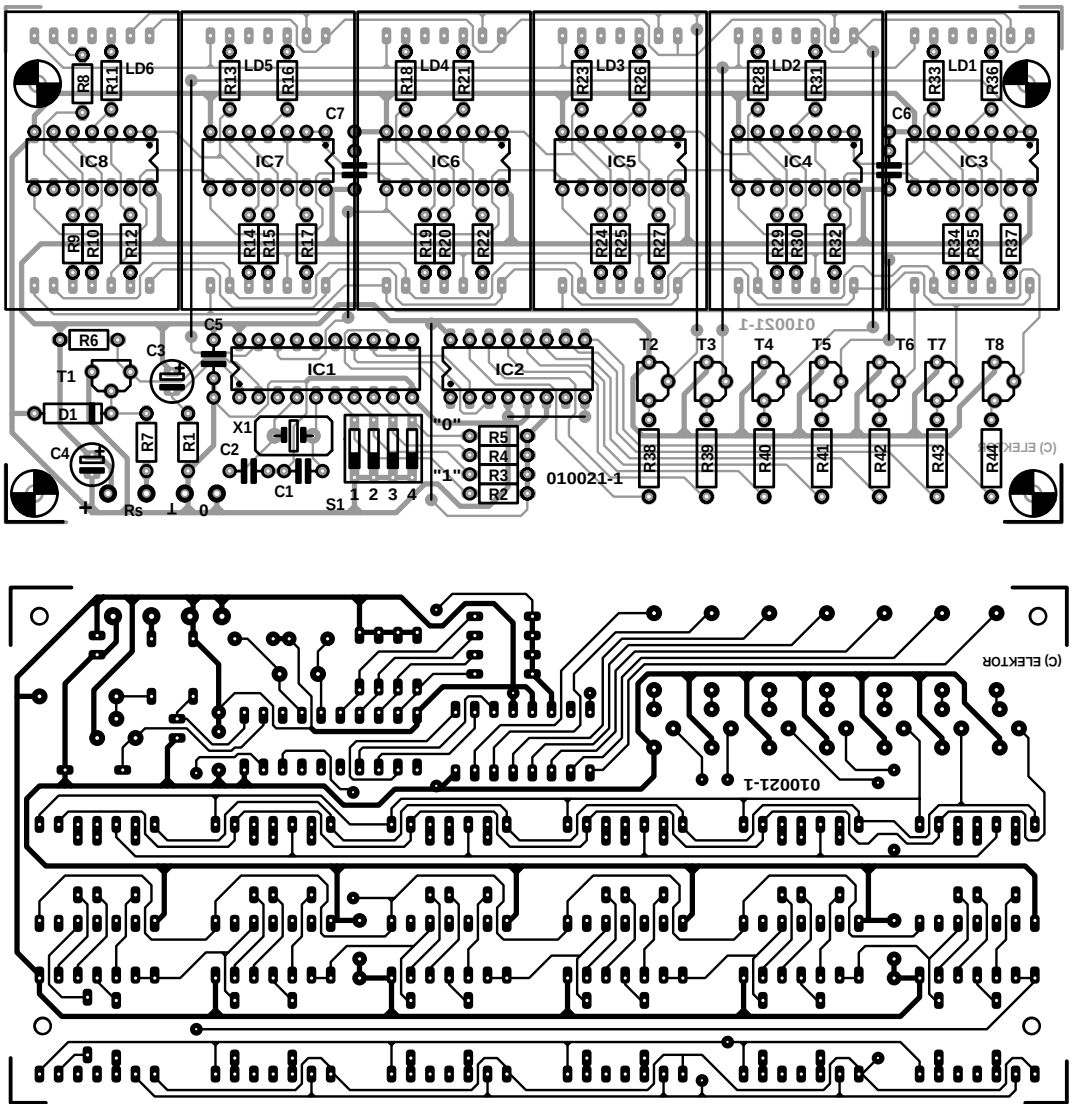


Bild 3. Das Layout und der Bestückungsplan der Platine für ein Modul.

Stückliste

Widerstände:
R1...R5 = 10 k
R6, R38...R44 = 2k2
R7 = 33 k
R8...R37 = 330 Ω

Kondensatoren:
C1,C2 = 18 p
C3,C4 = 10 µ/16 V stehend
C5...C7 = 100 n

Halbleiter:
D1 = 1N4148
LD1...LD6 = TC12-11HWA (Kingbright, bei Conrad)
T1 = BC547
T2...T8 = BC640
IC1 = 89C2051-24PC (programmiert)
IC2 = 74LS138
IC3...IC8 = 74LS164

Außerdem:
S1 = 4-facher DIP-Schalter
X1 = Quarz 22,1184 MHz
Prog. Controller EPS 010021-41
Diskette EPS 010021-11

Downloads siehe Seite 84

Betriebsspannung von 5 V. Die etwas altertümlichen LS-Bausteine fordern im Ruhezustand insgesamt fast 200 mA, dazu kommt noch der LED-Strom, der je nach Lage der Dinge bis 5·6·10 mA = 300 mA betragen kann. Das Netzteil sollte also 500 mA pro Modul liefern können.

Bild 3 zeigt Layout und Bestückungsplan für die Platine eines Moduls. Obwohl die Bauteile nicht allzu eng angeordnet sind, ergibt sich doch ein kompaktes Modul in einer alltagstauglichen Größe. Die Bestückung dürfte keine Probleme bereiten. Es sind insgesamt acht Drahtbrücken zu legen. Für die Schieberegister-ICs dürfen keine, für die Displays müssen Fassungen verwendet werden. Achten Sie bei der Montage der Bauteile unter den Displays auf eine ausreichend geringe Bauhöhe. Für den Mikrocontroller und den Dekoder IC2 dürfen (sollten) Sie Fassungen verwenden. Nach einer gründlichen Sichtkontrolle der Bestückungs- und Lötarbeiten ist das Modul einsatzbereit.

Controller-Software

Der wohl wichtigste Teil dieses Projektes ist die Software. Sie kann entweder in Form

```
REM Demoprogramm zur Ansteuerung der Punktmatrixanzeige
REM mit Leuchtdioden 5*7 Matrix
REM Der Text wird von rechts nach links durchgeschoben
REM bis die Anzeige wieder vollstaendig geloescht ist
REM Danach Pause von 1s und Beginn von vorn falls keine
REM Tastenbetaetigung
REM ————————————————————
REM Schnittstelleninit
REM ————————————————————
  OPEN "COM2:9600,N,8,1,CS,DS" FOR OUTPUT AS #1
  PRINT #1, CHR$(12);
REM ————————————————————
   CLS
   LOCATE 10, 10
   PRINT "Demoprogramm fuer Laufschrift"
   LOCATE 12, 10
   PRINT "Abbruch mit beliebiger Taste"
   Z$ = "Diese Schaltung wurde von Andreas Koehler fuer ELEKTOR entwickelt."
REM ————————————————————-
REM Ausgabeschleife
REM ————————————————————-
M2: V = 0
M3:    FOR ZA = V + 1 TO V + 12
       PRINT #1, MID$(Z$, ZA, 1);
       NEXT ZA
    V = V + 1
       FOR ZT = 1 TO 50000
       NEXT ZT
       PRINT #1, CHR$(12);
    IF V < LEN(Z$) THEN GOTO M3
    SLEEP (1)
    A$ = INKEY$
    IF A$ = "" GOTO M2
    END
```

Bild 4. BASIC-Programm für eine Laufschrift.

eines programmierten Controllers oder einer Diskette beim Elektor-Verlag oder kostenlos im Internet dem Download-Bereich der Elektor-Website entnommen werden. Dazu gehört nicht nur das HEX-File, mit dem der Controller direkt programmiert werden kann, sondern natürlich auch der gut kommentierte Assembler-Kode. Wer sich damit nicht auseinander setzen will, kann gleich zum nächsten Kapitel springen. Experimentierfreudige Assemblerkenner fahren aber mit der Beschreibung der Software fort.

Die Software beginnt mit der Initialisierung. Hier werden der Stackpointer, die Register für die serielle Schnittstelle und die Timerregister gesetzt. Weiterhin wird der Interrupt für die serielle Schnittstelle vorbereitet. In der Routine PBE, die nur beim Power-up durchlaufen wird, liest der Controller die am DIP-Schalter eingestellte Modul-Adresse. Nach einigen arithmetischen und logischen Operationen erhält er die Position des ersten Zeichens innerhalb des Verbands und speichert sie in einer Speicherstelle zwischen. Die übrigen fünf Positionen werden durch Inkrementieren berechnet.

Die Routine LOE löscht die Anzeige. Dazu wird jeder Ausgang des Schieberegisters mit einem High-Pegel geladen und der Zeilentreiber auf eine nicht genutzte Zeile gesetzt. Weiterhin wird der Zähler für die Zeichen, die über die serielle Schnittstelle übertragen werden, zurückgesetzt. Um die Anzeige ungültiger Zeichen auszuschließen, werden auch die Register der anzuzeigenden Zeichen gelöscht.

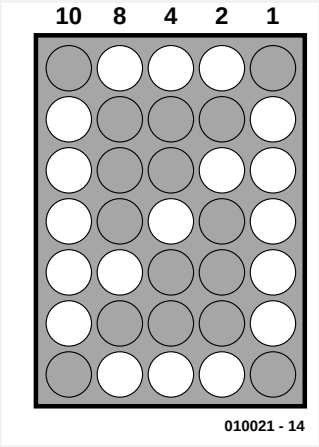
Danach beginnt der Mikrocontroller, die Anzeige anzusteuern. Da zunächst alle Register gelöscht sind, bleibt die Anzeige dunkel. Dies ändert sich erst, wenn vom PC ein ASCII-Zeichen übertragen wird. Dadurch wird die Interuptroutine SERIN abgearbeitet. Das erste Zeichen wird zwischengespeichert und zunächst gezählt. Danach findet eine Prüfung statt, ob es sich um das einzige zugelassene Steuerzeichen OCH (STRG L) handelt. Sollte dies der Fall sein, wird die Anzeige gelöscht und der Zeichenzähler zurückgesetzt. Eine weitere Kontrolle in SER0 stellt sicher, dass nur Zeichen größer 20H dargestellt werden. Ansonsten würde ein Teil des Programmcodes als Zeichengenerator verwendet werden, was unsinnige Anzeigen zur Folge hätte. Außerdem benötigt die Routine zur Umkodierung dann besonders viel Zeit und die Anzeige würde flimmern. Benutzerdefinierte Zeichen sollten deshalb stets den Anfang des Zeichengenerators belegen. SER1 überprüft, ob das übertragene Zeichen für die Anzeigeposition vorgesehen ist. Dazu werden alle durch PBE ermittelten Positionen mit dem Zeichenzähler verglichen. Bei Ungleichheit wird die Routine lediglich mit der Veränderung des Zeichenzählers verlassen. Nur ein für die Position gültiges Zeichen gelangt nach einer Korrektur des ASCII-Werts (alle Zeichen < 20H werden ausgeblendet) in ein Register für die Darstellung.

Dem Empfang und der Aufbereitung des Zeichens folgt die Darstellung. In einer Schleife werden nacheinander die Zeilen abgearbeitet. Aus der Zeilennummer und dem Zeichenkode im Anzeigeregister wird der Zeichengeneratorkode berechnet und im Register A zwischengespeichert. Die Routine ANZ untersucht den Zeichengeneratorkode bitweise. Je nach Pegel werden Bit für Bit in das Schieberegister übernommen. Das geschieht nacheinander für alle sechs Anzeigeregister R2 bis R7.

Ist eine Zeile im Schieberegister komplett, wird diese über den Zeilendekoder in drei Zeitschleifen aktiviert. Es folgt die achte, nicht vorhandene Zeile, in der die nächste Zeile aufgebaut wird. Das wiederholt sich, bis alle Zeilen angezeigt sind, danach beginnt alles wieder von vorne.

Andere Zeichen

Um eigene Zeichen zu definieren, muss in den Programmkode (punzelek.a51) des Controllers eingegriffen werden. Dies ist nicht den Spezialisten vorbehalten, man kommt auch mit wenig Erfahrung im Assembler-Programmieren aus. Alles, was man benötigt, ist (natürlich) ein einfaches Programmiergerät mit entsprechender Software, wie man sie beispielsweise unter
ftp://www.atmel.com/pub/atmel/asmb51.exe
ftp://www.atmel.com/pub/atmel/mlasm51.exe
kostenlos erhält.
Das Bild zeigt den Aufbau einer 5x7-Punktmatrix. In einer Zeile werden die Werte aller **nicht leuchtenden** LEDs zusammengezählt und zu 0E0H addiert. Die sich ergebenden sieben Werte werden wie folgt nacheinander im Zeichengenerator am Ende des Programmkodes eingetragen:




```
DEFB    0F1H,0EEH,0ECH,0EAH,0E6H,0EEH,0F1H    ;0
```

Das Ganze ist etwas mühsam, übt aber hexadezimales Rechnen!

PC-Software

Schauen wir nun auf die andere Seite der RS232-Verbindung, zum PC. Hier ist ein Terminalprogramm notwendig, das die ASCII-Zeichen im RS232-Format über die serielle Schnittstelle schiebt. Man könnte dazu beispielsweise Hyperterminal verwenden, anschaulicher ist es aber, ein eigenes, auf die jeweilige Aufgabe zugeschnittenes Prögrämmchen zu verfassen. **Bild 4** zeigt ein beispielhaftes kleines, kaum erläuterungsbedürftiges BASIC-Programm, das über COM2 den Text *Diese Schaltung wurde von Andreas Koehler fuer ELEKTOR entwickelt* als Laufschrift von rechts nach links durch zwei Module schiebt. Die ZA-Ausgabeschleife muss an die Zahl der angeschlossenen Module angepasst werden. Dazu trägt man in der Zeile

```
FOR ZA = V + 1 TO V + 12
```

Als letzte Zahl die Anzahl der Anzeigestellen ein, also 6 bei einem Modul, 12 bei zweien, 18 bei dreien und so weiter.
Die Verzögerung der folgenden ZT-Zeitschleife bestimmt die Geschwindigkeit, mit der der Text durchläuft. Die Anzahl der Schleifendurchläufe von 50000 ist für langsamere PCs ganz akzeptabel, bei “aktuellen” PC-Taktraten ist es notwendig, den Wert zu verzehn- oder gar verhundertfachen.

(010021)rg