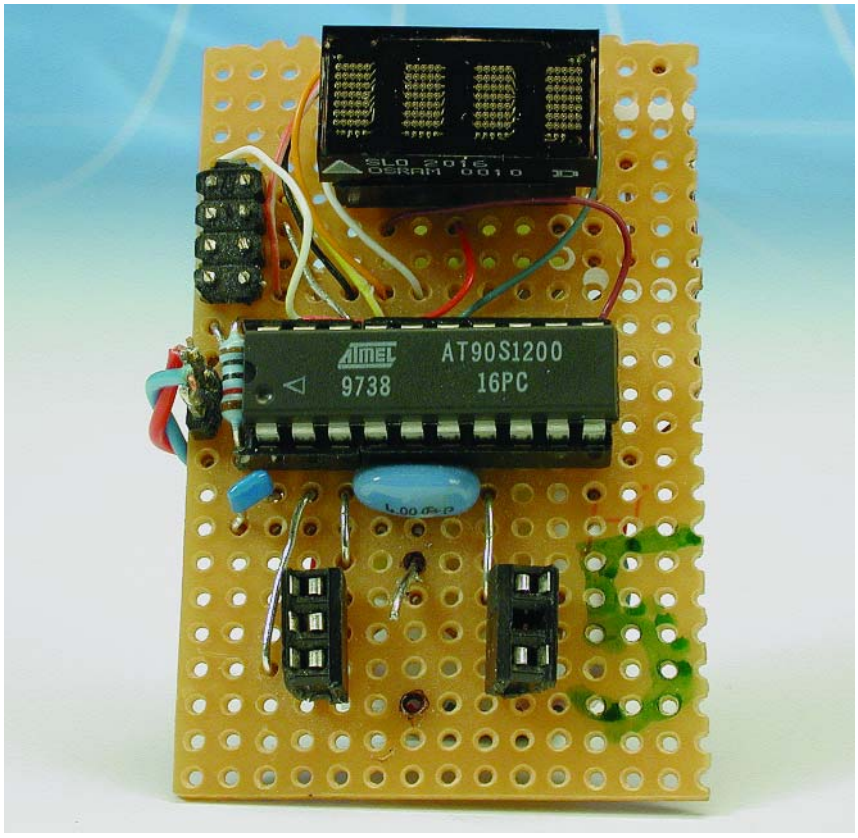
Mini-Laufschrift

Für Weihnachts- und andere Grüße

Von R. van Arem

Selbst gebastelte Geschenke sind immer die schönsten. Da macht auch diese pfiffige Mini-Laufschrift keine Ausnahme, wenn ein *Frohe Weihnachten* durch das Display schiebt.

Die Palette der traditionellen Weihnachtsschaltungen in Elektor reicht von ganz einfach bis mit wahrer „Intelligenz“ ausgestattet. Die Mini-Laufschrift, die wir Ihnen dieses Jahr präsentieren, gehört klar zur zweiten Sorte, obwohl (oder gerade weil) nur wenige Bauteile zum Einsatz kommen.
Wie die Schaltung in **Bild 1** zeigt, sind nur wenig mehr Bauteile als der Mikrocontroller AT90S1200, ein LED-Display mit vier Zeichen und ein 5-V-Spannungsregler erforderlich. Der Controller benötigt lediglich einen POR-Schaltkreis (PowerOnReset) in Form von R1 und C3 sowie einen 4-MHz-Quarz.
Mit einem 5-V-Spannungsregler 7805 sollte die Versorgungsspannung wenigstens 8 V betragen. Dafür kann ein gewöhnliches Steckernetzteil (9 V DC) oder auch eine 9-V-Batterie verwendet werden. Allerdings liegt die Stromaufnahme bei etwa 50 mA. Für ein Steckernetzteil kein Problem, eine Batterie ist aber nur für intermittierenden Betrieb mit kurzen Einschaltzeiten zu empfehlen, da sie im Dauerbetrieb nur ein paar Stunden durchhält.



Das Display ist ein Miniaturtyp von Osram mit der Bezeichnung SLO2016. Obwohl es mit nur 10·20 mm^2 sehr klein ist, sind die Zeichen deutlich und auch aus ungünstigen Betrachtungswinkeln gut zu erkennen. Es handelt sich um ein so genanntes „intelligentes“ Display, das vier alphanumerische Zeichen in 5x7-Matritzen darstellen kann. Der SLO2016 verfügt über einen internen Speicher, in dem 128x7 Bit ASCII-Zeichen untergebracht sind. Der Zeichensatz in **Bild 2** zeigt, dass auch zahlreiche Sonderzeichen enthalten sind, so dass Weihnachtsgrüße in Deutsch, Englisch, Italienisch und den skandinavischen Sprachen möglich sind. Die darzustellenden Zeichen werden vom Mikrocontroller in den RAM-Speicher übergeben. Der 7 bit breite ASCII-Kode (der gleiche wie beim PC) wird in die Punktmatrix umgesetzt.
Um die Ansteuerung der Reihen und Spalten und das ganze Multiplexing braucht man sich keine Gedanken machen, denn auch diese Funktionen sind im Display bereits integriert.

Controller programmieren

Der Mikrocontroller ist nicht fertig programmiert erhältlich, so dass Sie sich selbst ans Programmieren machen müssen. Das ist aber gar nicht so kompliziert und kostet auch nicht viel.
Zuallererst muss man natürlich einen geeigneten Assembler und

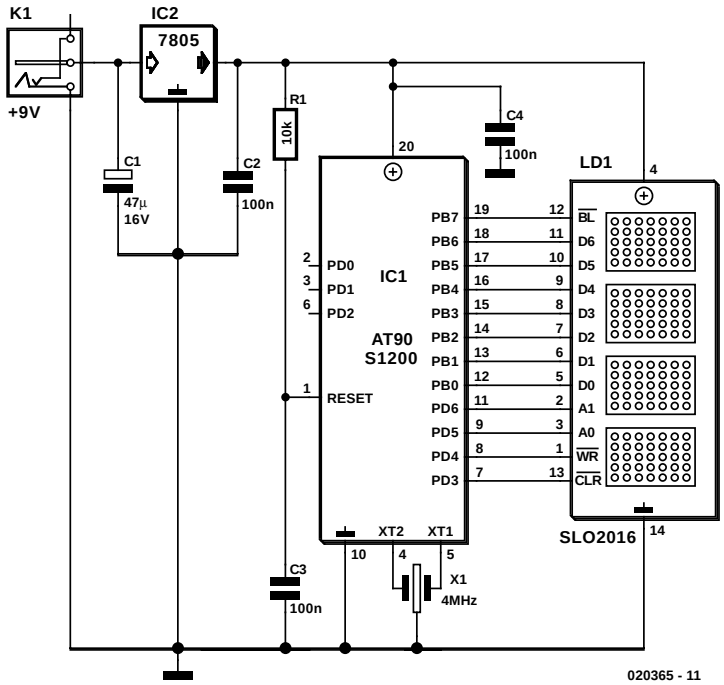


Bild 1. Weihnachtsgruß mit Controller und Display.

Uploader für den AT90S1200 besorgen. Kein Problem, auf der Website von Atmel (www.atmel.com) gibt es das kostenlose **AVR-Studio**, das unter anderem diese beiden Komponenten enthält, mit denen sich die Datei assemblieren und zum Controller hochladen lässt. Außerdem ist natürlich ein **Programmiergerät** erforderlich, das von ganz einfach bis luxuriös ausfallen kann. In den letzten Jahren haben wir Ihnen zahlreiche Bauvorschläge für geeignete Programmer vorgestellt. Und drittens ist natürlich der **Quellkode** wichtig, den Sie ohne weiteres unter der Nummer 020365 (zusammen mit dem Platinenlayout) von der Elektor-Site herunterladen oder auf Diskette EPS 020365-11 beim Verlag beziehen können, und zwar in zweifacher Ausfertigung (Weihnachtsgruß auf Deutsch und Englisch). Der Quellcode wird mit dem Assembler in eine HEX-Datei verwandelt und mit dem Uploader vom PC zum angeschlossenen Mikrocontroller übertragen.

Andere Texte

Natürlich kann man anstelle eines Weihnachtsgrußes auch andere Texte im Speicher des Mikrocontrollers unterbringen. Dazu muss man nicht einmal Assembler-Spezialist sein, es klappt auch fast ohne Programmiererfahrung. Der Quellkode mit der Bezeichnung Slo2016p.asm wird in den Assembler geladen. In **Bild 3** ist ausschnittsweise das Ende des Quellkodes zu sehen. Der Text ist in einer Tabelle untergebracht. Den ASCII-Kode der

ASCII CODE				D0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
				D1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
				D2	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
				D3	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
D6	D5	D4	HEX		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	0	0																	
0	0	1	1																	
0	1	0	2																	
0	1	1	3																	
1	0	0	4																	
1	0	1	5																	
1	1	0	6																	
1	1	1	7																	

020365 - 12

Notes:
1. High=1 level
2. Low=0 level
3. Upon power up, the device will initialize in a random state.

Bild 2. Zeichensatz des SLO2016.

Zeichen kann man dem Zeichensatz in Bild 2 entnehmen und hinter .db 0x eintragen (zuerst den Wert des oberen Nibbles D4...D7 und dann des unteren Nibbles D3...D0 jeweils im Hexadezimalformat). Der Buchstabe F zum Beispiel hat den ASCII-Kode 46, ein Leerzeichen 20 und ein „Vollzeichen“ 7F. Die Nummerierung der Zeichen hinter data ist ein Kommentar und nur zur besseren Orientierung erforderlich.

In der drittletzten Zeile vor der Tabelle mit dem Anzeigetext ist die Anzahl (hier 53) der dazustellenden Zeichen (inklusive Leerzeichen) eingetragen. Am Ende sollten mindestens vier Leerzeichen aufgenommen werden, damit das Textende vom Display verschwindet, bevor das erste Zeichen wieder erscheint. Der AT90S1200 bietet 64 Byte EEPROM, der ebenfalls einsetzbare AT90S2313 die doppelte Zahl von Zeichen. Wer also extra lange Botschaften durch das Display schieben will, muss den größeren Controller verwenden und außerdem zu Beginn des Quelltextes diesen Controller aktivieren (das Kommentarzeichen ; wegnehmen und vor die 90S1200-Zeile setzen).

(020365)rg

```
cpi     adrs, 53        ;if adrs = 53 (*end position)
breq    start           ; -> restart scrolling message
rjmp    scroll          ;scroll message (loop)


;* Message data = FROHE WEIHNACHTEN UND EIN GlÜCKLICHES NEUES
JAHR in EEPROM table
;* (Character(s) data see the SLO2016 datasheet (ROM ASCII set)

.ESEG

;* Data

table:
.db 0x46        ;data00 = F
.db 0x52        ;data01 = R
.db 0x4F        ;data02 = O
.db 0x48        ;data03 = H
.db 0x45        ;data04 = E
.db 0x20        ;data05 = ‘space’
.db 0x57        ;data06 = W
.db 0x45        ;data07 = E
.db 0x49        ;data08 = I
.db 0x48        ;data09 = H
.
.
.db 0x20        ;data43 = ‘space’
.db 0x4A        ;data44 = J
.db 0x41        ;data45 = A
.db 0x48        ;data46 = H
.db 0x52        ;data47 = R
.db 0x21        ;data48 = !
.db 0x20        ;data49 = ‘space’
.db 0x20        ;data50 = ‘space’
.db 0x20        ;data51 = ‘space’
.db 0x20        ;data52 = ‘space’


;* end position in this example is 35th EEPROM address (0 to 35
-> 36!)
```

Bild 3. Ausschnitt aus dem Quellkode.