80 Zeichen per Software

Gleich zwei Themen sind diesmal in der Assemblerecke enthalten: Erstens die Ausgabe von 80 Zeichen pro Zeile in einem GRAPHICS-8 Screen und zweitens kommt die MiniRAMDisk aus Heft 7/85 nochmal zum Zuge. Diesmal allerdings, auf vielfachen Leserwunsch, in einer für den Kassettenrekorder angepassten Form. Nebenbei bemerkt, feiert die Assemblerecke mit dieser Ausgabe ihren ersten Geburtstag. Und es gibt gleich zwei Geburtstagsgeschenke, ein gutes aber auch ein weniger erfreuliches. Fangen wir mit dem schlechteren an: Wie Sie sicherlich bereits bemerkt haben, erscheint die CK jetzt zweimonatlich, so dass es nur noch 6 Assemblerecken im. Jahr geben wird. Dafür, und das ist die gute Nachricht, steht in Zukunft mehr Platz zur Verfügung. Daher gibt’s diesmal auch gleich zwei Themen.

80 Zeichen

Bestimmt haben Sie schon eine Anzeige für das eine oder andere Textverarbeitungs-Programm gesehen, in der mit 80 Zeichen pro Zeile geworben wird. Genug Grund, um in der Assemblerecke zu zeigen, wie man so etwas machen kann. Das Programm in Listing 1 kann neben den 80 Zeichen sogar noch verschiedene Schriftgrößen darstellen (s. Bild 1).

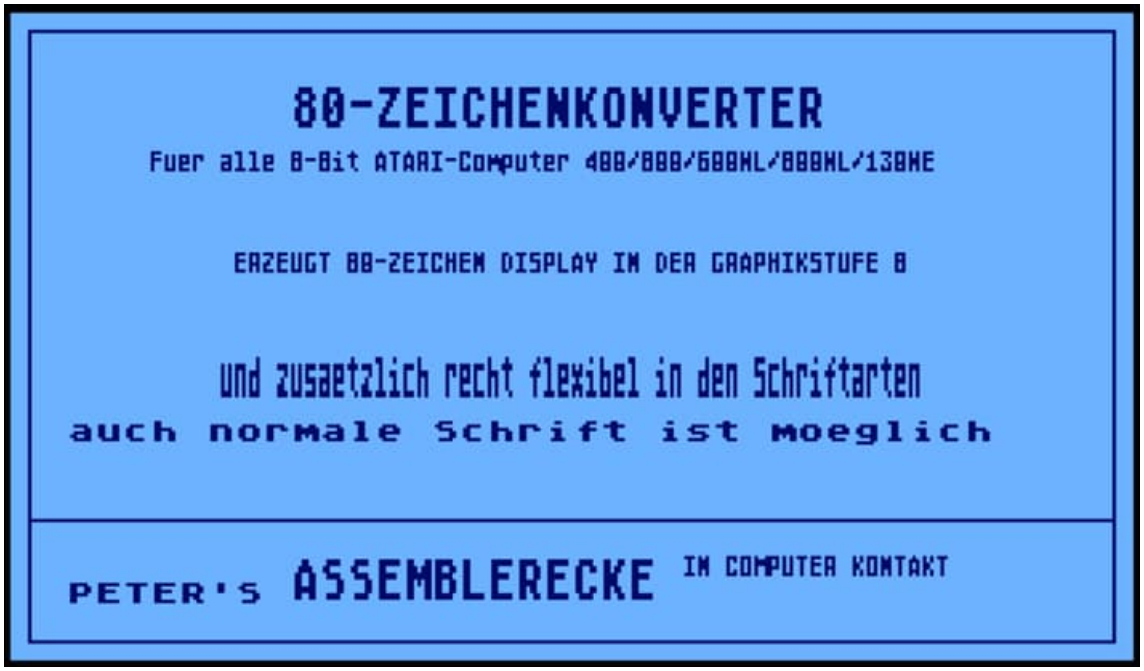

Bild 1

Aus eigener Erfahrung wissen Sie, dass Ihr Atari im normalen GRAPHICS-0 Modus nur 40 Zeichen in einer Zeile darstellen kann, also wo liegt der Trick? Nun ja, man muss statt dem gewohnten GR.0 Modus die am höchsten auflösende Grafikstufe 8 nehmen und die Zeichenmatrix (die Grafikpunkte, die das Zeichen darstellen) per Software in den Videospeicher eintragen. Wenn Sie ein eifriger Leser der Assemblerecke sind, dann kennen Sie das Prinzip noch von Heft 4/ 85, in dem nach gleichem Strickmuster Text und Grafik gemischt wurde.

Damals konnten wir aber auch nur 40 Zeichen in einer Zeile darstellen. Den Grund dafür kann man sich anhand einer einfachen Rechnung klarmachen. GR.8 kann in horizontaler Richtung 320 Grafikpunkte (Pixels) darstellen. Ein Zeichen aus dem im ROM abgelegten Zeichensatz ist 8 Pixels breit, so dass 320/8=40 Zeichen darstellbar sind. Für 80 Zeichen müssen wir zu einer List greifen: Wir machen jedes Zeichen statt 8 nur noch 4 Pixels breit und können dadurch doppelt so viele Zeichen darstellen.

4 x8 Matrix

Ein paar Haken hat die Sache natürlich: Statt 1 KByte für den 40-Zeichen Schirm, braucht die 80-Zeichen-Darstellung immerhin 8 KByte (GR.8). Schlimmer ist aber, dass die kleinen Zeichen bei weitem nicht so gut zu lesen sind. Das hat zwei Gründe: In der 4x8 Matrix lassen sich die Buchstaben längst nicht so schön wie in einer 8x8 Matrix darstellen. Das liegt daran, dass zum nächsten Buchstaben ja noch Platz sein sollte, so dass im Grunde nur 3 Pixels in der Breite zur Verfügung stehen.

Der zweite Grund ist eine Eigenheit von GR.8, die dem einen oder anderen Leser sicherlich schon in einem anderen Zusammenhang aufgefallen ist. Wenn zwei Grafikpunkte waagerecht nebeneinander gesetzt werden, so sind diese wesentlich heller als ein einzelner Punkt. Im ROM-Zeichensatz ist dieses Problem recht einfach umgangen, da immer (prüfen Sie es doch mal nach) zwei Punkte nebeneinander gesetzt sind und so eine gleichmäßige Helligkeit erreicht wird. Bei den kleineren 4x8 Zeichen ist das verständlicherweise nicht denkbar, so dass die Zeichen unterschiedlich hell sind.

Die Darstellung lässt sich etwas verbessern, wenn man den Hintergrund hell und die Zeichenfarbe dunkel stellt. Aber mit Farben und Helligkeitsstufen können Sie im Listing 1 selbst ausgiebig experimentieren. Gegen die schlechter lesbare 4x8 Darstellung lässt sich aber nichts machen, damit muss man sich einfach abfinden.

Zur Anwendung

Wie immer in der Assemblerecke finden Sie das Maschinenprogramm sowohl als Basic-Loader wie auch als Assemblerlisting abgedruckt. Der Basic-Loader enthält zusätzlich ein Beispiel zur Anwendung. Sie können die Zeilen ab 30.000 jedoch in eigene Programme einbauen. Das Assemblerlisting müssen Sie nicht unbedingt eintippen, es ist nur als Hintergrundinformation für erfahrene Leser gedacht. Das Maschinenprogramm befindet sich in Page 6 und kann vom Basic aus mit einem USR-Befehl aufgerufen werden.

Dazu bedarf es aber einiger Vorarbeit. Um das Maschinenprogramm kurz zu halten, wurde zu einem Trick gegriffen: Bevor die USR-Routine mit der Arbeit beginnt, muss die Position des Textes am Bildschirm durch einen PLOT-Befehl markiert werden. Dabei werden nicht alle 320 möglichen horizontalen Positionen ausgewertet, sondern nur ganze Zeichenpositionen, d.h., es werden im 80-Zeichenmodus 80 und im 40Zeichenmodus 40 Positionen berücksichtigt.

Vor dem PLOT-Befehl schalten Sie zweckmäßigerweise auf die Hintergrundfarbe um (COLOR 0), damit kein Punkt zurückbleibt. Der USR-Anweisung müssen Sie die Adresse eines auszugebenden Strings und dessen Länge mitgeben. Außerdem wird ein Attribut benötigt, mit dem man das Schriftbild beeinflussen kann. Folgende Werte sind zulässig:

0:	40-Zeichen, normale Schrift
64:	40-Zeichen, doppelt hoch
128:	80-Zeichen, normal hoch
192:	80-Zeichen, doppelt hoch

Sie sehen, damit läßt sich allerhand anfangen. Schreibt man eine Zeile mit mehreren USR-Befehlen (s. Demo), dann lässt sich die Schriftgröße auch innerhalb einer Zeile verändern. Hier ein Beispiel zur Ausgabe eines Textes.

```
COLOR 0: PLOT 120,100
X=USR (1536,ADR("HALLO"),5,128)
```

Diese Befehle schreiben das Wort ”HALLO” im 80-Zeichenmodus auf den Schirm. Voraussetzung ist natürlich, dass das Maschinenprogramm im Speicher liegt und GR.8 aufgerufen ist.

Wie läuft’s?

Wenden wir jetzt unsere Aufmerksamkeit dem Assemblerlisting zu. Zu Anfang werden die Parameter des USR-Befehles per PLA-Befehle vom Stack genommen und abgespeichert. Aus OLD ADR ($5E) wird die Adresse des Bytes entnommen, welches durch den letzten PLOT-Befehl angesprochen wurde. Dann wird mit Hilfe der Speicherzelle OLDCOL ($5B, $5C) die momentane Schreibspalte ermittelt und ein Merker (FLAG80) für ungerade Spalten gesetzt. Danach wird Zeichen um Zeichen aus dem String geholt, vom ATASCII in den internen Bildschirmcode gewandelt und die Position des Bitmusters im Zeichensatz berechnet. Danach wird das Bitmuster Zeile für Zeile in den Grafikspeicher übertragen, wobei die einzelnen Punkte den eventuell bereits vorhandenen Bildschirminhalt überlagern.

Die 80-Zeichen werden durch das Unterprogramm WANDEL80 ermöglicht. Hier wird das 8-Bit breite Bitmuster des normalen ROM-Zeichensatzes durch den Einsatz zweier kurzer Tabellen auf 4 Bit komprimiert und je nach Inhalt von FLAG80 um 4 Bit verschoben. Schließlich müssen ja im 80-Zeichenmodus pro Byte die Bitmuster zweier Zeichen eingetragen werden.

Die Wandelroutine wird nur aufgerufen, wenn Bit 7 des Attributes gesetzt ist, andernfalls wird im 80-Zeichenmodus ausgegeben. Ist zusätzlich Bit 6 gleich 1, dann wird jede Bitmusterzeile doppelt ausgegeben und man erhält doppelt hohe Zeichen, sowohl im 40- als auch im 80-Zeichenmodus.

Peter Finzel

```
10 REM ********************************
20 REM * AUSGABE VON 80-ZEICHEN
30 REM * BASIC-LOADER MIT DEMO
40 REM * P. FINZEL 1986
50 REM ********************************
100 DIM A$(127):GOSUB 30000
110 GRAPHICS 8+16:SETCOLOR 2,8,10:SETCOLOR 1,0,0
120 COLOR 1:PLOT 5,5:DRAWTO 314,5:DRAWTO 314,186:DRAWTO 5,186:DRAWTO 5,5
130 PLOT 5,150:DRAWTO 314,150
140 RESTORE 1000:FOR I=0 TO 7
150 READ X,Y,ATT,A$
160 COLOR 0:PLOT X,Y:REM WICHTIG!
170 A=USR(1536,ADR(A$),LEN(A$),ATT)
180 NEXT I
190 GOTO 190
200 REM *
210 REM * UND HIER EIN PAAR TEXTE
220 REM * ALS DEMO. . .
230 REM *
1000 DATA 80,20,64,80-ZEICHENKONVERTER
1010 DATA 40,40,128,Fuer alle 8-Bit ATARI-Computer
400/800/600XL/800XL/130XE
1015 DATA 64,70,128,ERZEUGT 80-ZEICHEN DISPLAY IN DER GRAPHIKSTUFE 8
1020 DATA 60,100,192,und zusaetzlich recht flexibel in den Schriftarten
1030 DATA 16,120,0,auch normale Schrift ist moeglich
1040 DATA 16,168,0,PETER'S
1050 DATA 80,160,64,ASSEMBLERECKE
1060 DATA 192,160,128,IN COMPUTER KONTAKT
30000 REM * 80-ZEICHEN ML-PROGRAMM
30010 S=0:RESTORE 30100
30020 FOR A=1536 TO 1746:READ D:POKE A,D:S=S+D:NEXT A
30030 IF S<>25816 THEN ? "DATEN-FEHLER!":STOP
30090 RETURN
30100 DATA 104,104,133,209,104,133,208,104,104,133,210,104,104,133
30110 DATA 214,169,0,133,211,133,205,133,213,165,92,133,216,165,91
30120 DATA 102,216,106,102,216,106,133,212,165,212,201,80,176,103
30130 DATA 106,102,213,165,94,133,203,165,95,133,204,160,0,132,207
30140 DATA 164,211,177,208,41,127,201,96,176,11,201,32,176,4,9,64
30150 DATA 208,3,56,233,32,10,10,38,207,10,38,207,133,206,24,165,207
30160 DATA 109,244,2,133,207,162,8,160,0,177,206,36,214,16,3,32,147
30170 DATA 6,32,191,6,36,214,80,3,32,191,6,230,206,202,208,230,230
30180 DATA 212,36,214,16,6,36,213,48,4,16,4,230,212,230,205,230,211
30190 DATA 198,210,208,147,96,133,215,169,0,133,216,160,3,185,183
30200 DATA 6,36,215,240,7,185,187,6,5,216,133,216,136,16,239,165,216
30210 DATA 36,213,48,4,10,10,10,10,96,1,4,16,32,1,2,4,8,72,164,205
30220 DATA 17,203,145,203,24,165,203,105,40,133,203,144,2,230,204,104,96
```

```
**********************************
* 80-ZEICHEN MIT GRAPHICS 8    *
*                              *
* PETER FINZEL           1986 *
*                              *
* ASSEMBLER: ATMAS-II          *
**********************************


* BETRIEBSSYSTEMADRESSEN:

OLDCOL   EQU $5B            X-POS. DES CURSORS
OLDADR   EQU $5E            CURSORADRESSE
CHBAS    EQU $02F4          PAGENR. DES ZEICHENSATZES

* BELEGUNG DER ZEROPAGE

         ORG $CB

BILD     EQU *              ZEIGER IN HI- RES BILDSCHIRM
XOFFSET  EQU *+2            ABSTAND IN BYTES VON OLDADR
ZSATZ    EQU *+3            ZEIGER IN ZEICHENSATZ
STRING   EQU *+5            ZEIGER IN TEXTSTRING
LAENGE   EQU *+7            LAENGE DES STRINGS
SINDEX   EQU *+8            INDEX IN STRING
SPALTE   EQU *+9            SCHREIBSPALTE
FLAG80   EQU *+10           80-ZEICHEN AUSGABE
ATTBUT   EQU *+11           DOPPELT HOHE ZEICHEN
ZALT     EQU *+12           ZWISCHENSPEICHER
HILF     EQU *+13           HILFSREGISTER

****************************************
* 80-ZEICHEN IN GRAPHICS 8
* AUFRUF: X=USR(1536,ADRESSE,LAENGE,ATT)
* ATTRIBUT:   0: NORMAL, 40 ZEICHEN
*            64: DOPPELT HOCH, 40 ZEICHEN
*           128: NORMAL , 80 ZEICHEN
*           192: DOPPELT HOCH, 80 ZEICHEN
****************************************

         ORG $0600          PROGRAMM IN ZEROPAGE

         PLA                # DER ARGS
         PLA                STRINGADRESSE
         STA STRING+1
         PLA                LSB
         STA STRING
         PLA                MSB DER LAENGE
         PLA                LAENGE
         STA LAENGE         (MAX. 255 ZEICHEN)
         PLA
         PLA                ATTRIBUT
         STA ATTBUT         ABLEGEN
*
* VORBESETZUNGEN, SCHREIBSPALTE BERECHNEN
*
         LDA #0             INDEX IN STRING
         STA SINDEX         ZURUECKSETZEN
         STA XOFFSET
         STA FLAG80
         LDA OLDCOL+1       POSITION DES
         STA HILF           GRAPHIK-CURSORS
         LDA OLDCOL         DURCH VIERTEILEN
         ROR HILF           ERGIBT DIE SCHEIB-
         ROR                ;SPALTE
         ROR HILF
```

```
          ROR
          STA SPALTE
*
* SCHLEIFE ZUR AUSGABE BEGINNT
*
WEITER    LDA SPALTE          ZEILENENDE?
          CMP #80             DANN NICHTS MEHR
          BCS W80ENDE         AUSGEBEN! - >
          ROR                 ;FLAG FUER UNGERADE
          ROR FLAG80          SPALTENZAHL (TRICK!)
          LDA OLDADR          CURSOR ADRESSE DES
          STA BILD            LETZTEN PLOT-PUNKTES
          LDA OLDADR+1        UEBERNEHMEN
          STA BILD+1
*
* ZEICHENCODE AUS STRING, ATASCII IN
* INTERNEN CODE UMRECHNEN
*
          LDY #0
          STY ZSATZ+1         VORBESETZUNG
          LDY SINDEX
          LDA (STRING),Y      ASCII-ZEICHEN IM AKKU
          AND #$7F            INVERSBIT MASKIEREN
          CMP #96             JETZT UMWANDLUNG I  BILDSCHIRMCODE
          BCS WENDE           CODE STIMMT - >
          CMP #32             IST GRAPHIKZEICHEN?
          BCS ALPHA           NEIN, IST ALPHAZEICHEN - >
          ORA #64             PLUS 64
          BNE WENDE           UMWANDLUNG FERTIG - >
ALPHA     SEC
          SBC #32             KORREKTUR BUCHSTABEN UND ZAHLEN
*
* ADRESSE DES BITMUSTERS IM ZEICHENSATZ
*
WENDE     ASL                 ;BILDSCHIRMCODE MAL
          ASL                 ;ACHT IST ZEIGER IN
          ROL ZSATZ+1         DEN ZEICHENSATZ
          ASL
          ROL ZSATZ+1
          STA ZSATZ           LSB IST FERTIG
          CLC
          LDA ZSATZ+1
          ADC CHBAS           BASISADRESSE DES
          STA ZSATZ+1         ZEICHENSATZES DAZU
*
* SCHLEIFE ZUR AUSGABE EINES ZEICHENS
*
          LDX #8              ACHT ZEILEN/ZEICHEN
ZEILE     LDY #0
          LDA (ZSATZ),Y       EINE ZEILE DES ZEICHENS
          BIT ATTBUT          40ER MODUS?
          BPL M2              JA - >
          JSR WNDEL80         IN 80ER UMWANDELN
M2        JSR DATOUT          AUSGEBEN
          BIT ATTBUT          DOPPELT HOCH?
          BVC M3              NEIN - >
          JSR DATOUT
M3        INC ZSATZ           NAECHSTES ZEICHENMUSTER
          DEX
          BNE ZEILE           NAECHST. ZEILE
          INC SPALTE          NAECHSTE SPALTE
          BIT ATTBUT          40-ZEICHEN-MODUS?
          BPL M4              JA - >
          BIT FLAG80          UNGERADE ADRESSE?
          BMI M5              JA, DANN INDEX WEITER
          BPL M6              NEIN, INDEX BLEIBT!
M4        INC SPALTE          FUER 40ER MODUS 2 SPALTEN
```

```
M5        INC XOFFSET       VIDEORAM WEITER
M6        INC SINDEX        ZEIGER IN STRING
          DEC LAENGE        NOCH ZEICHEN DA?
          BNE WEITER        JAWOHL - >
W80ENDE   RTS
*
*************************************
*  40ER ZEICHENSATZ IN 80ER UMRECHNEN
*************************************
*
WNDEL80   STA ZALT          40ER ZEICHENMUSTER
          LDA #0            NEUES MUSTER
          STA HILF          VORBESETZEN
          LDY #3            VIER BIT BEARBEITEN
SPALTE80  LDA BITTAB,Y      SPALTEN-MASKE
          BIT ZALT          BIT GESETZT?
          BEQ W80NIX        NEIN - >
          LDA U80TAB,Y      UMRECHENTABELLE
          ORA HILF
          STA HILF
W80NIX    DEY               SCHON ALLE SPALTEN?
          BPL SPALTE80      NEIN->
          LDA HILF          EBEN BERECHNETES MUSTER
          BIT FLAG80
          BMI UNGER         UNGERADE X-POSITION->
          ASL               ;SONST IN UNGERADE
          ASL               ;POSITION SHIFTEN
          ASL
          ASL
UNGER     RTS
*
* TABELLEN ZUR UMWANDLUNG IN 80ER Z-SATZ
*
BITTAB    DFB $01,$04,$10,$20
U80TAB    DFB $01,$02,$04,$08
*
*************************************
* BILD-DATEN IN SCREEN EINTRAGEN
* ZEIGER AUF NAECHSTE GRAPHIKZEILE
*************************************
*
DATOUT    PHA               AKKU MERKEN. . .
          LDY XOFFSET
          ORA (BILD),Y      MIT VORH. BILD VERKNUEPFEN
          STA (BILD),Y      UND EINTRAGEN
          CLC
          LDA BILD          NAECHSTE ZEILE IST
          ADC #40           40 BYTES WEITER
          STA BILD
          BCC DAT1          MSB NICHT VERGESSEN
          INC BILD+1
DAT1      PLA               AKKU RESTAURIEREN
          RTS
```