05 Scrolling - Teil 1 (CK 06/85)

Das Scrollen des Bildschirmes ist eine, der ältesten aber gleichzeitig auch interessantesten Fähigkeiten, über die Computer verfügen. Grundsätzlich kann man Scrolling dann brauchen, wenn man mehr Information am Bildschirm darstellen will, als in eine Füllung des Bildschirmspeichers passt, etwa wenn ein langes Basic-Listing am Bildschirm ausgegeben werden soll. Dass die Ausgabe eines Basic-Listing keine besonders anspruchsvolle Art des Scrollens ist, braucht nicht extra erwähnt zu werden, denn es handelt sich dabei um sogenanntes Software-Scrolling.

Die Möglichkeit des Scrollings reichen noch viel weiter. Programme zur Textverarbeitung können damit die Beschränkung auf 40 Zeichen pro Zeile elegant umgehen, die Darstellung von umfangreichen elektronischen Schaltungen wäre, denkbar und vieles, vieles mehr. Und natürlich wäre eine ganze Anzahl der besseren Spiele ohne Scrolling überhaupt nicht vorstellbar (Scramble, Caverns of Mars, Zaxxon..).

Jeder wird nun einsehen, dss das ruckartige Scrolling eines Basic-Listings mit dem eines Zaxxon recht wenig gemein hat. Daher ist es sinnvoll, einige verschiedene Arten des Scrollens zu unterscheiden: Software- und Hardware-Scrolling, vertikalem und horizontalem Scrolling, Grob- und Fein-Scrolling.

Artenvielfalt

Beginnen wir von vorne: Ein Beispiel für Software-Scrolling haben wir bereits besprochen. Charakteristisch dafür ist. dass jedes Byte des Bildschirmspeichers per Software an seine neue Stelle verschoben wird. Zu diesem Zweck ist es oft nötig, mehrere hundert oder gar mehrere tausend Byte zu bewegen . Wesentlich eleganter ist da schon das Hardware-Scrolling, eine Fähigkeit, die der Atari wie kein anderer auf dem Homecomputer Sektor beherrscht. Der wesentliche Grundgedanke dabei ist es, nicht die Bildinformation (in Form von Bytes) durch den Bildschirmspeicher zu schaufeln, sondern den Bildschirmspeicher über die Bildinformation zu verschieben . Damit kann man (im Extremfall) durch die Veränderung von nur zwei Bytes denselben Effekt herbeiführen, für den man beim Software-Scrolling nahezu 8000 Bytes verschieben müsste.

Vertikales und horizontales Scrolling unterscheiden sich einfach in der Richtung der Bewegung. Dabei ist horizontales Scrolling beim Atari etwas aufwendiger als rein vertikales. Selbstverständlich können auch beide Formen kombiniert auftreten (2D-Scrolling). Grob- und Fein-Scrolling bezieht sich auf die kleinste Bewegungseinheit des Scrollens. Grob heißt dabei, dass als Grundschritt ein Zeichen dient , als Fein-Scrolling wird eine pixelweise Bewegung des Bildes bezeichnet.

Vertikales Grob-Scrolling

Nach soviel trockener Theorie kann ein wenig Praxis nicht schaden. Das Atari-Basic-Programm in Listing 1 demonstriert, wie hardwaremäßig, vertikales Grob-Scrolling funktioniert. Es benutzt den Charakter Modus GRAPHICS 1 und scrollt den Bildschirm durch Veränderung der Displaylist. Bei einer durch einen GRAPHICS- Aufruf erzeugten Displaylist ist die Adresse des Bildschirmspeichers immer im fünften (LSB) und sechsten (MSB) Byte vom Anfang der Display List aus zu finden. Ändert man nun diese Adresse um jeweils die Anzahl der Bytes pro Zeile des benutzten Grafikmodus (im Beispiel 20 Bytes), so erzeugt man vertikales Scrolling. Obgleich es nicht unbedingt einen ästhetischen Genuss verspricht, wird dabei der Einfachheit halber das O.S. ROM von $E000 bis $FFFF als Bildspeicher missbraucht, über das Sie mit einem Joystick in Port 0 hinweg scrollen können.

Warum das mit diesem Programm erzeugte Scrolling nicht gerade einzigartig ist, hat zweierlei Gründe. Erstens haben wir nur Grob-Scrolling verwendet, so dass die Bewegung recht ruckartig wirkt. Zweitens müssen gerade beim Scrolling alle Veränderungen der Bildschirmgrafik immer im Gleichschritt mit der Bilderzeugung des Computers (und des Fernsehgeräten bzw. Monitors) sein, sonst treten vielerlei Störungen in der Grafik auf. Diesen zweiten Punkt können wir im Rahmen eines Basic-Programmes nur schwerlich erfüllen. Mit anderen Worten: hier muss Maschinensprache her.

Feinheiten

Wenn wir nun schon mal bei Assembler gelandet sind, dann machen wir auch Nägel mit Köpfen und gehen gleich zu Fein-Scrolling über. Dazu sind zwei weitere Schritte notwendig: In der Displaylist muss bei jeder ANTIC-Anweisung das Vscrol-Bit (Bit Nr. 5) gesetzt sein. Wir werden daher diesmal keine durch GRAPHICS erzeugte Displaylist, sondern eine "handgestrickte" verwenden. Die Feinverschiebung des Bildes wird dann durch Einschreiben eines Wertes von 0-15 in das VSCROL-Register ($D405) von ANTIC erreicht . Wie Sie richtig vermuten, ist damit eine maximale Bild Verschiebung von 15 Pixel möglich, etwa genau so weit, wie ein GRAPHICS-2-Zeichen hoch ist. Eine Fortsetzung des Scroll Vorganges ist dann mit Hilfe des Grob-Scrolling, also der Veränderung der Bildschirmadresse (der sog. LMS-Bytes, Load Memory Scan-Bytes) möglich. Konkret bedeutet dies, dass zum pixelweisen Scrolling über einen größeren Speicherbereich Grob- und Fein-Scrolling kombiniert angewendet werden müssen.

Scrolling im VBI

Ein Beispiel für den gerade geschilderten Sachverhalt können Sie dem Assembler Listing entnehmen. Wer die Assemblerecke bisher schon verfolgt hat, der wird dort einen alten Bekannten wiederfinden: den Vertical Blank Interrupt, der schon beim PM-Helfer zum Einsatz kam. Qualitativ hochwertiges Scrolling ist nur durch den VBI möglich, da wie schon erwähnt, unbedingt der Gleichtakt zur Bilderzeugung eingehalten werden muss. Grafikänderungen während des VBI führen bekanntlich nie zu Störungen, da während dieser Zeit kein Bild ausgegeben wird, Die VSINIT-Routine trägt zunächst die anfängliche Bildschirmadresse in die "handgestrickte" Displaylist ein, aktiviert die neue Displaylist. setzt das VSCROL- Register auf 0 und fügt die Routine VSVBI in den VBI ein. Da es sich bei VSCROL um ein Write-only Register handelt, wird ein SVSCR Schatten-Register eingerichtet, das immer am Ende des VBI in das Hardwareregister übertragen wird.

Das VBI-Programm, das von nun an 50 mal in der Sekunde durchlaufen wird, fragt zunächst den Joystick Nr.0 ab. Wird dieser nach oben gedrückt, kommt das Unterprogramm OBEN zum Zuge, umgekehrt ist das Unterprogramm UNTEN dran. Beide Programmteile enthalten die gleiche Strategie, so dass es genügt, eines davon näher zu erläutern, im Unterprogramm OBEN wird zunächst geprüft, ob eine Zeichengrenze erreicht ist, d.h. ob das Feinscroll-Register den Wert 0 besitzt. Nur in diesem Fall muss Überprüft werden, ob das Ende des Scroll Bereichs erreicht wurde. Ist das nicht der Fall, so wird das Feinscroll-Register um eins erhöht und damit der Bildschirm um ein Pixel nach oben verschoben. Falls SVSCR nun größer als VMAX geworden ist, dann reicht die Fein Verschiebung nicht aus, es muß also grob gescrollt werden. Dazu wird zur Adresse des Bildschirmspeichers die Länge einer Zeile addiert, wodurch die bisher zweite Zeile an die erste Position rückt.

Gähnende Leere *

Haben Sie das Assemblerlisting eingetippt und starten es an der Adresse $0601. dann sehen Sie - nichts! Das ist auch nicht weiter verwunderlich , denn der als Bildschirmspeicher verwendete Bereich von $7000 bis $7FFF ist ja noch jungfräulich unbenutzt. Sie können diesen Speicherbereich nun z. B. mit dem Fill -Befehl des Maschinensprache-Monitors füllen. Oder Sie benutzen das BASIC-Listing 3, das außer dem Basic-Loader für das Maschinenprogramm noch einen Programmteil zur Erzeugung einer Zufallslandschaft enthält, durch die Sie mittels eines Joysticks hindurch scrollen können. In diesem Programm können Sie übrigens gut beobachten, wie Basic- und Maschinenprogramm gleichzeitig laufen:

Während Sie sich durch das Terrain bewegen, werden Sie vom Basic eingeholt, das immerzu neue Landschaften generiert.

Stellen Sie sich nun vor, das Scrolling Programm in geeigneter Weise mit dem PM-Helfer der März-CK zu kombinieren. Damit hätten Sie dann ein Programm wie ”Caverns of Mars” schon fast fertig - die Assembler Ecke macht es möglich. Allerdings und das muß zur Dämpfung der Euphorie gleich gesagt werden, ist dies nur mit den Assemblerprogrammen und nicht mit den Basic-Loadern machbar. In einer der nächsten Ausgaben werden wir uns im zweiten Teil mit einigen Feinheiten sowie mit horizontalem Scrolling beschäftigen, bis dahin - Tschüß!

Peter Finzel

** Das unten abgedruckte Assemblerlisting wurde von mir so angepasst, dass ein Zufallsmuster in den Scrolling-Bildspeicher geschrieben wird und es eigenständig lauffähig ist. Es muss also nicht aus einem BASIC-Programm aufgerufen werden.*

Das ursprüngliche Assembler-Listing wie im Original-Artikel abgedruckt wird aber natürlich weiterhin im BASIC-Listing 3 verwendet.

Listing 1

```
100 REM * Listing 1:
110 REM *     DEMO fuer vertikales
120 REM *           Grobscrolling
130 GRAPHICS 1
150 ? :? "SCROLL-DEMO/ GRAPHICS 1"
160 LMS=PEEK(560)+PEEK(561)*256+4:REM LMS-Adresse (Display-List)
170 S=57344:REM *          $E000 (OS-ROM)
180 Z=20:REM *             GR.1 Zeile ist 20 Bytes lang
190 BA=57344:BE=65295:REM Anfang & Ende des Scrollbereiches
200 IF STICK(0)=13 THEN S=S-Z:IF S<BA THEN S=BA
210 IF STICK(0)=14 THEN S=S+Z:IF S>BE THEN S=BE
220 H=INT(S/256):REM *     Anfangsadresse des
230 L=S-H*256:REM *        Bildschirmspeichers festlegen
240 POKE LMS,L:POKE LMS+1,H
290 GOTO 200
```

```
100 REM * Listing 1:
110 REM *     DEMO fuer vertikales
120 REM *           Grobscrolling
130 GRAPHICS 1
150 ? :? "SCROLL-DEMO/ GRAPHICS 1"
160 LMS=PEEK(560)+PEEK(561)*256+4:REM LMS-ADRESE (Display-List)
170 S=57344:REM *          $E000 (OS-ROM)
180 Z=20:REM *             GR.1 Zeile ist 20 Bytes lang
190 BA=57344:BE=65295:REM Anfang und Ende des Scrollbereichs
200 IF STICK(0)=13 THEN S=S-Z:IF S<BA THEN S=BA
210 IF STICK(0)=14 THEN S=S+Z:IF S>BE THEN S=BE
220 H=INT(S/256):REM *     Anfangsadresse des
230 L=S-H*256:REM *        Bildschirmspeichers festlegen
240 POKE LMS,L:POKE LMS+1,H
290 GOTO 200
```

```
**************************************************
* LISTING2 (ATMAS-FORMAT)
* VERTIKALES FINE-SCROLLING
* PETER FINZEL 1985
**************************************************

YMIN     EQU $A800          SCROLLBEREICH-ANFANG
YMAX     EQU $A800+$1E0     SCROLLBEREICH-ENDE (24 ZEILEN)
VSMAX    EQU 15             MAX. FINESCROLL WERT
ZLAENG   EQU 20             BYTES PRO ZEILE
MOD      EQU 7+$20          ANTIC-MOSUS 7 + VSCROL BIT

SDLSTL   EQU $0230          DISPLAY LIST ADRESSE
STICK0   EQU $0278          JOYSTICK ABFRAGE
SETVBV   EQU $E45C          INTERRUPT SETZEN VEKTOR
XITVBV   EQU $E462          ABSCHLUSS VBI

VSCROL   EQU $D405          VERTIKALES SCROLL REGISTER
RANDOM   EQU $D20A          ZUFALLSZAHL

         ORG $0600          PROGRAMM STEHT IN PAGE 6

**************************************************
* HAUPTSCHLEIFE BESTEHT AUS INITIALISIERUNG
* UND LEERSCHLEIFE, DA ALLES IM VBI STATTFINDET
**************************************************

         JSR FILL           SCROLLSPEICHER MIT ZUFALLSMUSTER
FUELLEN
         JSR VSINIT         DISPLAYLIST UND VBI EINRICHTEN
LOOP     JMP LOOP           DAUERSCHLEIFE (FINET ALLES IM VBI
STATT)

**************************************************
* DISPLALIST FUER GR.2 SCREEN
* UND EINE VARIABLE
**************************************************

DLIST    DFB $70,$70,$70    DISPLAYLIST: 3X8 LEERZEILEN
         DFB MOD+$40        MODEZEILE MIT V-SCROLL UND LMS
LMSADR   DFW YMIN           STARTE BILDSPEICHER = START SCROLL-
SPEICHER
         DFB MOD,MOD,MOD    11 ZEILEN GR.2 MIT VSCROLL-BIT
         DFB MOD,MOD,MOD,MOD
         DFB MOD,MOD,MOD,MOD
         DFB MOD-$20        LETZTE GR.2 ZEILE BRAUCHT KEIN VS-
CROLL
         DFB $41            SPRUNG ZUM ANFANG DER DISPLAYLIST
         DFW DLIST

SVSCR    DFB 0              SHADOW-VARIABLE FR VSCROL
```

```
**************************************************
* INTITIALISIERUNG DER EIGENEN DISPLAYLIST
* UND DER VBI-ROUTINE
**************************************************

VSINIT   LDA #YMIN:L       LMS ADRESSE AUF ANFANG DES SCROLLBE-
REICHS
         STA LMSADR        SCHREIBEN. ERST LSB
         LDA #YMIN:H       DANN MSB
         STA LMSADR+1
         LDA #DLIST:L      NEUE DISPLAY-LIST NUTZEN
         STA SDLSTL        ERST LSB
         LDA #DLIST:H      DANN MSB
         STA SDLSTL+1
         LDA #0            VSCROL UND SEIN SCHATTENREGISTER AUF
NULL
         STA VSCROL
         STA SVSCR
         LDY #VSVBI:L      UNSEREN EIGENEN VBI AKTIVIEREN
         LDX #VSVBI:H
         LDA #7
         JSR SETVBV
         RTS               UND ZURCK IN DIE HAUPTSCHLEIFE

**************************************************
* EIGEN VBI ROUTINE FUER VERTIKALES SCROLLING
* MIT JOYSTICK-STEUERUNG (HOCH-RUNTER)
**************************************************

VSVBI    LDA STICK0        JOYSTICK ABFRAGEN
         AND #1            NACH OBEN?
         BNE TSTUNT        NEIN -> DANN NACH UNTEN ABFRAGEN
         JSR OBEN          BILDSCHIRM NACH OBEN SCROLLEN
         JMP VBIEND        VBI KANN VERLASSEN WERDEN
TSTUNT   LDA STICK0        JOYSTICK ABFRAGEN
         AND #2            NACH UNTEN?
         BNE VBIEND        AUCH NICHT -> VBI VERLASSEN
         JSR UNTEN         BILDSCHIRM NACH UNTEN SCROLLEN
VBIEND   LDA SVSCR         SCHATENREGISTER SCROLLING
         STA VSCROL        IN HARDWAREREGISTER SCHREIBEN
         JMP XITVBV        EIGENE VBI ROUTINE VERLASSEN

**************************************************
* UP: BILDSCHIRM NACH OBEN SCROLLEN
**************************************************

OBEN     LDA SVSCR         SOFTSCROLL AM ANSCHLAG?
         BNE OB1           NEIN -> KEIN PRUEFUNG NOETIG
         LDA LMSADR        ENDE DES SCROLLBEREICHS ERREICHT?
         CMP #YMAX:L       LSB PRUEFEN
         BNE OB1           NEIN-> DANN SCROLLEN
         LDA LMSADR+1      MSB PRUEFEN
         CMP #YMAX:H
         BEQ OBEND         ENDE ERREICHT -> KEIN SCROLLING
OB1      INC SVSCR         FEINSCROLL POSITION ERHOEHEN
         LDA SVSCR
         CMP #VSMAX+1      IST AUCH GROB-SCROLL NOTWENDIG?
         BCC OBEND         NEIN? -> SCROLLEN BEENDET
         LDA #0            FINESCROLL WIEDER AUF NULL
```

```
          STA SVSCR
          CLC                 ADDITION VORBEREITEN
          LDA LMSADR          LMS ADRESSE UM
          ADC #ZLAENG         EINE ZEILE ERHOEHEN
          STA LMSADR          ERST LSB
          BCC OBEND           FALLS NOTWENDIG
          INC LMSADR+1        AUCH MSB
OBEND     RTS                 UP-SCROLL BEENDET

*********************************************
* DOWN: BILDSCHIRM NACH UNTEN SCROLLEN
*********************************************

UNTEN     LDA SVSCR           SOFTSCROLL AM ANSCHLAG?
          BNE UN1             NEIN -> KEIN PRUEFUNG NOETIG
          LDA LMSADR          ENDE DES SCROLLBEREICHS ERREICHT?
          CMP #YMIN:L         LSB PRUEFEN
          BNE UN1             NEIN-> DANN SCROLLEN
          LDA LMSADR+1        MSB PRUEFEN
          CMP #YMIN:H
          BEQ UNEND           ENDE ERREICHT -> KEIN SCROLLING
UN1       DEC SVSCR           FEINSCROLL POSITION VERMINDERN

          BPL UNEND           SOLANGE NICHT 0 -> SCROLL BEENDET
          LDA #VSMAX          FALLS DOCH FINE-SCROLLWERT
          STA SVSCR           WIEDER AUF MAXIMUM
          SEC                 SUBSTRAKTION VORBEREITEN
          LDA LMSADR          LMS ADRESSE UM
          SBC #ZLAENG         EINE ZEILE VERMINDERN
          STA LMSADR          ERST LSB
          BCS UNEND           FALLS NOTWENDIG
          DEC LMSADR+1        AUCH MSB
UNEND     RTS                 UP-SCROLL BEENDET

*********************************************
* DEN SCROLL-SPEICHER MIT ZUFALLSMUSTER
* DAMIT MAN AUCH WAS SIEHT
*********************************************

FILL      LDX #3              3 SEITEN FLLEN
          LDY #0
          LDA #YMIN:L         SCROLLSEITEN-ANFANGSADRESSE
          STA $80             IN ZEROPAGE ADRESSE SPEICHERN
          LDA #YMIN:H
          STA $81
RND1      LDA RANDOM          ZUFALLSZAHL LADEN
          STA ($80),Y         INDIREKT INDIZIERT IN SCROLLSPEICHER
          INY                 INDIZIERUNGSWERT ERHHEN
          BNE RND1            UND WEITER BIS WIEDER BEI NULL
          INC $81             MSB AUCH UM 1 ERHHEN
          DEX
          BNE RND1            BIS 3 SEITEN GESCHRIEBEN SIND
          RTS                 DANN ZURCK
```

Das Listing wurde so angepasst, dass es eigenständig ist und nicht wie im Begleittext beschrieben, aus einem BASIC-Programm aufgerufen werden muss.

Listing 3

```
100 REM * LISTING 3:
110 REM * Feinscrolling in
120 REM *   Maschinensprache
130 REM *     Demo in BASIC    PF'85
150 BA=28672:REM *     Scrollber.:=$7000
160 ZL=20:REM *        Zeilenlaenge
170 POKE 756,226:REM * Graphikzeichen
180 GOSUB 30000:REM *  Maschinenpgm
190 A=USR(1536):REM *  VBI ein!
200 REM * Scroll-Bereich mit Bild fuellen
220 FOR A=BA TO BA+204*ZL STEP ZL
230 E=INT(RND(0)*8)+A
240 FOR I=A TO E:POKE I,10:NEXT I
250 FOR I=E+1 TO E+10:POKE I,92:NEXT I
260 FOR I=E+11 TO A+19:POKE I,8:NEXT I
290 NEXT A:GOTO 200:REM und neu fuellen...
30000 REM * Maschinenprogramm fuer Feinscrolling
30010 S=0:RESTORE 30100
30020 FOR A=1536 TO 1725:READ D:POKE A,D:S=S+D:NEXT A
30030 IF S<>14814 THEN ? "DATEN-FEHLER!":STOP
30090 RETURN
30100 DATA 104,76,26,6,112,112,112,103,0,112,39,39,39,39,39,39,39
30110 DATA 39,39,39,39,7,65,4,6,0,169,0,141,8,6,169,112,141,9,6,169
30120 DATA 4,141,48,2,169,6,141,49,2,169,0,141,5,212,141,25,6,160
30130 DATA 64,162,6,169,7,32,92,228,96,216,173,120,2,41,1,208,6,32
30140 DATA 97,6,76,88,6,173,120,2,41,2,208,3,32,146,6,173,25,6,141
30150 DATA 5,212,76,98,228,173,25,6,208,14,173,8,6,201,0,208,7,173
30160 DATA 9,6,201,127,240,29,238,25,6,173,25,6,201,16,144,19,169
30170 DATA 0,141,25,6,24,173,8,6,105,20,141,8,6,144,3,238,9,6,96,173
30180 DATA 25,6,208,14,173,8,6,201,0,208,7,173,9,6,201,112,240,24
30190 DATA 206,25,6,16,19,169,15,141,25,6,56,173,8,6,233,20,141,8
30200 DATA 6,176,3,206,9,6,96
```

```
100 REM * LISTING 3:
110 REM * FEINSCROLLING IN
120 REM *   MASCHINENSPRACHE
130 REM *     DEMO IND BASIC   PF'85
150 BA=28672
160 ZL=20
170 POKE 756,226
180 GOSUB 30000
190 A=USR(1536)
200 REM * SCROLL-BEREICH MIT BILD FUELLEN
220 FOR A=BA TO BA+204*ZL STEP ZL
230 E=INT(RND(0)*8)+A
240 FOR I=A TO E:POKE I,10:NEXT I
250 FOR I=E+1 TO E+10:POKE I,92:NEXT I
260 FOR I=E+11 TO A+19:POKE I,8:NEXT I
290 NEXT A:GOTO 200
30000 REM * MASCHINEN-UNTERPROGRAMM LADEN
30010 S=0:RESTORE 30100
30020 FOR A=1536 TO 1725:READ D:POKE A,D:S=S+D:NEXT A
30030 IF S<>14814 THEN ? "DATEN-FEHLER!":STOP
30090 RETURN
30100 DATA
104,76,26,6,112,112,112,103,0,112,39,39,39,39,39,39,39
30110 DATA
39,39,39,39,7,65,4,6,0,169,0,141,8,6,169,112,141,9,6,169
30120 DATA
4,141,48,2,169,6,141,49,2,169,0,141,5,212,141,25,6,160
30130 DATA
64,162,6,169,7,32,92,228,96,216,173,120,2,41,1,208,6,32
30140 DATA
97,6,76,88,6,173,120,2,41,2,208,3,32,146,6,173,25,6,141
30150 DATA
5,212,76,98,228,173,25,6,208,14,173,8,6,201,0,208,7,173
30160 DATA
9,6,201,127,240,29,238,25,6,173,25,6,201,16,144,19,169
30170 DATA
0,141,25,6,24,173,8,6,105,20,141,8,6,144,3,238,9,6,96,173
30180 DATA
25,6,208,14,173,8,6,201,0,208,7,173,9,6,201,112,240,24
30190 DATA
206,25,6,16,19,169,15,141,25,6,56,173,8,6,233,20,141,8
30200 DATA 6,176,3,206,9,6,96
```