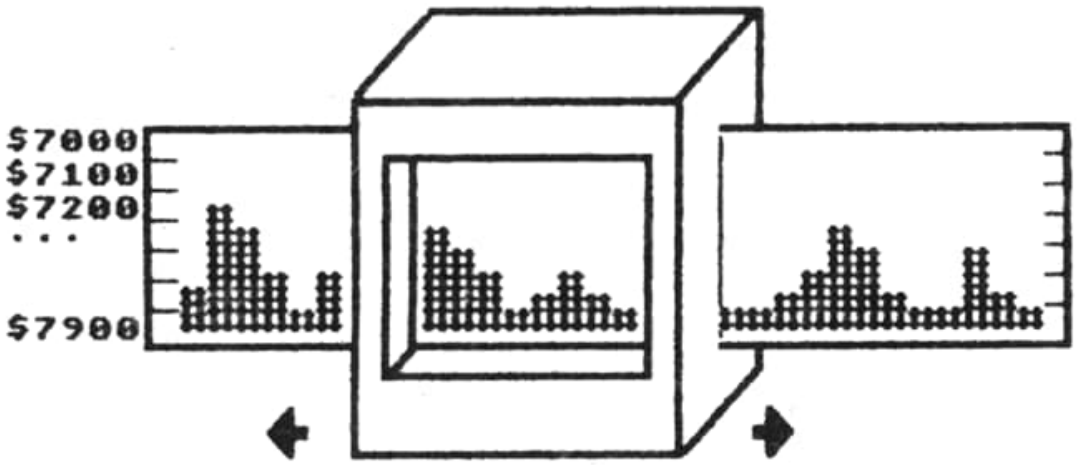
In der Assemblerecke dieses Monats dreht sich wieder alles um Scrolling. Nachdem wir die vertikale Bildverschiebung in der Juni-Ausgabe bereits ausführlich besprochen haben, geht es jetzt um horizontales Scrolling.

RAM und BILD

Beim vertikalen Scrolling konnte die Bildinformation wie gewohnt abgelegt werden, d.h. für einen GRAPHICS 2 Bildschirm 20 Bytes für die erste Zeile, anschließend wieder 20 Bytes für die nächste Zeile usw. Die Erweiterung des Bildschirm-RAMs kam einfach dadurch zustande, daß nicht nur 10 (bzw. 12) Zeilen sondern mehrere hundert solcher Zeilen vorgesehen waren. Die vertikale Grobverschiebung wurde durch Veränderung der Bildschirmadresse um Vielfache der Zeilenlänge erzeugt. Man kann sich unschwer vorstellen, daß mit dieser Anordnung der Bildinformation horizontales Scrolling nicht möglich ist. Erhöht man nämlich die Anfangsadresse des Bildschirms um eins, rutscht die erste Zeile zwar nach links, aber das erste Zeichen der zweiten Zeile taucht dann in der letzten Position der ersten Zeile auf.

Sie sehen, die Anordnung der Bildschirminformation ist beim horizontalen Scrolling von entscheidender Bedeutung. Wie in Bild 1 dargestellt, muß die Information gleich in langen Zeilen



angeordnet werden, um dann später den Bildschirm gleich einem Fenster darüber schieben zu können. Bei einer Zeilenlänge von 256 Zeichen wird ab der Anfangsadresse des Bildschirmspeichers die gesamte erste, 256 Bytes umfassende Zeile abgelegt. Jetzt folgt die ebenfalls 256 Bytes lange zweite Zeile, das heißt wir tun so, als ob der Atari einen Grafikmodus hätte, der 256 Zeichen in einer Zeile darstellen kann.

ANTIC schafft's

Jetzt wird das nächste Problem offensichtlich: Der darzustellende Bildschirmspeicher ist nicht zusammenhängend. Nehmen wir an, das Fenster befinde sich am linken Ende der Zeilen. Die erste Zeile beginnt bei der Adresse $7000 und endet, da ANTIC im Modus 7 (GRAPHICS 2) nur 20 Zeichen pro Zeile darstellen kann, an der Adresse $7013. Die zweite Zeile beginnt aber erst bei Adresse $7100, so daß ein "Loch" von 236 Bytes entstanden ist. Hier zeigt sich, daß

ANTIC im wahrsten Sinne des Wortes "programmierbar" ist: Wir können ein ANTIC-Programm (eine Display-List) schreiben, die jeder einzelnen Bildschirmzeile einen separaten Bildschirmspeicher zuordnet. Dies wird durch Setzen des sogenannten LMS (Load Memory Scan) Bits im ANTIC-Befehl erreicht. Eine ANTIC-Anweisung besteht dann aus dem Befehlscode mit nachfolgender Adresse des gewünschten Bildschirmspeichers im Low/High Format.

Nehmen wir wiederum an, das Fenster befinde sich am linken Ende des darzustellenden Bereiches. Soll das Fenster jetzt um eine Zeichenposition nach rechts gescrollt werden, so muß die Anfangsadresse des Videospeichers jeder Zeile um eins erhöht werden, wobei das zur Vermeidung von Verzerrungen am Schirm in allen 10 Zeilen möglichst gleichzeitig geschehen sollte. Wir wissen aber alle, daß ein Computer nichts gleichzeitig erledigen kann, sondern alles schön der Reihe nach bearbeitet. Aus diesem Grunde sind zwei Maßnahmen notwendig: So muß die Änderung der Adressen im Rahmen eines schnellen Maschinenprogrammes erfolgen, und außerdem sollte dieses nur in der vertikalen Austastlücke (im VBI) benutzt werden, so daß Grafikstörungen ausgeschlossen sind.

Feinscrolling

Wie im ersten Teil dieses Artikels soll auch beim horizontalen Scrolling die Feinverschiebung benutzt werden. Wird das Bit 4 (das HSCOL-Bit) jeder betroffenen ANTIC- Anweisung auf eins gesetzt, so kann diese Zeile durch das HSCROL-Register (Adresse $D404) bis zu 15 Pixel nach rechts verschoben werden. Für längere Scrollwege muß daher wiederum Grob mit Feinscrolling kombiniert werden, und dadurch wird's kompliziert: Wie schon festgestellt, entspricht eine Erhöhung der Zeilenadressen einer Verschiebung des Bildschirmfensters nach rechts. Da aber das Fenster (Ihr Fernsehgerät) im Regelfall ortsfest ist, entspricht dies einer Bewegung des Bildes nach links. Eine Erhöhung des Feinverschiebungsregisters HSCROL bewirkt dagegen eine Verschiebung des Bildes nach rechts, woraus zu folgern ist, daß Grob- und Feinscrolling immer konträr verwendet werden müssen. Eine Verschiebung nach links erfordert eine Verminderung des HSCROL-Registers bei Vergrößerung der Zeilenadressen. Alles klar?

Das Assemblerlisting

Ziel des Assemblerprogrammes soll es sein, zehn GRAPHICS 2 Zeilen über eine Länge von 256 Zeichen horizontal zu scrollen. Der Aufbau des Assemblerlistings entspricht im wesentlichen dem Programm des ersten Teils. Dabei wird wieder eine handgestrickte Display-List und der "deferred" VBI verwendet. Den Anfang bilden einige Definitionen. Unter anderem wird der Bildschirmspeicher beginnend ab der Adresse $7000 definiert. Das Programm selbst liegt in Page 6 und wird mit X=USR (1536) aufgerufen. Jetzt folgt die Display-List, die, wie bereits geschildert, aus ANTIC-Anweisungen mit gesetzten LMS- und HSCROL-Bits und den Zeilenadressen besteht. Zwei Variablen halten die momentane Scroll-Position fest:

SHSCR dient als Schattenregister des HSCROL-Registers (das selbst nur geschrieben aber nicht gelesen werden kann). GRBSCR speichert die Entfernung des Fensters vom Zeilenanfang in Zeichen und ist daher für das Grobscrolling verantwortlich.

VSINIT initialisiert die Display-List und das HSCROL-Register so, daß das Fenster ganz links zu liegen kommt. Die Display-List wird aktiviert und die Scroll-Routine HSVBI in den "deferred" VBI eingebunden. Das VBI-Programm testet, ob Joystick Null nach links oder rechts bewegt wurde und ruft dementsprechend die Unterprogramme LINKS und RECHTS auf. Nach getaner Arbeit wird noch das Schattenregister SHSCR ins Hardwareregister HSCROL kopiert und der VBI über XITVBV verlassen.

Im Unterprogramm LINKS wird zuerst abgefragt, ob der Rand des Scrollbereiches erreicht ist, der mit GRBMAX gleich 233 die Zeilenlänge minus der Bildschirmlänge ist. Nun wird das Schattenregister der Feinposition vermindert. Falls Feinscrolling nicht ausreicht, wird die Grobposition um eins erhöht. Die Zeilenadressen selbst werden durch das Unterprogramm SETLMS erzeugt, was sich durch die gewählte Zeilenlänge von 256 Bytes recht einfach gestaltet. SETLMS braucht nur die Grobposition aus GRBSCR zu entnehmen und in den niederwertigen Teil der Zeilenadresse einzutragen. Bei der Verwendung von anderen Zeilenlängen muß dieser Programmteil verändert bzw. erweitert werden. Es ist meist zweckmäßig, eine Tabelle der Zeilenanfänge anzulegen, dazu jeweils den Offset des Fensters zu addieren und den so errechneten Wert als LMS-Adresse in die Display-List einzutragen.

Das Unterprogramm RECHTS arbeitet nach dem gleichen Prinzip, nur wird eben in die andere Richtung gescrollt. Ein Demo des Programmes können Sie dem Listing 2 entnehmen. Hier wird eine Zufalls-Landschaft entworfen, die mit einem Joystick in Port 1 horizontal gescrollt werden kann. Wenn Sie das Programm nach eigenen Vorstellungen modifizieren wollen, sollten sie jedoch das mit MAC/65 (die Editor-Assembler Cartridge tut’s auch) verfasste Assemblerlisting eintippen.

Ein paar Kniffe

Durch das Einschalten der horizontalen Feinverschiebung verändert ANTIC selbsttätig die Anzahl der gelesenen Bytes pro Zeile. Wenn Sie mit dem GRAPHICS 2 Modus arbeiten, bei dem ANTIC normalerweise 20 Bytes pro Zeile holt, werden beim Einschalten des Feinscrolling 24 Bytes gelesen (aber nicht dargestellt). Das ist ja auch bitter nötig, wenn das erste Zeichen nur noch halb sichtbar ist, muß ANTIC ja wissen, welches Zeichen an der 21. Position dargestellt werden soll. Dieses Verhalten führt aber auch dazu, daß die ersten beiden Spalten außerhalb der Sichtweite bleiben. Einen gewissen Ausgleich bietet hier der ”breite” Bildschirmmodus (POKE 559,35), der Scrolling in Cinema-Scope bietet. Hier zeigt sich wieder einmal, daß der Atari in Grafikangelegenheiten der Rolls-Royce unter den Home-Computern ist. Andere Computer (welche wohl?) lösen das Problem auf eine recht simple Art: Der Bildschirm wird um zwei Spalten eingeengt, so daß beim horizontalen Scrolling genügend Bildschirminformation vorhanden ist.

Ein zweites Problem tritt an 4KByte-Grenzen auf: ANTICs interner Adresszähler ist nur 12 Bit breit und kann daher 4K-Grenzen nicht überspringen. Sollte jedoch ein Videospeicher von mehr als 4KByte Länge erforderlich sein, kann man sich recht gut helfen, indem man als Zeilenlänge eine Zweierpotenz (64, 128 oder wie im Beispiel 256) wählt. Falls das nicht möglich ist, kann man die Zeile, in der die 4K-Grenze auftritt, durch geschickte Wahl der Zeilenanfänge einfach überspringen. Der Videospeicher hat dann zwar „Löcher”, aber dieses Problem kann man mit Tabellen ganz gut in den Griff bekommen. Die einzige tatsächliche Beschränkung besagt, daß eine zusammenhängende Zeile nicht länger als 4096 Zeichen sein kann. Und das genügt doch, oder?

Peter Finzel

Das Assembler Listing 1 wurde im Unterschied zu dem Original-Artikel in ATMAS-II Assembler übersetzt und um eine Routine ergänzt, so dass es alleine (ohne BASIC) lauffähig ist. Die eigentlichen Routinen wurden aber nicht geändert.

```
*****************************************
* LISTING 1:
*
* HORIZONTALES FEIN-SCROLLING
* IN MASCHINENSPRACHE
*
* PETER FINZEL 85
*****************************************
*
* KONSTANTE
*
BASIS    EQU $5000
HSMAX    EQU 7              ;MAXIMAL VERSCHIEBUNG FEINSCROLLING
ZEILE    EQU 256            ;ZEILENLAENGE IN BYTE
ZZAHL    EQU 10             ;ANZAHL DER ZEILEN AM SCHIRM
GRBMAX   EQU 233            ;RECHTES ZEILENENDE
MOD      EQU 7+$50          ;ANTIC MODUS 7, HSCROL UND LMS
BASZP    EQU $E0            ;BASIS ADRESSE IN ZP
TMP1     EQU $E2            ;TEMP VARIABLE
*
* OPERATING SYSTEM
*
SDLSTL   EQU $0230          ;SCHATTENREG. DISPLAY-LISTADRESSE
STICK0   EQU $0278          ;SCHATTENREG. F. JOYSTICK NR. 0
SETVBV   EQU $E45C          ;ROUTINE F. INTERRUPTVEKOREN
XITVBV   EQU $E462          ;ABSCHLUSS DES VBI
*
* HARDWARE
*
RANDOM   EQU $D20A          ;ZUFALLSZAHL
HSCROL   EQU $D404          ;REGISTER HOR.VERSCHIEBUNG
*
*****************************************
* PROGRAMM-EINSPRUNG
*****************************************
*
         ORG $A800          ;WIE IMMER IN $A800
*
         JMP HSINIT         ;SPRUNG ZUM MP-ANFANG
*
*****************************************
* DISPLAY-LIST FUER HORIZONTAL
* SCROLLENDES 'GRAPHICS 2'-DISPLAY
* 10 ZEILEN ANTIC-MODUS 7
*****************************************
*
DLIST    DFB $70,$70,$70    ;3 LEERZEILEN
         DFB MOD            ;ANTIC-BEFEHL
LMSADR   DFW BASIS          ;ADRESSE BILDSPEICHER
         DFB MOD
         DFW BASIS+ZEILE
         DFB MOD
         DFW BASIS+(ZEILE*2)
         DFB MOD
         DFW BASIS+(ZEILE*3)
         DFB MOD
         DFW BASIS+(ZEILE*4)
         DFB MOD
         DFW BASIS+(ZEILE*5)
         DFB MOD
         DFW BASIS+(ZEILE*6)
         DFB MOD
```

```
          DFW BASIS+(ZEILE*7)
          DFB MOD
          DFW BASIS+(ZEILE*8)
          DFB MOD
          DFW BASIS+(ZEILE*9)
          DFB $41             ;ANTIC-JMP-BEFEHL
          DFW DLIST           ;ZUM ANFANG DER DISP.-LIST
*
**************************************
* INTERNE VARIABLE
**************************************
*
SHSCR     DFB 0               ;SCHATTENREGISTER HSCROL
GRBSCR    DFB 0               ;ZAEHLER FUER GROBSCROLLING
*
**************************************
* INITIALISIERUNGSROUTINE ZUM
* EINRICHTEN DES
* BILDSCHIRMHINTERGRUND
**************************************
*
HSINIT    LDA #BASIS:L
          STA BASZP
          LDA #BASIS:H
          STA BASZP+1
          LDY #0
LOOP0     LDA RANDOM
          AND #%00000111      ;0-7
          ASL
          ASL
          ASL
          TAX
          CLC
          ADC #8
          STA TMP1
LOOP1     LDA MUSTER,X
          STA (BASZP),Y
          INC BASZP+1
          INX
          CPX TMP1
          BNE LOOP1
          LDA #BASIS:H
          STA BASZP+1
          INY
          BNE LOOP0
*
**************************************
* INITIALISIERUNGSROUTINE ZUM
* EINRICHTEN DER NEUEN DISP.-LIST
* UND DER VBI -ROUTINE
* UND DEM BILDSCHIRMHINTERGRUND
**************************************
*
          LDA #0              ;GROBSCROLLING
          STA GRBSCR          ;AUF NULL
          JSR SETLMS          ;UND DISPLAYLIST VORBEREITEN
          LDA #DLIST:L        ;NEUE DISP.-LIST
          STA SDLSTL          ;EINSCHALTEN
          LDA #DLIST:H
          STA SDLSTL+1
          LDA #0              ;FEIN-SCROLLING IN AUSGANGS-
          STA HSCROL          ;POSITION SETZEN
          STA SHSCR           ;AUCH SCHATTENREGISTER!
          LDY #HSVBI:L        ;SCROLL-ROUTINE IN
          LDX #HSVBI:H        ;DEN VBI EINFUEGEN
          LDA #7              ;DEFERRED VBI GENUEGT
          JSR SETVBV          ;OHNE KOMMENTAR...
```

```
ENDLESS  JMP ENDLESS
*
**************************************
* VBI-ROUTINE F. HOR. SCROLLING
**************************************
*
HSVBI    CLD                ;6502 RECHNET BINAER
         LDA STICK0         ;JOYSTICK 0
         AND #8             ;NACH RECHTS?
         BNE TLINKS         ;NEIN, VERSUCHE LINKS - >
*
         JSR RECHTS         ;BILDSCHIRM NACH RECHTS
         JMP VBIEND         ;FERTIG !===>
*
TLINKS   LDA STICK0
         AND #4             ;NACH LINKS GEZOGEN?
         BNE VBIEND         ;NEIN, KEIN SCROLLING!
*
         JSR LINKS          ;NACH LINKS SCROLLEN
*
VBIEND   LDA SHSCR          ;SCHATTENREGISTER FEINSCROLL
         STA HSCROL         ;IN HARDWAREREG. UEBERTRAGEN
         JMP XITVBV         ;SCHLUSS FUER HEUTE!
*
**************************************
* UP BILDSCHIRM NACH LINKS
**************************************
*
LINKS    LDA SHSCR          ;ZEICHENGRENZE ERREICHT?
         BNE LNK1           ;NEIN - >
*
         LDA GRBSCR         ;WEITERES SCROLLING
         CMP #GRBMAX        ;NACH LINKS MOEGLICH?
         BCS LNKEND         ;NEIN, KEIN SCROLLING ->
LNK1     DEC SHSCR          ;EIN PIXEL NACH LINKS
         LDA SHSCR          ;REICHT FEINSCROLLING?
         BPL LNKEND         ;JAWOHL, DANN FERTIG
*
         LDA #HSMAX         ;FEINVERSCHIEBUNG ZURUECK
         STA SHSCR
         INC GRBSCR         ;GROBPOSITION WEITER
         JSR SETLMS         ;IN DISPLAYLIST EINTRAGEN
LNKEND   RTS
*
**************************************
* UP BILDSCHIRM NACH RECHTS
**************************************
*
RECHTS   LDA SHSCR          ;ZEICHENGRENZE ERREICHT?
         BNE RHT1           ;NEIN - >
*
         LDA GRBSCR         ;RAND ERREICHT?
         BEQ RHTEND         ;JA, KEIN SCROLLING ->
RHT1     INC SHSCR          ;EIN PIXEL RECHTS
         LDA SHSCR          ;FEINSCROLLING AUSREICHEND?
         CMP #HSMAX+1
         BCC RHTEND
*
         LDA #0             ;FEINSCROLLING
         STA SHSCR          ;RUECKSETZEN
         DEC GRBSCR         ;GROB-POS. ZURUECK
         JSR SETLMS         ;IN DISP.-LIST EINTRAGEN
RHTEND   RTS
*
**************************************
* UP EINTRAGEN DER LMS ADRESSE
**************************************
```

```
*
SETLMS   LDX #0             ;INDEX IN DISP.-LIST
         LDA GRBSCR         ;GROBSCROLL-POSITION
NXTLMS   STA LMSADR,X       ;IN ZWEITEN BYTE
         INX                ;JEDER ANTIC-ANWEISUNG
         INX                ;EINTRAGEN
         INX
         CPX #ZZAHL*3       ;ALLE ZEILEN BEARBEITET?
         BCC NXTLMS         ;NOCH NICHT - >
         RTS
*
*****************************************
* MUSTERTABELLE FUER ZUFALLSHINTERGRUND
*****************************************
*
MUSTER   DFB 0,0,0,0,0,0,0,67
         DFB 0,0,0,0,0,0,3,67
         DFB 0,0,0,0,0,3,3,67
         DFB 0,0,0,0,3,3,3,67
         DFB 0,0,0,3,3,3,3,67
         DFB 0,0,3,3,3,3,3,67
         DFB 0,3,3,3,3,3,3,67
         DFB 3,3,3,3,3,3,3,67
```

```
100 REM * DEMO-Horizontales Scrolling
110 REM * Peter Finzel 1985
120 REM *
130 GOSUB 1000:REM  * Maschinenpgm.
200 A=USR(1536):REM * Scrolling ein
300 REM LANDSCHAFT ENTWERFEN
310 BASIS=28672
320 FOR I=0 TO 255
330 A=INT(RND(0)*8)+1
340 FOR S=0 TO A:POKE BASIS+S*256+I,0:NEXT S
350 FOR S=A+1 TO 8:POKE BASIS+S*256+I,3:NEXT S
360 POKE BASIS+9*256+I,67
390 NEXT I
400 GOTO 400
1000 REM * Maschinenprogramm...
1010 S=0:RESTORE 1100
1020 FOR A=1536 TO 1726:READ D:POKE A,D:S=S+D:NEXT A
1030 IF S<>15960 THEN ? "DATEN-FEHLER!":STOP
1090 RETURN
1100 DATA 104,76,42,6,112,112,112,87,0,112,87,0,113,87,0,114,87,0
1110 DATA 115,87,0,116,87,0,117,87,0,118,87,0,119,87,0,120,87,0,121
1120 DATA 65,4,6,0,0,169,0,141,41,6,32,175,6,169,4,141,48,2,169,6
1130 DATA 141,49,2,169,0,141,4,212,141,40,6,160,78,162,6,169,7,32
1140 DATA 92,228,96,216,173,120,2,41,8,208,6,32,143,6,76,102,6,173
1150 DATA 120,2,41,4,208,3,32,111,6,173,40,6,141,4,212,76,98,228,173
1160 DATA 40,6,208,7,173,41,6,201,233,176,19,206,40,6,173,40,6,16
1170 DATA 11,169,7,141,40,6,238,41,6,32,175,6,96,173,40,6,208,5,173
1180 DATA 41,6,240,21,238,40,6,173,40,6,201,8,144,11,169,0,141,40
1190 DATA 6,206,41,6,32,175,6,96,162,0,173,41,6,157,8,6,232,232,232
1200 DATA 224,30,144,246,96
```