Zauberer und Monster

In der Serie "Spiele programmieren" hier der zweite Teil zum "Multi-Player-Animator"

Wie schon im letzten Heft angekündigt, geht es in dieser Folge wieder einmal um mehrfarbige Player. Wir stellen Ihnen heute ein Listing vor, das es erlaubt, die mit dem Multiplayer-Animator (kurz MPA) gezeichneten Figuren effektvoll in eigenen Programmen einzusetzen. Und das mit voller Animation und – natürlich – mit mehrfarbigen Objekten.

Ihrer Phantasie sind dabei keine Grenzen gesetzt. Sie können Zauberer und Monster zum Laufen bringen, Helikopter mit drehenden Rotoren darstellen oder auch einen PacMan das Mampfen lehren. Alles, was Sie dazu benötigen, ist der Multiplayer-Animator (von der letzten Leserdiskette LF8-687) und Listing 2 aus diesem Heft. Damit auch die Assembler-Programmierer auf ihre Kosten kommen, haben wir außerdem das Sourcelisting des Animationsprogramms in Listing 1 abgedruckt.

Das Programm wird bequem mit zwei USR-Befehlen von Basic aus gesteuert. Mit Hilfe des ersten werden alle wichtigen Parameter nach und nach an das Animationsprogramm weitergegeben. Dabei handelt es sich um Horizontal- und Vertikalposition sowie Anfangs- und End-Shape der Figur, Ablaufgeschwindigkeit der Animation und die Adresse, unter der die Shapes zu finden sind. Die Position wird in absoluten Player-Koordinaten angegeben. Beachten Sie daher, daß einige Werte die Figur ganz oder teilweise vom Bildschirm verschwinden lassen. Die Position eines Players können Sie z.B. mit folgendem Befehl festlegen:

```
X = USR (MP, XY1, 80, 100)
```

Die Liste der Parameter setzt sich dabei so zusammen: MP ist der Einsprung der USR-Adresse. Der zweite Parameter heißt XY1 und stellt eine Kennung dar, die besagt, welcher Art die nächsten beiden Werte sind, die an das Maschinenprogramm übergeben werden. Für das Kürzel XY1 z.B. wären dies die X- und Y-Positionen für Multicolor-Player 1. Bei den beiden restlichen Werten handelt es sich schließlich um die gewünschten Koordinaten, zuerst X und dann Y.

Es ist natürlich sonnenklar, daß MP und XY1 ganz normale Basic-Variablen sind, deren Werte zuvor unbedingt initialisiert werden müssen. Schreibfaule können natürlich auch gleich Zahlenwerte im USR-Kommando verwenden, jedoch sind Variablen aufgrund der besseren Lesbarkeit vorzuziehen. Neben XY1 gibt es noch eine Reihe weiterer Befehlskürzel. Hier eine Zusammenstellung:

XY1 (0) Position Figur 1
XY2 (1) Position Figur 2
AE1 (2) Anfangs- und End-Shape Figur 1
AE2 (3) Anfangs- und End-Shape Figur 2
GS1 (4) Geschwindigkeit und Page der
GS2 (5) Shapes für Figur 1/2

Die AE-Anweisungen legen fest, welche Shapes zur Animation herangezogen werden. Wie Sie wohl noch aus der letzten Folge wissen, enthält ein mit MPA erzeugtes File insgesamt acht Shapes, daher werden die entsprechenden Shape-Nummern immer im Bereich von 0 bis 7 erwartet. Dazu ein Beispiel:

```
X = USR (MP, AE2, 0, 3)
```

Dieser Befehl bewirkt, daß nacheinander die Shapes 0, 1, 2, 3 angezeigt werden; dann geht es wieder bei der 0 weiter. Die Anweisung

```
X = USR (MP, GS2, 5, 128)
```

legt dagegen fest, daß die Animation mit der Geschwindigkeitsstufe 5 abläuft und die acht Shapes in Page 128 (d.h. ab Adresse 128*256 = 32768) zu finden sind. Bedingt durch die Hardware des Atari, können maximal zwei voneinander unabhängige Figuren gleichzeitig am Bildschirm dargestellt werden. Aus diesem Grund existiert auch jeder der Befehle für Player 1 und Player 2.

```
30100 DATA 76,32,6,76,51,6,0,0,0,0,0,0
.0,0,0,0,0,0,0,0,1,4,5,8,9,0,0                    B:YO
30110 DATA 0,0,0,0,0,0,0,104,104,104,170
.169,18,6,170,104,104,157,6                       B:ZY
30120 DATA 6,104,104,157,8,6,96,216,10
4,104,104,141,7,212,24,105,4                      B:LX
30130 DATA 141,24,6,133,205,165,2,141,
25,6,169,0,133,204,162,4,168                      B:MO
30140 DATA 145,204,136,208,251,230,205
.202,208,246,169,17,141,111                       B:BQ
30150 DATA 2,169,56,141,47,2,169,2,141
.29,208,162,1,189,14,6,157,28                     B:FP
30160 DATA 6,189,10,6,157,26,6,202,16,
241,160,130,162,6,169,7,32,92                     B:CA
30170 DATA 228,96,162,0,32,154,6,141,0
,208,141,1,208,232,32,154,6                       B:ZV
30180 DATA 141,2,208,141,3,208,76,98,2
28,222,20,6,208,23,189,14,6                       B:FV
30190 DATA 157,20,6,254,26,6,189,12,6,
221,26,6,176,6,189,10,6,157                       B:FT
30200 DATA 26,6,189,26,6,10,10,10,10,1
33,204,189,16,6,133,205,189                       B:CB
30210 DATA 24,6,133,207,32,225,6,165,2
04,9,128,133,204,230,207,32                       B:AB
30220 DATA 225,6,189,8,6,157,30,6,189,
6,6,96,189,30,6,133,206,169                       B:HJ
30230 DATA 0,160,15,145,206,136,16,251
,189,8,6,133,206,160,15,177                       B:ER
30240 DATA 204,145,206,136,16,249,96              B:CI
```

ATMAS II – Sourcetext Listing 2

```
*****************************************
*  Animation von Multicolor-Players
*             die mit
*       MULTIPLAYER-ANIMATOR
*         gezeichnet wurden
*
* P. Finzel 1987  Assembler: ATMAS-II
*****************************************

HPOSP0   EQU $D000 Hor.-Position
GRACTL   EQU $D01D Graphik-Kontrollreg.
PMBASE   EQU $D407 PM-Basisadresse
SDMCTL   EQU $22F  DMA-Kontrollreg.
GPRIOR   EQU $26F  Prioritaeten
SETVBV   EQU $E45C Interrupt einfuegen
XITVBV   EQU $E462 Ende des Interrupts
*
         ORG $600  in PAGE 6
*
* Einsprung-Tabelle
*
         JMP USER  Werte uebergeben
         JMP START Animation starten
*
* Uebergabevariable
*
X1POS    DFB 0     X-Positionen
X2POS    DFB 0
Y1POS    DFB 0     Y-Positionen
Y2POS    DFB 0
ANF1     DFB 0     Anfangs-Shapes
ANF2     DFB 0
END1     DFB 0     End-Shapes
END2     DFB 0
GES1     DFB 0     Animations-Geschw.
GES2     DFB 0
SHPAS1   DFB 0     Page-Adresse der
SHPAS2   DFB 0     Shapetabellen
*
* Offsets zum Eintragen (fuer USR)
*
TAB      DFB 0,1,4,5,8,9

* Interne Variablen:

PMPAS1   DFB 0     Page-Adresse Player 1
PMPAS2   DFB 0     Page-Adresse Player 2
SHP1     DFB 0     aktuelle Shapes
SHP2     DFB 0
VERI1    DFB 0     Zaehler fuer Verzoegerung
VERI2    DFB 0
Y1ALT    DFB 0     alte Y-Positionen
Y2ALT    DFB 0
*
* Zero-Page Register
*
ZREG1    EQU $CC
ZREG2    EQU $CE
+-------------------------------------
* Routine zur Uebergabe der Parameter
+-------------------------------------
USER     PLA
         PLA       welche Parameter
         PLA       werden uebergeben?
         TAX       Ort des Eintrages
         LDA TAB,X aus Tabelle
         TAX
         PLA       ersten Parameter
         PLA       eintragen
         STA X1POS,X
         PLA       und zweiten Parameter
         PLA
         STA Y1POS,X
         RTS

+-------------------------------------
* Vorbereitungsroutine
*    - loescht PM-Bereich
*    - schaltet PM-Graphik ein
*    - aktiviert VBI
+-------------------------------------
```

Das zweite USR-Kommando an der Einsprungstelle MPSTART = 1539 dient dazu, die PM-Grafik vorzubereiten und das Animationsprogramm in Arbeitsbereitschaft zu versetzen. Das wird durch Einbindung eines Teils des Programms in den VBI erreicht. Der Aufruf kann z.B. lauten:

X = USR (MPSTART, 128)

MPSTART ist wiederum der Einsprung, 128 bezeichnet die Adresse (Page-Nummer!) des PM-Speichers. Im genannten Fall wäre das wieder die Adresse 32768. Bedenken Sie aber, daß die ersten drei Pages nach dieser Adresse von der Player-Hardware nicht verwendet werden und daher wunderbar Platz für die MPA-Shapes bieten. Falls Sie die Missiles nicht verwenden, stehen sogar ganze vier Pages zur Verfügung.

Listing 2 zeigt, wie man mit diesen USR-Kommandos sinnvoll umgeht. Wie immer ist auch ein kleines Beispiel integriert, das Sie natürlich für eigene Werke nicht übernehmen müssen. Am Anfang des Programms wird nach Initialisierung der Variablen mit Hilfe eines OPEN-Befehls und einer kleinen Schleife ein MPA-Shapefile von der Diskette geladen. Im Beispiel kommt ZAUBERER.MPA zum Einsatz, ein File, das zusammen mit dem MP-Animator auf der letzten Leserdiskette zu finden ist. Auf der Disk zu diesem Heft (LF8-1/88) sind übrigens Listing 1 und 2 sowie das MPA-File abgespeichert.

Nach dem Ladevorgang werden die Farben der Player durch POKE-Befehle festgelegt und die Voreinstellungen für Position, Shapes und Ablaufgeschwindigkeit getroffen. Im vorliegenden Fall kommen die Shapes 0 bis 3 für Figur 1 und die Shapes 4 bis 7 für Figur 2 zum Einsatz. Es ist natürlich auch möglich, mehrere MPA-Shapefiles zu laden und für jeden Multicolor-Player ein separates File zu verwenden.

Schließlich kann das Animationsprogramm eingeschaltet werden. Eine Schleife sorgt dafür, daß sich die beiden Figuren über den Bildschirm bewegen. Sieht doch gar nicht übel aus, oder?

Das Treiberprogramm für die Animation benötigt Page 6 ($0600 bis $06FF) fast bis zum letzten Byte. Darüber hinaus muß aber nur noch Speicherplatz für die PM-Grafik und die MPA-Shapes freigehalten werden. Wenn das Basic-Listing nicht allzu lang ist (wie zum Beispiel), kann man diese Bereiche leicht zwischen Programm und Videospeicher unterbringen, ohne sich großartig um Reservierung zu kümmern. Das Programm in Listing 2 benutzt dazu den Speicherbereich ab $8000. Selbstverständlich läßt sich das Animationsprogramm auch von Assemblerfreunden verwenden. In diesem Fall kann man die Routinen für die USR-Befehle von Basic einsparen.

Peter Finzel

PS

Listing 1: Basic

```
100 REM ********************************  BV
110 REM * MULTICOLOR-PLAYER ANIMATION    AM
115 REM * FUER PLAYERS, DIE MIT DEM      DO
117 REM * MPA HERGESTELLT WURDEN          PC
120 REM *                                 JK
130 REM * F. FINZEL            1987      FO
140 REM ********************************  MC
150 PCOLR0=704:PMPAG=128                  VK
160 PMADR=PMPAG*256:SHPAG=128             DO
170 MPSTART=1539:MP=1536                  UK
180 XY1=0:XY2=1:AE1=2:AE2=3               KJ
190 GS1=4:GS2=5                           CK
200 REM * Hintergrundgraphik              AA
210 GRAPHICS 2+16:COLOR ASC("-")          DB
220 PLOT 0,4:DRAWTO 19,4:PLOT 0,7         CJ
230 DRAWTO 19,7:GOSUB 30000               JA
240 REM * MPA-File einlesen               ZA
250 OPEN #1,4,0,"D:ZAUBERER.MPA"          ZC
260 FOR I=PMADR TO PMADR+255              MV
270 GET #1,D:POKE I,D:NEXT I              XP
280 POKE PCOLR0,40:POKE PCOLR0+1,132      BH
290 POKE PCOLR0+2,40:POKE PCOLR0+3,132    KJ
300 REM * Animation vorbereiten           QE
310 X=USR(MP,XY1,0,0)                     BD
320 X=USR(MP,XY2,0,0)                     BV
330 X=USR(MP,AE1,0,3)                     ZJ
340 X=USR(MP,AE2,4,7)                     ZX
350 X=USR(MP,GS1,6,128)                   JX
360 X=USR(MP,GS2,5,128)                   JX
390 X=USR(MPSTART,PMPAG)                  EX
400 REM * Figuren bewegen                 LO
410 FOR I=0 TO 205                        LH
420 X=USR(MP,XY1,232-I,80)                BV
430 X=USR(MP,XY2,24+I,134)                AH
450 NEXT I                                HF
490 GOTO 400                              BV
30000 REM * ANIMATIONS-PGM erzeugen       VK
30010 S=0:RESTORE 30100                   LH
30020 FOR A=1536 TO 1789:READ D:POKE A
,D:S=S+D:NEXT A                           SS
30030 IF S<>23103 THEN ? "DATEN-FEHLER
!":STOP                                   HJ
30090 RETURN                              XX
```

```
START   CLD         zur Sicherheit
        PLA
        PLA         wo soll PM-Bereich
        PLA         liegen?
        STA PMBASE
        CLC         Berechnung der Player
        ADC #4      Adressen
        STA PMPAG1
        STA IREG1+1
        ADC #2
        STA PMPAG2
*
* PM-Bereich loeschen
*
        LDA #0
        STA IREG1
        LDX #4      vier Pages loeschen
        TAY
PMCLR   STA (IREG1),Y
        DEY         Schleife zum Loeschen
        BNE PMCLR
        INC IREG1+1
        DEX
        BNE PMCLR
*
* PM-Graphik einschalten
*
        LDA #$11    Multicolor-Option
        STA GPRIOR  aktivieren
        LDA #$3A    Player-DMA ein-
        STA SDMCTL  schalten
        LDA #2      Player Darstellung
        STA GRACTL  einschalten
*
        LDX #1      Interne Variable
VORBER  LDA GES1,X  vorbereiten:
        STA VERZ1,X Verzoegerung
        LDA ANF1,X  und Anfangs-Shape
        STA SHP1,X
        DEX
        BPL VORBER
*
        LDY #PMVBI:L VBI einrichten
        LDX #PMVBI:H
        LDA #7      hier: deferred
        JSR SETVBV  VB-Interrupt
        RTS         fertig!

*-------------------------------------
* Interrupt-Routine fuer PM-Graphik
*-------------------------------------
PMVBI   LDX #0      Multiplayer 1
        JSR PMCOPY  bearbeiten
        STA HPOSP0  Horizontal-Werte
        STA HPOSP0+1 eintragen
        INX         dann kommt Multi-
        JSR PMCOPY  Player 2 dran
        STA HPOSP0+2 X-Wert eintragen
        STA HPOSP0+3
VBIENDE JMP XITVBV  Interrupt beenden

*-------------------------------------
* Animation eines Multiplayers
*   Eingabe:(X) 0:M.-Player 1 bearb.
*               1:M.-Player 2 bearb.
*   Ausgabe: (A): X-Position
*-------------------------------------
PMCOPY  DEC VERZ1,X neues Shape?
        BNE PMC2    nein -->
PMC1    LDA GES1,X  Verzoegerung
        STA VERZ1,X neu einrichten
        INC SHP1,X  naechstes Shape
        LDA END1,X  schon ueberm
        CMP SHP1,X  Ende des Bereiches?
        BCS PMC2    nein -->
        LDA ANF1,X  Anfangsshape
        STA SHP1,X  in Shape-Zaehler

PMC2    LDA SHP1,X  Relative Adresse
        ASL         ides Shapes berechnen
        ASL
        ASL
        ASL
        STA IREG1   in Zeropage-Register
        LDA SHPAG1,X fuer Daten-Quelle
        STA IREG1+1 eintragen
        LDA PMPAG1,X Ziel ist der PM-
        STA IREG2+1 Bereich

        JSR MOVE    Datentransfer Player 1

        LDA IREG1   Adressen fuer
        ORA #128    Ziel und Quelle
        STA IREG1   von Player 2
        INC IREG2+1
        JSR MOVE
        LDA YPOS,X  Alte Position
        STA YALT,X  merken
        LDA XPOS,X  X-Position zurueck-
        RTS         geben
*
*-------------------------------------
* Datentransfer vom Shapespeicher
* in den PM-Bereich
*  (X): Nummer des M.-Players
*-------------------------------------
MOVE    LDA YALT,X  zuerst wird das
        STA IREG2   alte Shape
        LDA #0      geloescht
        LDY #15     Laenge 16 Bytes
LOESCH  STA (IREG2),Y
        DEY
        BPL LOESCH
        LDA YPOS,X  Neue Position wird
        STA IREG2   und neues Shape
        LDY #15     kopieren
COPY    LDA (IREG1),Y
        STA (IREG2),Y
        DEY
        BPL COPY
        RTS
```