

# PETER'S ASSEMBLERECKE

für ATARI -Computer

## Bewegte Grafik

Nachdem wir uns in der letzten Ausgabe mit dem Atari-Betriebssystem befaßt haben, wollen wir uns diesmal etwas Handfesteres vornehmen. Genauer gesagt beschäftigen wir uns mit dem, was der Atari am besten kann: Mit bewegter Grafik. Richtig geraten, es geht um Player-Missile Grafik, kurz PMG genannt. Die PMG ist eine recht leistungsfähige Angelegenheit, aber leider auch ziemlich schwer zu handhaben. Besonders wenn man sie in Basic einsetzen möchte, gibt es einige Klippen zu umschiffen: Der Arbeitsspeicher der PMG umfaßt bei einzeiliger Auflösung, und nur diese wollen wir hier betrachten, etwas mehr als ein KByte. Vor dem Einschalten der PMG ist es notwendig, diesen Speicherbereich zu löschen, wobei sich deutlich zeigt, wie langsam eine BASIC FOR-NEXT Schleife sein kann. Weiterhin muß die vertikale Bewegung der Objekte durch Verschiebung der Bitmuster im PMG-Speicher bewerkstelligt werden, was bei Programmierung in BASIC nicht eben zu flüssigen Bewegungen führt. Durch Programmierung in Maschinensprache lassen sich diese beiden Hindernisse schnell beseitigen, und genau darin liegt der Grundgedanke des PM-Helfers.

Und so funktioniert's: Das Einrichten der PMG und Löschen des Speicherbereiches werden vom ersten Teil des PM-Helfers in Sekundenbruchteilen erledigt. Aufgerufen wird dieser Programmteil vom BASIC aus mit A=USR (1560, PM-Adresse), wobei die PM-Adresse die Basisadresse der PMG angibt. Wie man dies normalerweise macht, können Sie dem Demo (Listing 2) ab Zeile 1030 entnehmen.

Gleichzeitig wird in diesem Maschinenprogramm eine Interruptroutine in den Vertical Blank Interrupt (VBI) eingefügt, die den Umgang mit den Players entscheidend erleichtert. Sie brauchen ab jetzt nur noch die Adressen der darzustellenden Shapes und deren X- und Y-Koordinaten angeben, das VBI-Programm erledigt den Rest für Sie. Diese Methode der PM-Programmierung bringt uns noch einen zusätzlichen Vorteil: Da alle Grafikänderungen im VBI stattfinden, sind die Bewegungen absolut störungsfrei und fließend.

Im PM-Helfer Demoprogramm (Listing 2) sehen Sie, wie's gemacht wird: man definiert die Variablen HPOS = 1536 und VPOS = 1540 und kann dann mit deren Hilfe die Koordinaten poken. POKE VPOS+1,100 würde z.B. die vertikale Position von Player 2 (daher +1) auf 100 setzen. Vorher müssen Sie noch die Länge der Shapes in die Speicherzelle LAENGE = 1552 schreiben und die Adressen der Shapes eintragen. Letzteres wurde, um größere Poke-Orgien zu verhindern, mittels eines USR-Programmes erheblich vereinfacht. Mit A = USR (1566, Shapeadresse 1,...) können bis zu vier Adressen gleichzeitig übergeben werden, immer beginnend bei der Shape-Adresse von Player eins. Im Demoprogramm finden Sie den entsprechenden Befehl in Zeile 2040, hierbei wird allerdings die gleiche Adresse viermal übergeben, da auch viermal das gleiche Shape verwendet werden soll. Ausgeschaltet wird der ganze Zauber mit A=USR (1563).

Jetzt ist es an der Zeit, sich die inneren Vorgänge im Maschinenprogramm anzusehen.

Im ersten Teil von Listing 1 werden der Koppelspeicherbereich mit Basic sowie einige interne Variablen definiert. Es schließt sich eine kleine Sprungtabelle an, die die Einsprungsadressen ins Programm vereinheitlicht. Die Einschaltoutine PMHEIN beginnt mit dem Löschen der Variablen und des PMG-Speichers, fügt die Routine PMHVBI dem VBI ein und teilt ANTIC und GTIA mit, daß ab jetzt Players darzustellen sind. Genau umgekehrt arbeitet die Ausschaltoutine PMHAUS, welche ANTIC und GTIA in den Normalmodus zurücksetzt und die Interruptroutine beendet.

Das VBI-Programm PMHVBI besteht hauptsächlich aus einer Schleife, in der alle vier Players beginnend bei Player Nummer vier bearbeitet werden. Zuerst wird dabei immer die Horizontalposition ins zugehörige Hardware-Register geschrieben und anschließend anhand der alten und neuen Vertikalposition entschieden, ob das Shape an der alten Stelle

im PMG-Speicher gelöscht und an eine neue kopiert werden muß. Es folgt das Shape-Set Utility, in dem schlicht und ergreifend die Shapeadressen vom Stack genommen und in die dafür vorgesehenen Speicherzellen geschrieben werden. Vorsicht: Bitte beim Aufruf der SHPSET-Routine nie mehr als vier Adressen übergeben, da dies nicht überprüft wird.

Am leichtesten läßt sich das Programm als BASIC-Loader (s. Demo-Listing 2) in ein eigenes Programm integrieren. Sie brauchen dazu nur die Zeilen ab 30000 zu übernehmen. Wir von Computer-Kontakt würden uns natürlich freuen, wenn ein selbstgeschriebenes Programm bei uns eingeht, das den PM-Helfer benützt.

Das war's für diesen Monat und nicht vergessen, wenn jemand etwas gerne im Rahmen der Assembler-Ecke besprochen haben möchte, dann soll er sich über den Verlag an mich wenden.

Peter Finzel

## Assembler-Listing

```

0000          90          .OPT LIST
0100          ;*****
0110          ;PM-Helfer: Players im VBI bewegen
0120          ;
0130          ;Peter Finzel '85
0140          ;*****
0150          ;
0160          ;Koppelbereich mit BASIC
0170          ;
0180          HPOS = 1536      Vier Bytes f. Horiz. Positon
0190          VPOS = 1540      Vertikale Positionen
0200          SHPADR = 1544    ...die Adressen der Shapemuster
0210          LAENGE = 1552    Die Laenge der Shapes
0220          ;
0230          ;jetzt noch einige interne Adressen
0240          ;
0250          PMBADR = $0611   Hier wird PMBASE aufbewahrt
0260          VPALT = $0612     der jeweils alte Wert von VPOS
0270          PMJM = $0616     Aktuelle Playernummer
0280          PADR = $CC       Zeropage-Register Players
0290          SADR = $CE       Zeropage-Register Shapes
0300          ;
0310          ;Hardware und Betriebssystemadressen
0320          ;
0330          PMBASE = $D407
0340          DMACTL = $D400
0350          GRACL = $D01D
0360          GRAFPO = $D00D
0370          HFOSPO = $D000
0380          ;
0390          DMCTL = $022F
0400          SETBV = $E45C     OS-Routine fuer Interruptvektoren
0410          XITBV = $E462     Interrupt Abschluss
0420          ;
0430          ;*****
0440          ;jetzt geht's richtig los ...
0450          ;*****
0460          ;
0470          == $0618
0480          JMP PMHEIN        PM-Helfer einschalten
0490          JMP PMHAUS        und ausschalten
0500          JMP SHPSET        Uebergabe der Shapeadressen
0510          ;
0520          ;
0530          ;
0540          PMHEIN PLA        Anzahl der Args von BASIC-USR()
0550          PLA              MSB der Playerpage (immer 0)
0560          PLA              PageNr. an der PM-Bereich beginnt
0570          STA PMBADR        einstweilen aufbewahren
0580          CLD              es wird binär gerechnet!
0590          LDA #0            Variablen löschen...
0600          LDX #3            Pro Player einen Variablensatz
0610          VARCLR STA HPDS,X Hor. Position
0620          STA VPOS,X
0630          STA VPALT,X
0640          STA HFOSPO,X alle Players vom Schirm weg.
0650          DEX              das alles 4-mal bitte
0660          BPL VARCLR        und nochmal -->
0670          ;

```



```

063B 1B      0680 CLC      gleich wird addiert...
063C AD1106  0690 LDA PMBADR Player Missile Basis-Page
063F 6903    0700 ADC #3   Drei leere Pages bei einz. PM
0641 B5CD    0710 STA PADR+1 Zeropagezeiger aufbauen
0643 A900    0720 LDA #0   Loeschen des PM-Bereiches vorber.
0645 B5CC    0730 STA PADR  LSB Zeiger:=0
0647 A205    0740 LDX #5   15 Pages loeschen (incl. Miss.)
0649 A000    0750 PLCLR1 LDY #0   Index fuer eine Page loeschen
064B 91CC    0760 PLCLR2 STA (PADR),Y und los geht's
064D CB      0770 INY     der naechste bitte!
064E D0FB    0780 BNE PLCLR2 noch nicht fertig ->
0650 E6CD    0790 ;      INC PADR+1 naechste Page
0652 CA      0800 DEX     schon alle 5??
0653 D0F4    0810 BNE PLCLR1 nein, weiter -->
0655 A206    0820 ;      schliesslich das VBI-Programm anstarten
0657 A0BC    0830 ;
0659 A907    0840 LDX #PMH/VBI/256 MSB in X
065B 205CE4  0850 LDY #PMH/VBI/255 LSB in Y
065E AD1106  0860 LDA #PMH/VBI/255 LSB in Y
0661 B007D4  0870 LDY #PMH/VBI/255 LSB in Y
0664 A93E    0880 JSR SETVBV Das OS erledigt alles...
0666 B02F02  0890 ;
0669 A903    0900 ; und (endlich!) die PM-Graphik einschalten
066B B01DD0  0910 ;
066E 60      0920 ;
0670 A900    0930 LDA PMBADR hatten wir uns gemerkt
0672 B01DD0  0940 STA PMBASE jetzt ist ANTIC im Bild...
0674 A204    0950 LDA #33E und...
0676 9D0DD0  0960 STA SDNCTL auch scharf gemacht!
0678 7A CA    0970 LDA #3   auch der GTIA will's wissen
067A 7A CA    0980 STA GRACL  jetzt ist PMG eingeschaltet.
067C 7A CA    0990 RTS     Tscheuss!
067E 7A CA    1000 ;
0680 7A CA    1010 ;Ausschaltroutine
0682 7A CA    1020 ;
0684 7A CA    1030 PMHAUS PLA  USR()-Rest vom Stack nehmen
0686 7A CA    1040 LDA #0   GTIA ausschalten
0688 7A CA    1050 STA GRACL
068A 7A CA    1060 LDX #4
068C 7A CA    1070 GTCLR STA GRAFFO,X interne GTIA-Reg loeschen
068E 7A CA    1080 DEX     alle 5 geloescht?
0690 7A CA    1090 BPL GTCLR nein -->
0692 7A CA    1100 ;
0694 7A CA    1110 LDA #222 und ANTIC normal schalten
0696 7A CA    1120 STA SDNCTL
0698 7A CA    1130 LDX #XITVBV/256 VBI abschalten
069A 7A CA    1140 LDY #XITVBV/255
069C 7A CA    1150 LDA #7
069E 7A CA    1160 JSR SETVBV ...fertig!
069F 7A CA    1170 RTS     zurueck zu BASIC
06A1 7A CA    1180 ;
06A3 7A CA    1190 ;
06A5 7A CA    1200 ; Die PM-Helfer VBI-Routine
06A7 7A CA    1210 ;
06A9 7A CA    1220 ;
06AB 7A CA    1230 PMH/VBI CLD  Wir rechnen binar!!
06AD 7A CA    1240 LDA PMBADR PMG-Basis Pagenummer
06AF 7A CA    1250 CLC
06B1 7A CA    1260 ADC #7   letzter Player 7 Pages weiter
06B3 7A CA    1270 STA PADR+1 in Zeropageregister
06B5 7A CA    1280 LDA #3   wir beginnen bei Player Nr. 4!
06B7 7A CA    1290 STA PNUM
06B9 7A CA    1300 ;
06BB 7A CA    1310 ;
06BD 7A CA    1320 ; In d. Schleife werden alle 4 Players bewegt
06BF 7A CA    1330 ;

069A AD1606  1340 ALLE4 LDA PNUM  nochmal laden
069D 0A      1350 ASL A     mal 2
069E AA      1360 TAX     als Adressindex
069F B00806  1370 LDA SHPADR,X Shape Adresse LSB
06A2 B5CE    1380 STA SADR  in Zeropagereg.
06A4 B00906  1390 LDA SHPADR+1,X Shape Adresse MSB
06A7 B5CF    1400 STA SADR+1 in Zeropagereg.
06A9 AE1606  1410 LDX PNUM  aktuelle Playernummer
06AC B00006  1420 LDA HPOS,X Horizontale Position
06AF 9D00D0  1430 STA HPSP0,X in Hardwareregister
06B2 B01206  1440 LDA VPALT,X alte Figur loeschen
06B5 D00406  1450 CMP VPOS,X Figur Vertikal bewegt?
06B8 F025    1460 BEQ NAECHSTER nein, fertig! -->
06BA B5CC    1470 ;
06BC A000    1480 STA PADR  VPALT als LSB in PM-Zeiger eintr.
06BE A900    1490 LDY #0   Zeilenzaehler auf 0
06C0 91CC    1500 LDA #0   Null wie loeschen
06C2 CB      1510 PLCLR STA (PADR),Y
06C4 CB      1520 INY
06C6 D0FB    1530 CPY LAENGE noch weiter?
06C8 D0FB    1540 BNE PLCLR ja -->
06CA B5CC    1550 ;
06CB B00406  1560 LDA VPOS,X neue Vertikalposition
06CD B5CC    1570 STA PADR  als LSB in PM-Zeiger
06CE A000    1580 LDY #0   Kopierschleife vorbereiten
06CF B1CE    1590 PLRSHP LDA (SADR),Y Shapemuster laden
06D1 91CC    1600 STA (PADR),Y in PM-Bereich eintragen
06D3 CB      1610 INY     Byte um Byte ...
06D5 CB      1620 CPY LAENGE schon alle?
06D7 D0F6    1630 BNE PLRSHP noch nicht -->
06D9 B00406  1640 ;
06DB B00406  1650 LDA VPOS,X neue Vertikalposition
06DD 9D1206  1660 STA VPALT,X als 'alte' fuer naechsten Lauf
06DF C6CD    1665 NAECHSTER DEC PADR+1 naechster Player
06E1 CE1606  1670 DEC PNUM  schon alle 4 Players?
06E4 10B4    1680 BPL ALLE4 nein -->
06E6 4C62E4  1690 ;
06E8 4C62E4  1700 JMP XITVBV VBI-Ende ==>
06EA 4C62E4  1710 ;
06EC 4C62E4  1720 ;
06EE 4C62E4  1730 ;
06F0 4C62E4  1740 ;
06F2 4C62E4  1750 ;
06F4 4C62E4  1760 ;
06F6 4C62E4  1770 ;
06F8 4C62E4  1780 ;
06FA 4C62E4  1790 ;
06FC 4C62E4  1800 ;
06FE 4C62E4  1810 ;
0700 4C62E4  1820 ;
0702 4C62E4  1830 ;
0704 4C62E4  1840 ;
0706 4C62E4  1850 ;
0708 4C62E4  1860 ;
070A 4C62E4  1870 ;
070C 4C62E4  1880 ;
070E 4C62E4  1890 ;
0710 4C62E4  1900 ;
0712 4C62E4  1910 ;
0714 4C62E4  1920 ;
0716 4C62E4  1930 ;
0718 4C62E4  1940 ;

06E9 A000    1790 SHPSET LDY #0   Zeiger einrichten
06EB 68      1800 PLA     Anzahl der Args von USR()
06EC AA      1810 TAX     im X-Reg. aufbewahren
06ED F00E    1820 ;
06EF 68      1830 NXTARG BEQ NIXMEHR fertig-->
06F0 990906  1840 PLA     zuerst MSB
06F3 68      1850 STA SHPADR+1,Y
06F4 990806  1860 PLA     jetzt LSB von USR()
06F7 CB      1870 STA SHPADR,Y
06F8 CB      1880 INY     Zeiger zwei weiter
06F9 CA      1890 INY
06FA 4CED06  1900 DEX     Argumentzaehler berichtigen
06FD 60      1910 JMP NXTARG und weiter==>
06FE 60      1920 ;
06FF 60      1930 NIXMEHR RTS  fertig!
0700 60      1940 ;

```

ASSEMBLY ERRORS: 0 22110 BYTES FREE

## Basic-Listing

```

1000 REM *** Listing 2: PM-Helfer Demo ***
1010 HPOS=1536:VPOS=1540:LAENGE=1552:PMCOL=704:REM * Koppelvariable
1020 PMHEIN=1560:PMHAUS=1563:SHPSET=1566:REM * PMH-Einsprungadressen
1030 RAMTOP=PEEK(106)-16:REM Speicherplatz reservieren
1040 POKE 106,RAMTOP:GRAPHICS 3+16
1050 GOSUB 30000:REM * Maschinencode 'poken'...
1060 POKE LAENGE,16:REM * die Shapes sind 16 Bytes lang
1070 A=USR(PMHEIN,RAMTOP):REM PM einschalten
1080 SHP1=(RAMTOP+1)*256:REM * Hier ist Platz fuer das Shape
2000 REM * ab hier folgt eine einfache Bewegingsroutine
2020 RESTORE 9100:FOR I=0 TO 15:READ D:POKE SHP1+I,D:NEXT I
2030 POKE PMCOL,12:POKE PMCOL+1,186:POKE PMCOL+2,134:POKE PMCOL+3,46
2040 A=USR(SHPSET,SHP1,SHP1,SHP1,SHP1):REM * Shapeadresse an PMH
2050 POKE VPOS,120:POKE HPOS+1,120:REM und Bewegung...
2060 FOR I=16 TO 232:H=255-I
2070 POKE HPOS,I:POKE VPOS+1,I:REM PLAYER 1 & 2
2080 POKE HPOS+2,I:POKE VPOS+2,I:REM PLAYER 3
2090 POKE HPOS+3,H:POKE VPOS+3,H:REM PLAYER 4
2100 NEXT I:GOTO 2060
9000 REM * Shape-Beispiel
9100 DATA 0,24,60,60,126,90,255,231,255,255,90,102,60,60,24,0
30000 REM * Binaer-File laden
30010 S=0:RESTORE 30100
30020 FOR A=1560 TO 1789:READ D:POKE A,D:S=S+D:NEXT A
30030 IF S<>24772 THEN ? "DATEN-FEHLER!":STOP

```



## 30090 RETURN

```

30100 DATA 76,33,6,76,111,6,76,233,6,104,104,104,141,17,6,216,169
30110 DATA 0,162,3,157,0,6,157,4,6,157,18,6,157,0,208,202,16,241,24
30120 DATA 173,17,6,105,3,133,205,169,0,133,204,162,5,160,0,145,204
30130 DATA 200,208,251,230,205,202,208,244,162,6,160,140,169,7,32
30140 DATA 92,228,173,17,6,141,7,212,169,62,141,47,2,169,3,141,29
30150 DATA 208,96,104,169,0,141,29,208,162,4,157,13,208,202,16,250
30160 DATA 169,34,141,47,2,162,228,160,98,169,7,32,92,228,96,216,173
30170 DATA 17,6,24,105,7,133,205,169,3,141,22,6,173,22,6,10,170,189
30180 DATA 8,6,133,206,189,9,6,133,207,174,22,6,189,0,6,157,0,208
30190 DATA 189,18,6,221,4,6,240,37,133,204,160,0,169,0,145,204,200
30200 DATA 204,16,6,208,248,189,4,6,133,204,160,0,177,206,145,204
30210 DATA 200,204,16,6,208,246,189,4,6,157,18,6,198,205,206,22,6
30220 DATA 16,180,76,98,228,160,0,104,170,240,14,104,153,9,6,104,153
30230 DATA 8,6,200,200,202,76,237,6,96

```

## Atari-Leserfragen

Hier wieder einige, häufig gestellte Fragen, die von allgemeinem Interesse sind:

### Gibt es für die ATARI-Computer "Turbotape"-Programme?

Theoretisch ist es auch bei den ATARI-Computern möglich, über bestimmte Veränderungen der Kassettenroutinen eine schnellere Ein- und Ausgabe zu bewirken. Dies wird auf den ATARI-Rechnern jedoch praktisch nicht angewandt, da die Datensicherheit zu sehr darunter leiden würde. Der Grund hierfür ist einfach: Der ATARI-Rekorder verwendet nur eine Halbspur für die Programmdateien – die andere Halbspur kann (auch während des Ladevorganges) Musik und/oder Sprache wiedergeben. Dies kann man übrigens sehr

gut beobachten, wenn man eine Musikkassette einlegt, PLAY drückt und POKE 54018,52 eingibt – 60 schaltet wieder aus.

### Kann man auf den ATARI-Computern auch Kreise und Ovale zeichnen oder wird hierfür eine BASIC-Erweiterung benötigt?

Natürlich ist es auf den ATARI-Geräten auch ohne Erweiterung möglich, Kreise und Ovale zu zeichnen. Hierfür müssen allerdings einige Punkte auf dem Kreisbogen berechnet und mittels DRAWTO verbunden werden – schon ist der Kreis fertig. Für die Berechnungen der Punkte schaltet man seinen Computer mit dem DEG-Befehl in den DEGREE-Modus. Eine Schleife von 0 bis

360 (STEP 15 liefert genügend Punkte) stellt die Gradangaben, aus denen mit Hilfe von SINus und COSinus die zugehörigen Koordinaten berechnet werden. Da die SIN- und COS-Werte jedoch immer zwischen -1 und 1 liegen, werden sie mit dem Radius multipliziert. Für OVALE kommen noch zusätzlich die Faktoren für die Streck-

kung in X- und Y-Richtung hinzu (Normalwert: 1). Die Parameter, die an das abgedruckte Unterprogramm übergeben werden müssen, sind die Koordinaten des Mittelpunktes (X und Y), der Radius (R) und die Streckungsfaktoren (XF und YF).

Thomas Tausend

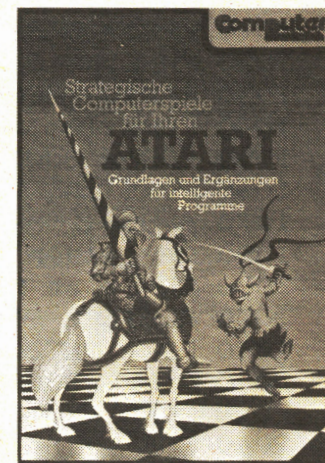
## Strategische Computerspiele für Ihren ATARI

von John White  
150 Seiten, DM 32,-  
Verlag Markt & Technik  
ISBN 3-89090-004-6

Wie viele andere ist auch dieses Buch eine Übersetzung eines englischsprachigen Titels ("Writing strategy games on your ATARI computer").

Für all diejenigen, denen die Space Invaders in den Ohren dröhnen, die PacMans Schmatzen nervt oder die Mitleid mit Donkey Kong bekommen haben, ist dieses Buch eine willkommene Abwechslung.

Auf ca. 150 Seiten findet der Leser ausführlich allerlei Wissenswertes über strategische Computerspiele. Alle verwendeten Verfahren und Techniken werden ausführlich erklärt, so daß sehr leicht eigene Spiele mit intelligenten Komponenten ausgestattet werden können. Als Beispiele finden Teilpro-



bleme bei Dame und Schach Verwendung. Aber auch einige komplette Strategiespiele sind tippfertig enthalten.

Wer aus diesem Buch einen Nutzen ziehen möchte, sollte bereits über grundlegende BASIC-Kenntnisse verfügen.

Thomas Tausend

```

10 GRAPHICS 8
20 COLOR 1
30 FOR W=0 TO 360 STEP 15
40 X=160+SIN(W)*40
50 Y=95+COS(W)*40
60 R=40:XF=1:YF=1
70 GOSUB 9000
80 NEXT W
90 END
9000 REM Unterprogramm fuer Kreise
9010 DEG :PLOT X,Y+R*YF
9020 FOR N=0 TO 360 STEP 15
9030 DRAWTO X+SIN(N)*R*XF,Y+COS(N)*R*YF
9040 NEXT N
9050 RETURN
9990 REM *****
9991 REM * Unterprogramm f. Kreise *
9992 REM * (c) by Thomas Tausend *
9993 REM *****

```