

## **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 einzelzeiliger 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 Playern 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, Shapeadresse1,...}` 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)`.

## **Inneren Vorgänge im Maschinenprogramm**

---

Im ersten Teil von Listing 1 werden der Koppelspeicherbereich mit Basic sowie einigen internen Variablen definiert. Es schließt sich eine kleine Sprungtabelle an, die die Einsprungsadressen ins Programm vereinheitlicht. Die Einschalt routine `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 Ausschalt routine `PMHAUS`, welche `ANTIC` und `GTIA` in den Normalmodus zurücksetzt und die Interrupt routine 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 position 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-Leader (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 benutzt. 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

 [Assemblerecke\\_02.pdf](#) (Original-Artikel)

 [Assemblerecke\\_02.atr](#) (Disketten-Image mit getesteten Listings)

 [PMHELPER-BAS.txt](#) (Listing als ASC- PC-Textfile)

 [PMHDEMO.txt](#) (Demo für den Einsatz in reinen Assemblerprogrammen\*)

\* Das Demo zeigt wie die PM-Helper Routine auch in reinen Assemblerprogrammen eingesetzt werden kann. Das Assembler-Listing stellt dabei die BASIC-Routine komplett in Maschinensprache nach. (Das Demo wurde von mir ergänzt und steht so nicht im Originalartikel)

# 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 ;
=0600 0180 HPOS  = 1536    Vier Bytes f. Horiz. Positon
=0604 0190 VPOS  = 1540    Vertikale Positionen
=0608 0200 SHPADR = 1544    ...die Adressen der Shapemuster
=0610 0210 LAENGE = 1552    Die Laenge der Shapes
0220 ;
0230 ;jetzt noch einige interne Adressen
0240 ;
=0611 0250 PMBADR = $0611    Hier wird PMBASE aufbewahrt
=0612 0260 VPALT  = $0612    der jeweils alte Wert von VPOS
=0616 0270 PNUM   = $0616    Aktuelle Playernummer
=00CC 0280 PADR   = $CC      Zeropage-Register Players
=00CE 0290 SADR   = $CE      Zeropage-Register Shapes
0300 ;
0310 ;Hardware und Betriebssystemadressen
0320 ;
=D407 0330 PMBASE = $D407
=D400 0340 DMACTL = $D400
=D01D 0350 GRACLT = $D01D
=D00D 0360 GRAFPO = $D00D
=D000 0370 HPOSPO = $D000
0380 ;
=022F 0390 SDMCTL = $022F
=E45C 0400 SETVBV = $E45C    OS-Routine fuer Interruptvektoren
=E462 0410 XITVBV = $E462    Interrupt Abschluss
0420 ;
0430 ;*****
0440 ;jetzt geht's richtig los ...
0450 ;*****
0460 ;
0000 0470          *= $0618
0618 4C2106 0480          JMP PMHEIN    PM-Helfer einschalten
061B 4C6F06 0490          JMP PMHAUS    und ausschalten
061E 4CE906 0500          JMP SHPSET    Uebergabe der Shapeadressen
0510 ;
0520 ;
0530 ;
0621 68      0540 PMHEIN PLA          Anzahl der Args von BASIC-USR()
0622 68      0550          PLA          MSB der Playerpage (immer 0)
0623 68      0560          PLA          Pagenr. an der PM-Bereich beginnt
0624 8D1106 0570          STA PMBADR    einstweilen aufbewahren
0627 D8      0580          CLD          es wird binär gerechnet!
0628 A900    0590          LDA #0      Variablen löschen...
062A A203    0600          LDX #3      Pro Player einen Variablensatz
062C 9D0006 0610 VARCLR STA HPOS,X    Hor. Position
062F 9D0406 0620          STA VPOS,X
0632 9D1206 0630          STA VPALT,X
0635 9D00D0 0640          STA HPOSPO,X alle Players vom Schirm weg.
0638 CA      0650          DEX          das alles 4-mal bitte
0639 10F1    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 vorher.
0645 B5CC        0730      STA PADR   LSB Zeiger:=0
0647 A205        0740      LDX #5      ;5 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 ->
0790 ;
0650 E6CD        0800      INC PADR+1  naechste Page
0652 CA          0810      DEX          schon alle 5??
0653 D0F4        0820      BNE PLCLR1  nein, weiter -->
0830 ;
0840 ; schliesslich das VBI-Programm anstarten
0850 ;
0655 A206        0860      LDX #PMHVBI/256 MSB in X
0657 A0BC        0870      LDY #PMHVBI&255 LSB in Y
0659 A907        0880      LDA #7      Deferred VBI ist gemeint
065B 205CE4      0890      JSR SETVBV  Das OS erledigt alles...
0900 ;
0910 ; und (endlich!) die PM-Graphik einschalten
0920 ;
065E AD1106      0930      LDA PMBADR  hatten wir uns gemerkt
0661 BD07D4      0940      STA PMBASE  jetzt ist ANTIC im Bild...
0664 A93E        0950      LDA #$3E    und...
0666 BD2F02      0960      STA SDMCTL  auch scharf gemacht!
0669 A903        0970      LDA #3      auch der GTIA will's wissen
066B BD1DD0      0980      STA GRACL   jetzt ist PMG eingeschaltet.
066E 60          0990      RTS        Tschuess!
1000 ;
1010 ;Ausschaltroutine
1020 ;
066F 68          1030 PMHAUS PLA      USR()-Rest vom Stack nehmen
0670 A900        1040      LDA #0      GTIA ausschalten
0672 BD1DD0      1050      STA GRACL
0675 A204        1060      LDX #4
0677 9D0DD0      1070 GTCLR  STA GRAFPO,X interne GTIA-Reg loeschen
067A CA          1080      DEX          alle 5 geloescht?
067B 10FA        1090      BPL GTCLR  nein -->
1100 ;
067D A922        1110      LDA #$22    und ANTIC normal schalten
067F BD2F02      1120      STA SDMCTL
0682 A2E4        1130      LDX #XITVBV/256 VBI abschalten
0684 A062        1140      LDY #XITVBV&255
0686 A907        1150      LDA #7
0688 205CE4      1160      JSR SETVBV  ...fertig!
068B 60          1170      RTS        zurueck zu BASIC
1180 ;
1190 ;*****
1200 ; Die PM-Helfer VBI-Routine
1210 ;*****
1220 ;
068C DB          1230 PMHVBI CLD      Wir rechnen binaer!!
068D AD1106      1240      LDA PMBADR  PMG-Basis Pagenummer
0690 1B          1250      CLC
0691 6907        1260      ADC #7      letzter Player 7 Pages weiter
0693 B5CD        1270      STA PADR+1  in Zeropageregister
0695 A903        1280      LDA #3      wir beginnen bei Player Nr. 4!
0697 BD1606      1290      STA PNUM
1310 ;
1320 ;In d. Schleife werden alle 4 Players bewegt
1330 ;

```

```

069A AD1606      1340 ALLE4   LDA PNUM      nochmal laden
069D 0A          1350       ASL A          mal 2
069E AA          1360       TAX           als Adressindex
069F BD0806      1370       LDA SHPADR,X Shape Adresse LSB
06A2 85CE        1380       STA SADR      in Zeropagereg.
06A4 BD0906      1390       LDA SHPADR+1,X Shape Adresse MSB
06A7 85CF        1400       STA SADR+1    in Zeropagereg.
06A9 AE1606      1410       LDX PNUM      aktuelle Playernummer
06AC BD0006      1420       LDA HPOS,X    Horizontale Position
06AF 9D00D0      1430       STA HPOS0,X in Hardwareregister
06B2 BD1206      1440       LDA VPALT,X   alte Figur loeschen
06B5 DD0406      1450       CMP VPOS,X   Figur Vertikal bewegt?
06B8 F025        1460       BEQ NAECHSTER nein, fertig! -->
                   1470 ;
06BA 85CC        1480       STA PADR      VPALT als LSB in PM-Zeiger eintr.
06BC A000        1490       LDY #0       Zeilenzaehler auf 0
06BE A900        1500       LDA #0       Null wie loeschen
06C0 91CC        1510 PLRCLR STA (PADR),Y
06C2 C8          1520       INY
06C3 CC1006      1530       CPY LAENGE    noch weiter?
06C6 D0F8        1540       BNE PLRCLR   ja -->
                   1550 ;
06C8 BD0406      1560       LDA VPOS,X   neue Vertikalposition
06CB 85CC        1570       STA PADR      als LSB in PM-Zeiger
06CD 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 C8          1610       INY          Byte um Byte ...
06D4 CC1006      1620       CPY LAENGE    schon alle?
06D7 D0F6        1630       BNE PLRSHP   noch nicht -->
                   1640 ;
06D9 BD0406      1650       LDA VPOS,X   neue Vertikalposition
06DC 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 -->
                   1690 ;
06E6 4C62E4      1700       JMP XITVBV   VBI-Ende ==>
                   1710 ;
                   1720 ;*****
                   1730 ; Hier noch ein Utiliy, mit dem die Shape-
                   1740 ; adressen leicht eingetragen werden koennen
                   1750 ; AUFRUF: X=USR(1574,<Adr. von Shape 1>,...)
                   1760 ; bis max. 4-Adressen (nie mehr!)
                   1770 ;*****
                   1780 ;
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
                   1820 ;
06ED F00E        1830 NXTARG BEQ NIXMEHR fertig-->
06EF 68          1840       PLA          zuerst MSB
06F0 990906      1850       STA SHPADR+1,Y
06F3 68          1860       PLA          jetzt LSB von USR()
06F4 990806      1870       STA SHPADR,Y
06F7 C8          1880       INY          Zeiger zwei weiter
06F8 C8          1890       INY
06F9 CA          1900       DEX          Argumentzaehler berichtigen
06FA 4CED06      1910       JMP NXTARG   und weiter==>
                   1920 ;
06FD 60          1930 NIXMEHR RTS        fertig!
                   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 Bewegungsroutine
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

```

```

1000 REM *** LISTING 2: PM-HELPER DEMO ***
1010 HPOS=1536:VPOS=1540:LAENGE=1552:PMCOL=704
1020 PMHEIN=1560:PMHAUS=1536:SHPSET=1566
1030 RAMTOP=PEEK(106)-16
1040 POKE 106,RAMTOP:GRAPHICS 3+16
1050 GOSUB 30000
1060 POKE LAENGE,16
1070 A=USR(PMHEIN,RAMTOP)
1080 SHP1=(RAMTOP+1)*256
2000 REM * LADEN DES SHAPES UND BEWEGUNG
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)
2050 POKE VPOS,120:POKE HPOS+1,120
2060 FOR I=16 TO 232:H=255-I
2070 POKE HPOS,I:POKE VPOS+1,I
2080 POKE HPOS+2,I:POKE VPOS+2,I
2090 POKE HPOS+3,H:POKE VPOS+3,H
2100 NEXT I:GOTO 2060
9000 REM * SHAPE
9100 DATA 0,24,60,60,126,90,255,231,255,255,90,102,60,60,24,0
30000 REM * MASCHINEN-UNTERPROGRAMM 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
30105 DATA 104,104,141,17,6,216,169
30110 DATA 0,162,3,157,0,6,157,4,6,157,18,6
30115 DATA 157,0,208,202,16,241,24
30120 DATA 173,17,6,105,3,133,205,169,0,133
30125 DATA 204,162,5,160,0,145,204
30130 DATA 200,208,251,230,205,202,208,244
30135 DATA 162,6,160,140,169,7,32
30140 DATA 92,228,173,17,6,141,7,212,169,62,141
30145 DATA 47,2,169,3,141,29
30150 DATA 208,96,104,169,0,141,29,208,162,4
30155 DATA 157,13,208,202,16,250
30160 DATA 169,34,141,47,2,162,228,160,98,169
30165 DATA 7,32,92,228,96,216,173
30170 DATA 17,6,24,105,7,133,205,169,3,141,22,6
30175 DATA 173,22,6,10,170,189
30180 DATA 8,6,133,206,189,9,6,133,207,174,22,6
30185 DATA 189,0,6,157,0,208
30190 DATA 189,18,6,221,4,6,240,37,133,204,160
30195 DATA 0,169,0,145,204,200
30200 DATA 204,16,6,208,248,189,4,6,133,204,160
30205 DATA 0,177,206,145,204
30210 DATA 200,204,16,6,208,246,189,4,6,157,18
30215 DATA 6,198,205,206,22,6
30220 DATA 16,180,76,98,228,160,0,104,170,240
30225 DATA 14,104,153,9,6,104,153
30230 DATA 8,6,200,200,202,76,237,6,96

```

```
*****
* PM-HELPER DEMO
* ASSEMBLERECKE 02
* COMPUTER KONTAKT 03/85
* PETER FINZEL
*****
```

```
HPOS      EQU $0600
VPOS      EQU $0604
SHPADR     EQU $0608
LAENGE     EQU $0610
PMBADR     EQU $0611
VPALT      EQU $0612
PNUM       EQU $0616
PADR       EQU $80
SADR       EQU $82
PMBASE     EQU $D407
DMACTL     EQU $D400
GRACTL     EQU $D01D
GRAFP0     EQU $D00D
HPOSPO     EQU $D000
PCOLR0     EQU $02C0
SDMCTL     EQU $022F
SETVBV     EQU $E45C
XITVBV     EQU $E462
```

```
* HAUPTPROGRAMM-SCHLEIFE
      ORG $A800
```

```
      LDA #16          SHAPE LAENGE
      STA LAENGE
      LDX #$B0         MSB PLAYER-SPEICHER
      LDY #0          LSB PLAYER-SPEICHER
      STX PMBADR       PLAYER-SPEICHER IN ZWISCHENVARIABLE
      STY SHPADR       SHAPE-VORLAGE AM ANFANG DES PLAYER-SPEICHER
      STX SHPADR+1     FUER ALLE 4 PLAYER GLEICHES SHAPE
      STY SHPADR+2
      STX SHPADR+3
      STY SHPADR+4
      STX SHPADR+5
      STY SHPADR+6
      STX SHPADR+7
      LDA SHPADR       SHAPE IN SHAPE-ADRESSE KOPIEREN
      STA SADR
      LDA SHPADR+1
      STA SADR+1
      LDY #0
SHPLOA  LDA SHAPE1,Y
      STA (SADR),Y
      INY
      CPY #16
      BNE SHPLOA
      JSR PMHEIN       PM-GRAFIK UND VBI EINSCHALTEN
      LDA #12          PLAYER FARBEN SETZEN
      STA PCOLR0
      LDA #186
      STA PCOLR0+1
      LDA #134
      STA PCOLR0+2
      LDA #46
      STA PCOLR0+3
      LDA #120         PLAYER BEWEGUNG
      STA VPOS
      STA HPOS+1
SRTPOS  LDX #0
      LDY #255
NXTPOS  STX HPOS
```

---



```

        STX VPOS+1
        STX HPOS+2
        STX VPOS+2
        STX VPOS+2
        STY HPOS+3
        STY VPOS+3
        LDA Z0
WAIT1   CMP Z0
        BEQ WAIT1
        LDA $D01F
        AND %00000001    START GEDRUECK?
        BEQ ENDE         JA -> DANN ENDE
        DEY
        INX
        CPX #255
        BEQ SRTPOS
        JMP NXTPOS
ENDE    JSR PMHAUS        PM-GRAFIK UND VBI AUS
        RTS              RUECKSPRUNG

* PLAYER MUSTER
SHAPE1  DFB 0,24,60,60,126,90,255
        DFB 231,255,255,90,102
        DFB 60,60,24,0

* PM-HELPER EINSCHALTEN
PMHEIN  LDA #0           ALLE PLAYER VARIABLEN AUF 0
        LDX #3
VARCLR  STA VPOS,X
        STA HPOS,X
        STA VPALT,X
        STA HPOSP0,X
        DEX
        BPL VARCLR
        CLC              PM-SPEICHERBEREICH LOESCHEN
        LDA PMBADR
        ADC #3
        STA PADR+1
        LDA #0
        STA PADR
        LDX #5
PLCLR1  LDY #0
PLCLR2  STA (PADR),Y
        INY
        BNE PLCLR2
        INC PADR+1
        DEX
        BNE PLCLR1
        LDX #PMHUVBI:H   VBI-PROGRAMM STARTEN
        LDY #PMHUVBI:L
        LDA #7
        JSR SETVBV
        LDA PMBADR       PM-GRAFIK EINSCHALTEN
        STA PMBASE
        LDA #$3E
        STA S0MCTL
        LDA #3
        STA GRCTL
        RTS

* PM-HELPER AUSSCHALTEN
PMHAUS  LDA #0
        STA GRCTL
        LDX #4
GTCLR   STA GRAFP0,X
        DEX
        BPL GTCLR

```

```

        LDA #$22
        STA SDMCTL
        LDX #XITVBU:H
        LDY #XITVBU:L
        LDA #7
        JSR SETVBU
        RTS

* DIE PM-HELPER VBI-ROUTINE
PMHVBI  LDA PMBADR
        CLC
        ADC #7
        STA PADR+1
        LDA #3
        STA PNUM
ALLE4   LDA PNUM          FUER ALLE 4 PLAZER ANWENDEN
        ASL
        TAX
        LDA SHPADR,X
        STA SADR
        LDA SHPADR+1,X
        STA SADR+1
        LDX PNUM
        LDA HPOS,X
        STA HPOSPO,X
        LDA UPALT,X
        CMP UPOS,X       POSITION VERAENDERT?
        BEQ NAECHST      NEINE -> WEITER
        STA PADR          ALTEN PLAYER LOESCHEN
        LDY #0
        LDA #0
PLRCLR  STA (PADR),Y
        INY
        CPY LAENGE
        BNE PLRCLR
        LDA UPOS,X       NEUEN PLAYER MALEN
        STA PADR
        LDY #0
PLRSHP  LDA (SADR),Y
        STA (PADR),Y
        INY
        CPY LAENGE
        BNE PLRSHP
        LDA UPOS,X
        STA UPALT,X
NAECHST DEC PADR+1
        DEC PNUM
        BPL ALLE4
        JMP XITVBU       SPRUNG ZUM OS-VBI

```