Grafik für Profis

Allen Grafikbegeisterten soll dieser Kurs Tips und Programmierkniffe zum Thema hochauflösende Grafik vermitteln. Wir zeigen Ihnen leistungsstarke Grafik-Routinen mit sehr schnellen Befehlen. Außerdem bekommen Sie ein Programm für 3-D-Grafik.

Wohl jeder Assembler-Alchimist, der gerade das kleine ABC der Maschinensprache kennengelernt hat, wird bei dem Versuch, ein einfaches Basic-Programm in Assembler zu übersetzen, auf nahezu unüberwindliche Schwierigkeiten gestoßen sein. Erfordert doch schon die Multiplikation zweier 16-Bit-Zahlen ein kompliziertes Unterprogramm, so wird der Aufwand bei dem Versuch, den Sinus oder den Logarithmus aus einer Fließkommazahl zu berechnen, geradezu gigantisch. Vorausgesetzt, man will alle erforderlichen Unterprogramme selber schreiben. Glücklicherweise sind die benötigten Routinen jedoch schon im Basic-ROM des C 64 vorhanden. Woran liegt es aber, daß man auf solche Schwierigkeiten stößt, wenn man einen Algorithmus, der sich in Basic relativ einfach bewältigen läßt, in Assembler formulieren will? Nun, die Antwort liegt darin begründet, daß Basic eine sogenannte »höhere« Programmiersprache ist. Die Befehle, die Sie im Wortschatz der Sprache Basic finden, werden Sie im Wortschatz des 6510-Prozessors vergeblich suchen. Das liegt daran, daß jeder einzelne Basic-Befehl sich aus vielen Maschinenbefehlen zusammensetzt. Jedes dieser Maschinenprogramme simuliert praktisch einen Basic-Befehl. Sie werden fragen, was dies alles mit diesem Kurs zu tun hat. Wie Sie der Überschrift entnehmen können, geht es um die Programmierung von hochauflösender Grafik. Auch hier geht es darum, mit Hilfe des 6510-Wortschatzes kompliziertere Befehle aufzubauen, die dazu dienen, Punkte zu zeichnen, Linien zu ziehen etc.

Die meisten dieser Befehle enthalten bestimmte Rechenalgorithmen, die dazu dienen, die Koordinaten eines zu zeichnenden Punktes zu bestimmen. Eine gute Voraussetzung für diesen Grafikkurs sind die beiden Kurse »Reise durch die Wunderwelt der Grafik« und »Assembler ist keine Alchimie« von H. Ponnath. Wenn Sie diese beiden Kurse aufmerksam verfolgt haben, dann sind Sie mit den Grafikfähigkeiten des C 64 vertraut. Mit einigen der dort erworbenen Assemblerkenntnissen dürften Sie wohl in der Lage sein, die zum Zeichnen in der Hi-Res-Grafik wichtigen Befehle selbst in Maschinensprache zu formulieren.

Ein Ergebnis eines solchen Versuchs könnte etwa das Programm »HiRes-3« von H. Ponnath sein. Wenn Sie sich aber eine zeitlang intensiv mit diesem Programm beschäftigt haben, werden Sie merken, daß die Zeichengeschwindigkeit der meisten Befehle noch Wünsche offen läßt. Der Grund hierfür liegt nicht etwa darin, daß das Programm schlecht programmiert wurde, sondern das Problem liegt in der Berechnung der Algorithmen. Das betrifft besonders die Befehle, die mit der herkömmlichen 16-Bit-Arithmetik scheinbar nicht mehr zu bewältigen sind (Circle-Befehl). Gerade hierin aber zeigt sich die wahre Programmierkunst. Nämlich die Fähigkeit, mit einigen Programmierkniffen und mit etwas Fantasie das scheinbar Unmögliche doch noch möglich zu machen. Ich möchte versuchen, Ihnen in diesem Kurs einige dieser »Tricks« zu vermitteln. Dazu eignet sich meiner Ansicht nach nichts besser als das reizvolle Thema »Hochauflösende Grafik«. Dazu bekommen Sie nebenbei auch noch ein professionelles Grafikprogramm, mit dem es sich hervorragend arbeiten läßt. Ich möchte Ihnen nun dieses Programm, das der Hauptgegenstand dieses Kurses sein wird, etwas genauer vorstellen. »Profi-Grafik 64«, so der Name des Programms, besteht aus vielerlei Grafikroutinen, die der besseren Handhabung wegen zu einer Basic-Erweiterung zusammengefaßt wurden. Profi-Grafik 64 hat einige hervorstechende Merkmale:

— Es stehen zwei Grafikseiten zur Verfügung
— Die Befehle fallen durch ihre Leistungsstärke, Schnelligkeit und leichte Handhabung auf
— Multicolor-Grafik wurde ohne Einschränkungen verwirklicht
— Es können gleichzeitig acht Sprites interruptgesteuert über den Bildschirm bewegt werden
— Durch einfache Befehle wird 3-D-Grafik möglich

Nachdem ich Ihnen hoffentlich ein wenig den Mund wässrig gemacht habe, wollen wir nun mit der Besprechung des Programms beginnen.

In dieser Folge finden Sie ein ziemlich langes Assemblerprogramm (Listing 1) sowie ein MSE-Listing (Listing 2). Dieses Listing bildet den Grundstock für eine Basic-Erweiterung und hat eigentlich nichts mit den Grafikroutinen zu tun. Deshalb tippen Sie dieses Listing am besten erst mal ab und speichern es.

Schauen Sie sich nun einmal das Assemblerlisting an. Sie finden dort die Routinen der ersten neun Befehle von Profi-Grafik 64. Ihre Aufgabe ist es vor allem, die hochauflösende Grafik einzuschalten und die Parameter für andere Zwischenbefehle zu setzen. Die Befehle, die sich auf die Hardware beziehen, möchte ich so kurz wie möglich behandeln, weil deren Theorie schon ausführlich im Grafikkurs von H. Ponnath behandelt wurde.

1. SCREEN nr.

Da wäre als erstes der SCREEN-Befehl.

Durch ihn bestimmt man die Nummer des Bildschirms, den man anwählen will. Der Parameter »nr.« kann 0 oder 1 sein. Die Bitmap von Screen0 nimmt den Bereich von $A000-$BFFF ein und das Video-RAM den Bereich von $8C00-$8FFF.

Bei Screen1 sind dies die Bereiche $E000-$FFFF für die Bitmap und $CC00-$CFFF für das Video-RAM.

Übrigens wird im Register »Scrnum« nicht die Nummer selbst abgelegt, wie man denken könnte, sondern das High-Byte der Bitmap-Anfangsadresse.

Dies ist deshalb möglich, weil die Bytes $AO und $EO bei einer Bit-Abfrage die Flags unterschiedlich beeinflussen. Dies wird beispielsweise beim HiRes-Befehl ausgenutzt.

2. HIRES

Der Befehl dient nur dazu, den Bildschirm, der mit SCREEN festgelegt wurde, einzuschalten. Das Zustandekommen der einzelnen Werte, mit denen die VIC-Register versorgt werden, soll in der nächsten Folge dieses Kurses beschrieben werden. Ganz Ungeduldige können im H. Ponnaths Grafikkurs, Ausgabe 7/84 nachschauen.

3. MULTI

Schaltet den Multicolormodus ein. Ansonsten wie HIRES.

4. TEXT

Stellt die ursprünglichen Werte in den VIC-Registern wieder her, schaltet also auf den Textbildschirm zurück. Dieser Befehl wird auch bei jedem Warmstart (Programmende) und bei einem Druck auf die RUN/STOP-Taste ausgeführt. Man kann die Grafikbefehle also nur im Programm-Modus verwenden. Der Vorteil dabei ist, daß bei einer Fehlermeldung automatisch in den Text-Modus geschaltet wird.

5. CLEAR

Dieser Befehl löscht die Bitmap des mit SCREEN angewählten Bildschirms.

```
2
1000:  8390                    .OPT OO,P1
                        ;
                        ;GRAFIKROUTINEN 'PROFI-GRAFIK 64'
                        ;
1020:  8390                    *=  $8390           ;STARTADRESSE
                        ;
1030:  8390            GETCOM  =   $AEFD   ;PRUEFT AUF KOMMA
1035:  8390            GETBYTE =   $B79E   ;HOLT BYTE INS X-REGISTER
1040:  8390            GETADR  =   $B7EB   ;ADRESSWERT NACH $14/$15 + GETBYTE
1045:  8390            CHRGET  =   $0073   ;HOLT NAECHSTES ZEICHEN
1050:  8390            CHRGOT  =   $0079   ;HOLT LETZTES ZEICHEN
1055:  8390            ILLEGAL =   $B248   ;FEHLERMELDUNG 'ILLEGAL QUANTITY'
                        ;
1065:  8390            SCRNUM  =   $9FF1   ;AKTUELLE BILDSCHIRMNUMMER
                        ;
1075:  8390 20 9E B7 SCREEN    JSR  GETBYTE
1080:  8393 E0 02              CPX  #2
1085:  8395 90 03              BCC  SCROK
1090:  8397 4C 48 B2           JMP  ILLEGAL
1095:  839A BD A1 83 SCROK     LDA  SCRTAB,X
1100:  839D 8D F1 9F           STA  SCRNUM         ;MERKEN
1105:  83A0 60                 RTS
                        ;
1115:  83A1 A0 E0      SCRTAB  .BYT $A0,$E0
                        ;
1125:  83A3 AD 00 DD HIRES     LDA  $DD00          ;16K-BEREICH,DEN DER VIC ADRESS
IERT
1130:  83A6 29 FC              AND  #%11111100     ;FESTLEGEN $C000-$FFFF
1135:  83A8 2C F1 9F           BIT  SCRNUM
1140:  83AB 70 02              BVS  *+4            ;BEI $E0 IST DAS V-FLAG GESETZT
 !
1145:  83AD 09 01              ORA  #%00000001     ;BEREICH VON $8000-$BFFF
1150:  83AF 8D 00 DD           STA  $DD00
1155:  83B2 AD 18 D0           LDA  $D018          ;POSITION DES VIDEORAMS
1160:  83B5 09 2B              ORA  #%00101000     ;FESTLEGEN $0C00-$0FFF
1165:  83B7 8D 18 D0           STA  $D018
1170:  83BA 20 EA B3           JSR  TEX1
1175:  83BD AD 11 D0           LDA  $D011          ;EINZELPUNKTMODUS EINSCHALTEN
1180:  83C0 09 20              ORA  #%00100000     ;BIT5=1
1185:  83C2 8D 11 D0           STA  $D011
1190:  83C5 60                 RTS
                        ;
1200:  83C6 20 A3 83 MULTI     JSR  HIRES          ;HIRES-MODUS EIN
1205:  83C9 AD 16 D0           LDA  $D016          ;MULTICOLORMODUS EINSCHALTEN
1210:  83CC 09 10              ORA  #%00010000     ;BIT4=1
1215:  83CE 8D 16 D0           STA  $D016
1220:  83D1 60                 RTS
                        ;
1230:  83D2 AD 00 DD TEXT      LDA  $DD00          ;VIC ADRESSIERT JETZT WIEDER
1235:  83D5 09 03              ORA  #%00000011     ;BEREICH VON $0000-$3FFF
1240:  83D7 8D 00 DD           STA  $DD00
1245:  83DA AD 18 D0           LDA  $D018          ;VIDEORAM AB $0400-$07FF
1250:  83DD 29 D7              AND  #%11010111
1255:  83DF 8D 18 D0           STA  $D018
1260:  83E2 AD 11 D0           LDA  $D011          ;EINZELPUNKTMODUS ABSCHALTEN
1265:  83E5 29 DF              AND  #%11011111
1270:  83E7 8D 11 D0           STA  $D011
1275:  83EA AD 16 D0 TEX1      LDA  $D016          ;MULTICOLORMODUS ABSCHALTEN
1280:  83ED 29 EF              AND  #%11101111
1285:  83EF 8D 16 D0           STA  $D016
1290:  83F2 60                 RTS
                        ;
1300:  83F3            HELP    =   $20             ;HILFSZEIGER
                        ;
1310:  83F3 AD F1 9F CLEAR     LDA  SCRNUM
1315:  83F6 85 21              STA  HELP+1
1320:  83F8 A9 00              LDA  #0
1325:  83FA 85 20              STA  HELP           ;ZEIGER AUF ANFANG DER BITMAP
1330:  83FC AB                 TAY
1335:  83FD A2 20              LDX  #32            ;32 BLOECKE = 8 KBYTE
1340:  83FF 91 20      CLOOP   STA  (HELP),Y       ;BYTE LOESCHEN
1345:  8401 C8                 INY
1350:  8402 D0 FB              BNE  CLOOP
1355:  8404 E6 21              INC  HELP+1         ;NAECHSTER BLOCK
1360:  8406 CA                 DEX
1365:  8407 D0 F6              BNE  CLOOP
1370:  8409 60                 RTS
                        ;
1380:  840A 20 9E B7 HICOL     JSR  GETBYTE        ;ZEICHENFARBE HOLEN
1385:  840D 86 20              STX  HELP           ;MERKEN
1390:  840F 20 FD AE           JSR  GETCOM
1395:  8412 20 9E B7           JSR  GETBYTE        ;HINTERGRUNDFARBE
1400:  8415 86 21              STX  HELP+1         ;MERKEN
1405:  8417 A5 20              LDA  HELP           ;ZEICHENFARBE
1410:  8419 0A                 ASL  A
1415:  841A 0A                 ASL  A              ;MAL 16
1420:  841B 0A                 ASL  A
1425:  841C 0A                 ASL  A
1430:  841D 18                 CLC
1435:  841E 65 21              ADC  HELP+1         ;+ HINTERGRUNDFARBE
1440:  8420 20 35 B4           JSR  HIC1           ;MIT DIESEM WERT VIDEORAM FUELL
EN
1445:  8423 20 79 00           JSR  CHRGOT
1450:  8426 C9 2C              CMP  #","           ;WENN NOCH EIN KOMMA FOLGT,
1455:  8428 D0 27              BNE  MODE-1
1460:  842A 20 73 00           JSR  CHRGET
1465:  842D 20 9E B7           JSR  GETBYTE        ;DANN 3.ZEICHENFARBE HOLEN
1470:  8430 8A                 TXA
1475:  8431 A2 D8              LDX  #$D8           ;UND DAS FARBRAM ($D800-$DFFF)
1480:  8433 D0 0A              BNE  HIC2           ;DAMIT FUELLEN
                        ;
1490:  8435 2C F1 9F HIC1      BIT  SCRNUM
1495:  8438 70 03              BVS  SCR1A
1500:  843A A2 BC              LDX  #$BC           ;SCREEN0 VIDEORAM VON $BC00-$BF
FF
1505:  843C 2C                 .BYT #2C            ;BIT-OPCODE
1510:  843D A2 CC      SCR1A   LDX  #$CC           ;SCREEN1 VIDEORAM VON $CC00-$CF
FF
1515:  843F 86 21      HIC2    STX  HELP+1
1520:  8441 A0 00              LDY  #0
1525:  8443 84 20              STY  HELP           ;ZEIGER AUF ANFANG VIDEORAM
1530:  8445 A2 04              LDX  #4             ;4 BLOECKE SIND ZU FUELLEN
1535:  8447 91 20      HLOOP   STA  (HELP),Y
1540:  8449 C8                 INY
1545:  844A D0 FB              BNE  HLOOP
1550:  844C E6 21              INC  HELP+1         ;NAECHSTER BLOCK
1555:  844E CA                 DEX
1560:  844F D0 F6              BNE  HLOOP
1565:  8451 60                 RTS
                        ;
1575:  8452            PLOTMODE =  $9FF2           ;AKTUELLER PLOTMODUS
                        ;
1585:  8452 20 9E B7 MODE      JSR  GETBYTE
1590:  8455 E0 03              CPX  #3
1595:  8457 90 03              BCC  MODEOK
1600:  8459 4C 48 B2 ILLMODE   JMP  ILLEGAL
1605:  845C BD 63 84 MODEOK    LDA  MODETAB,X
1610:  845F 8D F2 9F           STA  PLOTMODE       ;MODUS MERKEN
1615:  8462 60                 RTS
                        ;
1625:  8463 00 40 80 MODETAB   .BYT 0,64,128
                        ;
1635:  8466            MULTICOL =  $9FF3           ;ZEICHENFARBE FUER MULTI
                        ;
1645:  8466 20 9E B7 INK       JSR  GETBYTE        ;ZEICHENFARBE SETZEN
1650:  8469 E0 04              CPX  #4
1655:  846B B0 EC              BCS  ILLMODE        ;>=4, DANN FEHLER
1660:  846D 8E F3 9F           STX  MULTICOL       ;MERKEN
1665:  8470 60                 RTS
                        ;
1675:  8471            XLO     =   $14
1680:  8471            XHI     =   $15
                        ;
                        ;BYTENUMMER ERRECHNEN (HIRES-MODUS)
1695:  8471 4A         HIPLOT  LSR         ;Y-KOORD. SCHON IM AKKU !
1700:  8472 4A                 LSR         ;(Y/8)
1705:  8473 4A                 LSR
1710:  8474 A8                 TAY         ;INS Y-REGISTER
1715:  8475 18                 CLC         ;ALLE LOBYTES ADDIEREN
1720:  8476 8A                 TXA
1725:  8477 29 07              AND  #%00000111     ;(Y AND #7)
1730:  8479 79 32 85           ADC  MALTAB,Y       ;+ (320*Y/8)LO (C=0!)
1735:  847C 85 F7              STA  $F7
1740:  847E A5 14              LDA  XLO
1745:  8480 29 F8              AND  #%11111000     ;+ (XLO AND #248)
1750:  8482 65 F7              ADC  $F7
1755:  8484 85 F7              STA  $F7            ;NACH #F7
                        ;ADDITION DER HIBYTES
1765:  8486 B9 4B 85           LDA  MALTAB1,Y      ;(320*Y/8)HI
1770:  8489 6D F1 9F           ADC  SCRNUM         ;+ ANFANG BITMAP
1775:  848C 65 15              ADC  XHI            ;+ XHI
1780:  848E 85 F8              STA  $F8            ;NACH #F8
1785:  8490 A5 14              LDA  XLO            ;BITPOSITION ERRECHNEN
1790:  8492 29 07              AND  #%00000111     ;(XLO AND #7)
1795:  8494 A8                 TAY
1800:  8495 B9 64 85           LDA  HOCHTAB,Y      ;2↑(7-YREGISTER)
1805:  8498 60                 RTS
                        ;
1815:  8499 4A         MUPLOT  LSR         ;(Y/8)
1820:  849A 4A                 LSR
1825:  849B 4A                 LSR
1830:  849C A8                 TAY
1835:  849D A5 14              LDA  XLO
1840:  849F 29 FC              AND  #%11111100     ;(XLO AND #252)
1845:  84A1 0A                 ASL         ;MAL 2 (BIT7 INS CARRY !)
1850:  84A2 85 F7              STA  $F7            ;NACH #F7
1855:  84A4 B9 4B 85           LDA  MALTAB1,Y      ;(320*Y/8)HI
1860:  84A7 6D F1 9F           ADC  SCRNUM         ;+ANFANG BITMAP
1865:  84AA 85 F8              STA  $F8            ;NACH #F8
1870:  84AC 8A                 TXA
1875:  84AD 29 07              AND  #%00000111     ;(Y AND #7)
1880:  84AF 79 32 85           ADC  MALTAB,Y       ;+(320*Y/8)LO
1885:  84B2 A8                 TAY         ;INS Y-REGISTER
1890:  84B3 A5 14              LDA  XLO            ;BITPOSITION ERRECHNEN
1895:  84B5 29 03              AND  #%00000011     ;(XLO AND #3)
1900:  84B7 AA                 TAX
1905:  84B8 BD 6C 85           LDA  MULTAB,X       ;BITWERT LADEN
1910:  84BB 38                 SEC
1915:  84BC 60                 RTS
                        ;
1925:  84BD 20 EB B7 PLOT      JSR  GETADR         ;KOORDINATEN HOLEN
1930:  84C0 AD 16 D0           LDA  $D016
1935:  84C3 29 10              AND  #%00010000
1940:  84C5 D0 2B              BNE  MPLOT          ;MULTICOLORMODUS
1945:  84C7 E0 C8              CPX  #200
1950:  84C9 B0 F1              BCS  PLOT-1         ;YKOORD.>199 (C=1)
1955:  84CB A4 15              LDY  XHI
1960:  84CD F0 09              BEQ  OK
1965:  84CF 88                 DEY
1970:  84D0 D0 E9              BNE  PLOT-2         ;XHI>1 (C=1)
1975:  84D2 A5 14              LDA  XLO            ;XHI=1, DANN XLO TESTEN
1980:  84D4 C9 40              CMP  #<320
1985:  84D6 B0 E4              BCS  PLOT-1         ;XLO>$40 (C=1)
1990:  84D8 78         OK      SEI
1995:  84D9 A9 34              LDA  #$34           ;SPEICHER AUF RAM UMSCHALTEN
2000:  84DB 85 01              STA  1              ;UM BIT-MAP LESEN ZU KOENNEN
2005:  84DD 8A                 TXA
2010:  84DE 48                 PHA         ;Y-KOORD. MERKEN
2015:  84DF 20 71 84           JSR  HIPLOT         ;BYTENUMMER BERECHNEN
2020:  84E2 A0 00              LDY  #0
2025:  84E4 2C F2 9F           BIT  PLOTMODE
2030:  84E7 70 37              BVS  LOESCH
2035:  84E9 30 3A              BMI  INVERT
2040:  84EB 11 F7              ORA  ($F7),Y        ;PUNKT SETZEN
2045:  84ED D0 38              BNE  STORE          ;UNBEDINGTER SPRUNG
                        ;
2055:  84EF E0 C8      MPLOT   CPX  #200
2060:  84F1 B0 C9              BCS  PLOT-1
2065:  84F3 A5 15              LDA  XHI            ;XHI<>0 (C=1)
2070:  84F5 D0 C4              BNE  PLOT-2
2075:  84F7 A5 14              LDA  XLO
2080:  84F9 C9 A0              CMP  #160
2085:  84FB B0 BE              BCS  PLOT-2         ;XLO>159 (C=1)
2090:  84FD 78                 SEI
2095:  84FE A9 34              LDA  #$34           ;SPEICHER AUF RAM UMSCHALTEN
2100:  8500 85 01              STA  1
2105:  8502 8A                 TXA
2110:  8503 48                 PHA         ;YKOORD. MERKEN
2115:  8504 20 99 84           JSR  MUPLOT         ;BYTENUMMER ERRECHNEN
2120:  8507 2C F2 9F           BIT  PLOTMODE
2125:  850A 70 14              BVS  LOESCH
2130:  850C 30 17              BMI  INVERT
2135:  850E 48                 PHA         ;PUNKT SETZEN
2140:  850F 49 FF              EOR  #255           ;VORHER LOESCHEN
2145:  8511 31 F7              AND  ($F7),Y
2150:  8513 91 F7              STA  ($F7),Y
2155:  8515 AE F3 9F           LDX  MULTICOL       ;ZEICHENFARBE LADEN
2160:  8518 68                 PLA
2165:  8519 3D 70 85           AND  MULTAB1,X      ;BITMUSTER DER FARBE SETZEN
2170:  851C 11 F7              ORA  ($F7),Y
2175:  851E D0 07              BNE  STORE
2180:  8520 49 FF      LOESCH  EOR  #255           ;PUNKT LOESCHEN
2185:  8522 31 F7              AND  ($F7),Y
2190:  8524 2C                 .BYT #2C
2195:  8525 51 F7      INVERT  EOR  ($F7),Y        ;PUNKT INVERTIEREN
2200:  8527 91 F7      STORE   STA  ($F7),Y        ;BITMUSTER SETZEN
2205:  8529 68                 PLA         ;YKOORD. WIEDERHERSTELLEN
2210:  852A AA                 TAX
2215:  852B A9 37      PLOTEND LDA  #$37           ;NORMALE SPEICHERKONFIGURATION
2220:  852D 85 01              STA  1
2225:  852F 58                 CLI
2230:  8530 18                 CLC         ;C=0 WENN PUNKT GESETZT
2235:  8531 60                 RTS
                        ;
2245:  8532 00 40 80 MALTAB    .BYT 0,<320,<640,<960  ;MULTIPLIKATIONSTABELLE
2250:  8536 00 40 80           .BYT $00,$40,$80,$C0   ;MAL 320
2255:  853A 00 40 80           .BYT $00,$40,$80,$C0   ;LOBYTES
2260:  853E 00 40 80           .BYT $00,$40,$80,$C0
2265:  8542 00 40 80           .BYT $00,$40,$80,$C0
2270:  8546 00 40 80           .BYT $00,$40,$80,$C0
2275:  854A 00                 .BYT $00
                        ;
2285:  854B 00 01 02 MALTAB1   .BYT 0,>320,>640,>960
2290:  854F 05 06 07           .BYT $05,$06,$07,$08   ;MAL 320
2295:  8553 0A 0B 0C           .BYT $0A,$0B,$0C,$0D   ;HIBYTES
2300:  8557 0F 10 11           .BYT $0F,$10,$11,$12
2305:  855B 14 15 16           .BYT $14,$15,$16,$17
2310:  855F 19 1A 1B           .BYT $19,$1A,$1B,$1C
2315:  8563 1E                 .BYT $1E
                        ;
2325:  8564 80 40 20 HOCHTAB   .BYT $80,$40,$20,$10  ;ZWEIERPOTENZEN
2330:  8568 08 04 02           .BYT $08,$04,$02,$01
                        ;
2340:  856C C0         MULTAB  .BYT %11000000      ;XPOSITION0
2345:  856D 30                 .BYT %00110000      ;XPOSITION1
2350:  856E 0C                 .BYT %00001100      ;XPOSITION2
2355:  856F 03                 .BYT %00000011      ;XPOSITION3
                        ;
2365:  8570 00         MULTAB1 .BYT %00000000      ;FARBE0=HINTERGRUND
2370:  8571 55                 .BYT %01010101      ;FARBE1
2375:  8572 AA                 .BYT %10101010      ;FARBE2
2380:  8573 FF                 .BYT %11111111      ;FARBE3
```

Listing 1. Die ersten neun Befehle von »Profi-Grafik 64«.

6. HICOL zf,hf (,c3)

Der Befehl HICOL setzt die Farben im Video-RAM des mit SCREEN angewählten Bildschirms. Im HiRes-Modus brauchen nur die Parameter »zf« und »hf« für Zeichenfarbe und Hintergrundfarbe angegeben werden. Folgt noch ein Komma, so wird der Parameter »c3« geholt und damit das Farb-RAM gefüllt. Dieser Parameter braucht nur im Multicolor-Modus angegeben zu werden. Dann gelten die drei Parameter als Zeichenfarben 1, 2 und 3.

7. MODE m

Dieser Befehl gestattet es, die Wirkungsweise des PLOT-Befehls zu beeinflußen. Der Parameter »m« darf zwischen 0 und 2 liegen, wobei bedeuten:

0 = Punkt setzen
1 = Punkt löschen
2 = Punkt invertieren

Damit der PLOT-Befehl zwischen den einzelnen Modi unterscheiden kann, wird das Register »Plotmode« durch einen Bit-Befehl abgefragt. Dabei gilt folgende Definition:

0 = Punkt setzen (Z-Flag gesetzt)
64 = Punkt löschen (V-Flag gesetzt)
128 = Punkt invertieren (N-Flag gesetzt)

	0	8	16	312	x
	0	8	16	312	
	1	9	17	313	
	2	10	18	314	
	3	11	19	315	
	4	12	20	316	
	5	13	21	317	
	6	14	22	318	
	7	15	23	319	
8	320	328	336	632	
	321	329	.	633	
	322	330	.	634	
	323	331	.	635	
	324	332	.	636	
	325	333	.	637	
	326	334	.	638	
	327	335	.	639	
16					
	.	.			
	.	.			
192	7680	7688		7992	
	7681	.		7993	
	7682	.		7994	
	7683	.		7995	
	7684	.		7996	
	7685	.		7997	
	7686	.		7998	
	7687	.		7999	
y					

Bild 1. Aufbau der Bitmap: die Anfangsadresse liegt bei 8192; die Punkte zeigen die einzelnen Positionen auf dem Bildschirm

Das Register »Plotmode« wird im MODE-Befehl 0 entsprechend gesetzt.

8. INK co

Dieser Befehl ist nur im Multicolor-Modus wirksam. Mit ihm wird die Zeichenfarbe festgelegt. Der Parameter »co« darf zwischen 0 und 3 liegen, wobei 0 = Hintergrundfarbe bedeutet. Die Farbe wird im Register »Multicol« abgelegt.

9. PLOT x,y

So, nun endlich kommen wir zum ersten interessanten Befehl, bei dem die Rechnerei in Maschinensprache anfängt.

Der PLOT-Befehl dient dazu, ein Bit in der Bitmap, das durch die Koordinaten »x« und »y« festgelegt wird, zu setzen, zu löschen oder zu invertieren. Die x-Koordinate darf sich dabei zwischen 0 und 319 bewegen, die y-Koordinate zwischen 0 und 199, wobei der Ursprung des Koordinatensystems in der linken oberen Bildschirmecke liegt. Nun ist es, wie Sie wohl wissen, nicht so einfach herauszufinden, welches Bit in welchem Byte zu einer bestimmten Bildschirmkoordinate gehört. Um das festzustellen, müssen wir uns den Aufbau der Bitmap anschauen (Bild 1).

Sie sehen, daß die ersten acht Byte untereinander liegen, die nächsten acht rechts daneben und so weiter, insgesamt 40 Spalten mal 8 Byte = 320 Byte. Im gleichen Stil sind 25 Zeilen untereinander aufgebaut. Das ergibt zusammen 25 mal 320 Byte = 8000 Byte. Der Bildschirmaufbau hat also große Ähnlichkeit mit dem des Textbildschirms. Nun gilt es, einen Algorithmus zu finden, der uns zu einer Bildschirmkoordinate x, y das entsprechende Byte in der Bitmap sowie die Bitposition innerhalb dieses Bytes liefert. Überlegen wir uns zunächst, wie sich die x-Koordinate auf die Byte-Position auswirkt. Wir müssen hier zwischen

```
programm : pg-mse          8000 8390
-----------------------------------

8000 : 20 80 09 80 c3 c2 cd 38  ad
8008 : 30 20 bc f6 20 e1 ff d0  09
8010 : 0c 20 76 80 20 a3 fd 20  31
8018 : 18 e5 20 7b e3 4c 72 fe  03
8020 : 20 a3 fd 20 90 fd 20 76  fc
8028 : 80 20 5b ff 58 a2 0b bd  d1
8030 : 8e 80 9d 00 03 ca 10 f7  1c
8038 : 20 bf e3 a0 0f a9 00 99  b6
8040 : f0 9f 88 10 fa 85 37 85  e8
8048 : 33 a9 80 85 38 85 34 a9  f4
8050 : a0 8d f1 9f a9 5a a0 82  1c
8058 : 20 1e ab 20 30 e4 20 6f  00
8060 : 82 a2 fb 9a 20 d2 83 8a  41
8068 : 30 03 4c 3a a4 4c 74 a4  3c
8070 : 20 d2 83 4c 83 a4 20 15  6c
8078 : fd a9 9a 8d 24 03 a9 80  a4
8080 : 8d 25 03 a9 22 8d 26 03  c3
8088 : a9 81 8d 27 03 60 64 80  00
8090 : 70 80 3b 81 fe 81 33 82  0d
8098 : 86 ae a5 99 f0 03 4c 66  37
80a0 : f1 a5 d3 85 ca a5 d6 85  4a
80a8 : c9 ad f0 9f d0 03 4c 32  33
80b0 : e6 8a d0 31 18 ad fe 01  23
80b8 : 65 fe 85 14 ad ff 01 69  32
80c0 : 00 85 15 20 13 a6 08 a6  a0
80c8 : 14 a5 15 20 cd bd 28 90  85
80d0 : 07 a9 5f 20 d2 ff d0 05  02
80d8 : a9 20 20 d2 ff a9 00 a4  8a
80e0 : d6 85 ca 84 c9 20 32 e6  f0
80e8 : 08 48 c9 5f d0 14 a5 14  df
80f0 : 8d fe 01 a5 15 8d ff 01  b1
80f8 : 68 28 a9 0d 20 d2 ff 4c  b2
8100 : 7b a4 c9 0d d0 19 a4 c8  db
8108 : 88 30 0c b1 d1 c9 3a b0  97
8110 : 0e c9 30 90 0a b0 f1 a9  62
8118 : 00 8d f0 9f 4c 74 a4 68  db
8120 : 28 60 48 a5 9a c9 03 f0  25
8128 : 03 4c d5 f1 ad 11 d0 29  fe
8130 : 20 d0 04 68 4c 16 e7 68  ac
8138 : 4c 16 e7 a6 7a a0 04 84  24
8140 : 0f bd 00 02 10 07 c9 ff  cf
8148 : f0 3e e8 d0 f4 c9 20 f0  ab
8150 : 37 85 08 c9 22 f0 55 24  cc
8158 : 0f 70 2d c9 3f d0 04 a9  02
8160 : 99 d0 25 c9 30 90 04 c9  0f
8168 : 3c 90 1d 84 71 a0 00 84  e9
8170 : 0b 88 86 7a ca c8 e8 bd  c2
8178 : 00 02 38 f9 9e a0 f0 f5  65
8180 : c9 80 d0 2f 05 0b a4 71  c1
8188 : e8 c8 99 fb 01 c9 00 f0  fa
8190 : 38 38 e9 3a f0 04 c9 49  8f
8198 : d0 02 85 0f 38 e9 55 d0  76
81a0 : a0 85 08 bd 00 02 f0 e0  52
81a8 : c5 08 f0 dc c8 99 fb 01  94
81b0 : e8 d0 f0 a6 7a e6 0b c8  ae
81b8 : b9 9d a0 10 fa b9 9e a0  a3
81c0 : d0 b5 f0 0f bd 00 02 10  8d
81c8 : bd 99 fd 01 c6 7b a9 ff  e0
81d0 : 85 7a 60 a0 00 b9 ef 82  51
81d8 : d0 02 c8 e8 bd 00 02 38  4d
81e0 : f9 ef 82 f0 f5 c9 80 d0  e1
81e8 : 04 05 0b d0 99 a6 7a e6  d2
81f0 : 0b c8 b9 ee 82 10 fa b9  b3
81f8 : ef 82 d0 e0 f0 c6 10 0f  1c
8200 : 24 0f 30 0b c9 ff f0 07  88
8208 : c9 cc b0 06 4c 24 a7 4c  41
8210 : f3 a6 38 e9 cb aa 84 49  58
8218 : a0 ff ca f0 08 c8 b9 ef  16
8220 : 82 10 fa 30 f5 c8 b9 ef  db
8228 : 82 30 05 20 47 ab d0 f5  09
8230 : 4c ef a6 20 73 00 20 3c  52
8238 : 82 4c ae a7 c9 cc 90 04  ce
8240 : c9 f5 90 06 20 79 00 4c  4f
8248 : ed a7 38 e9 cc 0a aa bd  97
8250 : a0 82 48 bd 9f 82 48 4c  c3
8258 : 73 00 93 12 20 50 52 4f  5f
8260 : 46 49 2d 47 52 41 46 49  5a
8268 : 4b 20 36 34 20 20 00 a9  2e
8270 : 01 a8 91 2b 20 33 a5 18  f1
8278 : a5 22 69 02 85 2d a5 23  68
8280 : 69 00 85 2e 4c 63 a6 20  cb
8288 : eb b7 86 fe 38 a5 14 e5  9d
8290 : fe 8d fe 01 a5 15 e9 00  df
8298 : 8d ff 01 ee f0 9f 60 6e  ad
82a0 : 82 86 82 8f 83 a0 83 c5  cf
82a8 : 83 d1 83 f2 83 09 84 51  88
82b0 : 84 65 84 bc 84 00 00 00  e8
82b8 : 00 00 00 00 00 00 00 00  b9
82c0 : 00 00 00 00 00 00 00 00  c1
82c8 : 00 00 00 00 00 00 00 00  c9
82d0 : 00 00 00 00 00 00 00 00  d1
82d8 : 00 00 00 00 00 00 00 00  d9
82e0 : 00 00 00 00 00 00 00 00  e1
82e8 : 00 00 00 00 00 00 00 4f  87
82f0 : 4c c4 41 55 54 cf 53 43  31
82f8 : 52 45 45 ce 48 49 52 45  bb
8300 : d3 4d 55 4c 54 c9 54 45  c8
8308 : 58 d4 43 4c 45 41 d2 48  5f
8310 : 49 43 4f cc 4d 4f 44 c5  54
8318 : 49 4e cb 50 4c 4f d4 00  18
8320 : 00 00 00 00 00 00 00 00  21
8328 : 00 00 00 00 00 00 00 00  29
8330 : 00 00 00 00 00 00 00 00  31
8338 : 00 00 00 00 00 00 00 00  39
8340 : 00 00 00 00 00 00 00 00  41
8348 : 00 00 00 00 00 00 00 00  49
8350 : 00 00 00 00 00 00 00 00  51
8358 : 00 00 00 00 00 00 00 00  59
8360 : 00 00 00 00 00 00 00 00  61
8368 : 00 00 00 00 00 00 00 00  69
8370 : 00 00 00 00 00 00 00 00  71
8378 : 00 00 00 00 00 00 00 00  79
8380 : 00 00 00 00 00 00 00 00  81
8388 : 00 00 00 00 00 00 00 00  89
```

Listing 2. MSE-Listing: Der erste Teil einer Basic-Erweiterung.

HiRes- und Multicolor-Grafik unterscheiden. Während im HiRes-Modus jedes Bit einem Punkt entspricht, benötigt man im Multicolor-Modus zwei Bits, um einen Punkt darzustellen. Deshalb gibt es dort nur 160 Punkte auf der x-Achse. Die Tabellen 1 und 2 zeigen, wie sich x-Koordinate und Byte-Nummer in der Bitmap zueinander verhalten, wenn wir die y-Koordinate gleich 0 setzen.

Sie sehen, daß im HiRes-Modus die unteren drei Bit für die Byte-Nummer unwichtig sind und deshalb gelöscht werden müssen. Dies geschieht mit der AND-Funktion:

```
xlo and #%11111000
```

Da die x-Werte größer als 255 sein können, müssen wir auch ein Highbyte berücksichtigen, bei dem allerdings nur das Bit 0 gesetzt sein kann, weil der höchste x-Wert 319 = $013F ist. Für die HiRes-Grafik sähe der vorläufige Algorithmus also so aus:

```
256 * xhi+(xlo and #248)
```

Da es im Multicolor-Modus nur 160 x-Werte gibt, brauchen wir dort kein Highbyte zu berücksichtigen. Sie sehen in Tabelle 2, daß beim Lowbyte die unteren zwei Bit keine Rolle spielen und gelöscht werden müssen: xlo and #%11111100

Mit dieser Berechnungsweise würden wir allerdings nur auf eine maximale Byte-Nummer von 156 kommen. Der Punkt würde also anstatt am rechten Bildrand in der Bildschirmmitte gesetzt. Deshalb müssen alle Byte-Nummern noch mit 2 multipliziert werden, um die richtige Byte-Nummer zu erhalten:

```
2 * (xlo and #252)
```

Wie wirkt sich nun die y-Koordinate auf die Byte-Nummer aus?

Wenn diese nicht größer als 7 wird, kann sie direkt zur Byte-Nummer addiert werden. Wird y größer als 7, gelangen wir in eine neue Bildschirmzeile und müssen jeweils 320 Byte addieren. Um zu ermitteln, in welcher Zeile wir sind, müssen wir y durch 8 teilen und gelangen so zur Formel:

```
(y and #7) +320 * (y/8)
```

Y=0 x-Koordinate	HiRes-Modus Bytennummer hi low
0-7	0 00000000 = 0
8-15	0 00001000 = 8
16-23	0 00010000 = 16
.	.
.	.
312-319	1 00111000 = 312
.	.
.	.

Tabelle 1. So verhalten sich die x-Werte und die Bytenummer zueinander bei y=0.

y=0 x-Koordinate	Multicolor-Modus Bytennummer
0-3	00000000 = 0 * 2 = 0
4-7	00000100 = 4 * 2 = 8
8-11	00001000 = 8 * 2 = 16
.	
.	
156-159	10011100 = 156 * 2 = 312

Tabelle 2. Multicolor-Modus: x-Werte und Bytenummer für y=0.

Damit hätten wir die beiden Algorithmen zusammen. Sie lauten:

```
1. Hires-Modus:
Byte-Nummer = 256 * xhi +
(xlo and #248) + (Y and #7)
+ 320 * (Y/8)
2. Multi-Modus:
Byte-Nummer = 2 * (xlo and
#252) + (Y and #7) + 320 *
(Y/8)
```

Nun kommen wir zur Formulierung in Maschinensprache. Dazu legen wir erst fest, wie wir der Berechnungsroutine die Koordinaten übergeben. Und zwar soll der y-Wert im x-Register und der x-Wert in den Registern $14/$15 übergeben werden. Schauen sie sich nun die Routine »Hiplot« im Assemblerlisting an. Dies ist die Unterroutine zur Berechnung der Byte-Nummer im HiRes-Modus. Das Geheimnis der Routine liegt in der Art, wie der Term 320*y/8 berechnet wird. Diese Möglichkeit besteht allerdings nur dann, wenn der Multiplikant (320) konstant und der Multiplikator (y/8) in einem gewissen Bereich (hier 0 bis 24) schwankt. In so einem Fall berechnet man alle möglichen Ergebnisse vorher und legt diese in zwei Tabellen (Low- und Highbyte) ab. In der Berechnungsroutine braucht man dann nur noch den Multiplikator als Zeiger ins y-Register zu übertragen und lädt sich das benötigte Ergebnis aus der Tabelle. Hier sind die Ergebnisse in den Tabellen »Maltab« für die Lowbyte und »Maltab1« für die Highbyte abgelegt. Entscheidend bei der Berechnungsroutine ist weiterhin die Anordnung der einzelnen Summanden. Achten Sie außerdem immer auf den Zustand des Carry-Flags, das für die Addition wichtig ist! Die Nummer des Bytes in der Bitmap wird in den beiden Byte $F7/$F8 abgelegt. Am Ende der Routine wird noch die Bitposition innerhalb des Bytes errechnet. Dazu isolieren wir die x-Position mittels (xlo and #7) und laden den Akku mit der Zweier-Potenz, die der x-Position entspricht. Diese Zweier-Potenzen sind ebenfalls in einer Tabelle (Hochtab) abgelegt.

Im Multi-Modus sieht das Feststellen der Bitposition etwas anders aus. Da wir hier nur vier x-Positionen in einem Byte haben, isolieren wir diese mittels (xlo and #3). Dann laden wir den Akku mit dem Wert, bei dem die beiden Bits, die dieser x-Position entsprechen, gesetzt sind. Für die x-Position 0 wäre dies der Wert %11000000. Die vier Werte finden Sie in der Tabelle »Multab«.

Jetzt kommen wir zur Besprechung der PLOT-Routine an sich. Nachdem die Koordinaten aus dem Basic-Text geholt wurden, wird auf Multicolor-Modus geprüft. Dann verlaufen in beiden Teilzweigen der Routine (HPLOT und MPLOT) die Wege ähnlich. Zuerst werden die Koordinaten auf ihre Richtigkeit überprüft. Falls sie den zulässigen Bereich überschreiten, wird die Routine frühzeitig mit gesetztem Carry-Flag verlassen. Ansonsten wird in den Unterroutinen »Hiplot« oder »Muplot« die Byte-Nummer berechnet und die Bit-Position im Akku bereitgestellt. Nun wird das Register »Plotmode« mittels BIT-Befehl abgefragt und je nach Modus verzweigt. Die einzelnen logischen Verknüpfungen machen Sie sich am besten an Hand von Beispielen klar (ausprobieren!).

Zu erklären bleibt noch, wie ein Punkt im Multicolor-Modus gesetzt wird. Sie sehen, daß, nachdem die Bit-Position auf den Stack gerettet wurde, die betreffenden beiden Bits erst einmal gelöscht werden. Dann wird die Bit-Position wiedergeholt und die aktuelle Zeichenfarbe ins x-Register geladen. Sodann wird das Bit-Muster der momentanen Farbe hergestellt, indem es mit dem Bit-Muster der aktuellen Farbe AND-verknüpft wird. Die Bit-Muster der Farben stehen in einer Tabelle »Multab«. Sie erkennen, daß das Bit-Muster an allen vier x-Positionen steht. Versuchen Sie nun herauszufinden, warum die beiden Bits zuerst gelöscht werden müssen, bevor das Muster der neuen Farbe gesetzt werden kann!

Am Ende der Plot-Routine wird dann noch das Carry-Flag gelöscht, um anzuzeigen, daß die Koordinaten in Ordnung waren. Außer den Grafikbefehlen steht Ihnen auch noch ein Mini-Toolkit zur Verfügung. Zum einen ein OLD-Befehl, mit dem Sie ein versehentlich gelöschtes Programm wiederholen können sowie ein AUTO-Befehl, der die automatische Zeilennumerierung übernimmt. Dazu geben sie ein: AUTO Zeilennummer, Schrittweite (0-255). Verlassen können Sie den AUTO-Modus durch Drücken von »RETURN« bei einer neuen Zeile. Trifft der AUTO-Befehl auf eine schon vorhandene Zeile, so wird hinter der Zeilennummer ein Pfeil nach links ausgegeben. Geben Sie danach nur RETURN ein, wird die Zeile nicht überschrieben. Wollen Sie die Zeile überschreiben, dann ist vorher der Pfeil zu löschen.

Eingabehinweise

Listing 2 ist zunächst mit dem MSE einzugben und zu speichern. Im Anschluß daran muß der C 64 aus- und wieder eingeschaltet werden. Laden Sie nun einen Assembler, tippen den Quellcode (Listing 1) ab, lassen ihn übersetzen und laden anschließend Listing 2 absolut (LOAD »PG-MSE«, 8,1). Jetzt muß der Speicherbereich von $800 bis $8574 mit einem Monitor gespeichert werden. Das Programm läßt sich nun mit SYS 64738 aktivieren.

(Andreas Schömann/cg)