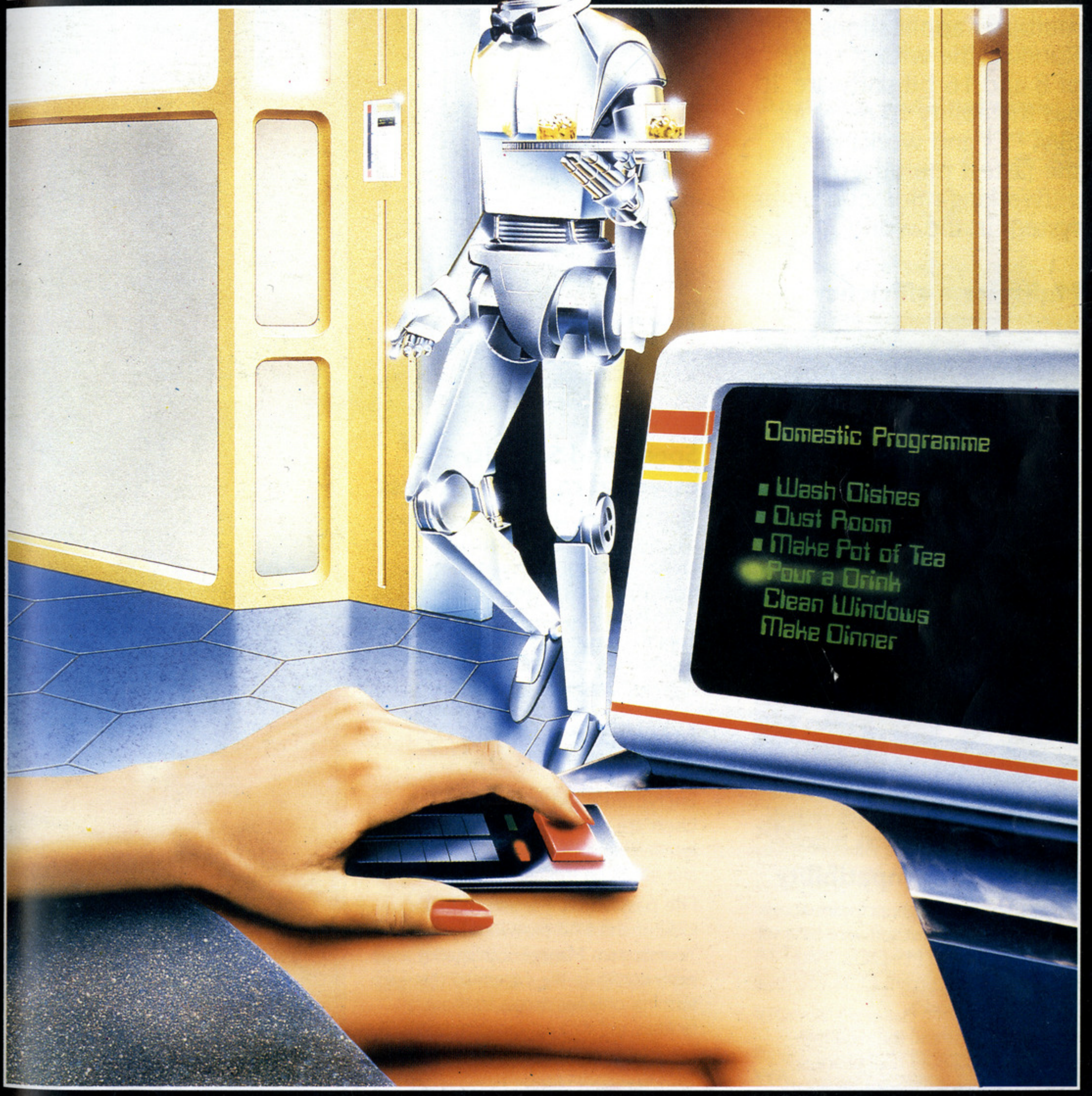
Australian Personal Computer

FREE BASIC CONVERSION WALLCHART INSIDE

ISSN 0725—4115 NZ $3.50
REGISTERED BY AUSTRALIA POST PUBLICATIONS VBP 3691

MARCH 1986 $3.50

AUSTRALIA'S TOP SELL[illegible] COMPUTER MAGAZINE



MAKING MICROS MORE HUMAN

Special feature includes natural languages in business and mutationary programming

```
(RESTARTS it)";AT 20,3; INK 2;"Press any key to begin..."
1250 FOR p=1 TO 7
1260 PRINT AT 4,2; PAPER P; INK 9;"          ";AT 5,2;"  K A L
E I D O S C O P E  ";AT 6,2;"          "
1270 IF INKEY$="" THEN  GO TO 1290
1280 RETURN
1290 PAUSE 9: NEXT p: GO TO 1250
1300
1390 REM I/A Stop
1400 FOR s=0 TO 1
1410 LET s=0
1420 IF INKEY$="r" OR INKEY$="R" THEN  LET s=1
1430 NEXT s
1440 RETURN
1450
1590 REM Select Graphics Pointer
1600 LET g=INT (RND*8.999)+2
1610 IF g<=8 THEN  GO SUB g*1000
1620 IF g=9 THEN  GO SUB 2500
1630 IF g=10 THEN  GO SUB 3500
1640 RETURN
1650
1990 REM Selects Graphics
2000 LET a=144
2010 LET b=145
2020 LET c=146
2030 LET d=147
2040 RETURN
2050

2500 LET a=160
2510 LET b=161
2520 LET c=162
2530 LET d=163
2540 RETURN
2550
3000 LET a=150
3010 LET b=152
3020 LET c=153
3030 LET d=151
3040 RETURN
3050
3500 LET a=164
3510 LET b=164
3520 LET c=164
3530 LET d=164
3540 RETURN
3550
4000 LET a=148
4010 LET b=148
4020 LET c=148
4030 LET d=148
4040 RETURN
4050
5000 LET a=149
5010 LET b=149
5020 LET c=149
5030 LET d=149
5040 RETURN
5050
6000 LET a=154
6010 LET b=155
6020 LET c=155
6030 LET d=154
6040 RETURN
6050
7000 LET a=156
7010 LET b=157
7020 LET c=157
7030 LET d=156
7040 RETURN
7050
8000 LET a=158
8010 LET b=159
8020 LET c=159
8030 LET d=158
8040 RETURN
8050
8990 REM Graphics Data
9000 DATA 3,15,31,63,127,127,255,255
9010 DATA 192,240,248,252,254,254,255,255
9020 DATA 255,255,127,127,63,31,15,3
9030 DATA 255,255,254,254,252,248,240,192
9040 DATA 255,231,195,129,129,195,231,255
9050 DATA 24,60,126,255,255,126,60,24
9060 DATA 255,254,252,248,240,224,192,128
9070 DATA 1,3,7,15,31,63,127,255
9080 DATA 255,127,63,31,15,7,3,1
9090 DATA 128,192,224,240,248,252,254,255
9100 DATA 240,224,192,128,1,3,7,15
9110 DATA 15,7,3,1,128,192,224,240
9120 DATA 192,192,252,252,63,63,3,3
9130 DATA 3,3,63,63,252,252,192,192
9140 DATA 1,2,4,8,16,32,64,128
9150 DATA 128,64,32,16,8,4,2,1
9160 DATA 3,12,16,32,64,64,128,128
9170 DATA 192,48,8,4,2,2,1,1
9180 DATA 128,128,64,64,32,16,12,3
9190 DATA 1,1,2,2,4,8,48,192
9200 DATA 24,24,24,255,255,24,24,24
```

Amstrad Pascal GRAPHLIB

by Robin Shipp

This program is a library of routines written in Hisoft Pascal for the Amstrad CPC 464. It is intended for use in scientific/mathematic programs requiring graphical output, but could also be used to produce business charts or graphs. When tne routines that make up GRAPHLIB are used, real coordinates should be used, rather than the computer's. This means that if you want to plot a graph to display data between, say, −2 and 5, then these values should be used rather than screen coordinates. This is set up by the initialising routine, GSTART, which specifies the user's coordinates for the bottom-left and top right-hand corners of the graphics window.

When the GRAPHLIB routines have been entered and checked, they should be saved as the file GRAPHLIB.PAS. This can then be included in the user's program using $F just after the VAR declarations. GRAPHLIB includes the global variables used by the routines. The example program gives a demonstration of how the routines should be used.

```
1000 {    Graphics Library - Graphlib.pas
1010       (c) Robin Shipp 1985 }
1020
1030 {$i-}
1040
1050   { Global Variables  }
1060  gxr,gyr,          {Current position IN user's coordinates}
1070  gxc,gxm,gyc,gym, {Values FOR 'transform' }
1080 ,gxmin,gxmax,gymin,gymax {Limits OF graphics window IN
1090                          user's coordinates}
1100       :real;
1110
1120 PROCEDURE gtext(switch:boolean);
1130
1140 {  switch = true : text at graphics cursor on
1150    switch = false: text at graphics cursor off}
1160
1170 BEGIN
1180  ra:=chr(ord(switch));
1190  user(#bb63)  {TXTSETGRAPHIC}
1200  END;
1210
1220 PROCEDURE gwindow(left,right,top,bottom:integer);
1230
1240 {  Sets size OF graphics window
1250    Arguments are IN pixel coordinates}
1260
1270 BEGIN
1280  rde:=right; rhl:=left;
1290  user(#bbcf); {GRAWINWIDTH}
1300  rde:=top; rhl:=bottom;
1310  user(#bbd2)  {GRAWINHEIGHT}
1320  END;
1330
1340 PROCEDURE gclear;
1350
1360 {  Clears graphics window}
1370
1380 BEGIN
1390  user(#bbdb)  {GRACLEARWINDOW}
1400  END;
1410
1420 PROCEDURE gpen(ink:integer);
1430
1440 {  Sets ink FOR graphics pen}
1450
1460 BEGIN
1470  ra:=chr(ink);
1480  user(#bbde)  {GRASETPEN}
1490  END;
1500
1510 PROCEDURE gpaper(ink:integer);
1520
1530 {  Sets ink FOR graphics paper}
1540
1550 BEGIN
1560  ra:=chr(ink);
1570  user(#bbe4)  {GRASETPAPER}
1580  END;
1590
1600 PROCEDURE mode(m:integer);
1610
1620 {  Sets screen mode}
1630
```

```
1640 BEGIN
1650  ra:=chr(m);
1660  user(#bc0e)  {SCRSETMODE}
1670  END;
1680
1690 PROCEDURE setink(ink,colour1,colour2:integer);
1700
1710 {  Sets ink - both colours same -> steady
1720              different colours -> flashing}
1730
1740 BEGIN
1750  ra:=chr(ink); rb:=chr(colour1); rc:=chr(colour2);
1760  user(#bc32)  {SCRSETINK}
1770  END;
1780
1790 PROCEDURE transform(x,y:real;VAR xp,yp:integer);
1800
1810 {  Converts user's coordinates (x,y) into
1820    pixel coordinates (xp,yp) - internal
1830    use only}
1840
1850 BEGIN
1860  xp:=round(gxc+gxm*x);
1870  yp:=round(gyc+gym*y)
1880  END;
1890
1900 PROCEDURE move(x,y:real);
1910
1920 {  Moves TO (x,y) }
1930
1940 VAR
1950  xp,yp:integer;
1960
1970 BEGIN
1980  gxr:=x; gyr:=y;
1990  transform(x,y,xp,yp);
2000  rde:=xp; rhl:=yp;
2010  user(#bbc0)  {GRAMOVEABSOLUTE}
2020  END;
2030
2040 PROCEDURE rmove(dx,dy:real);
2050
2060 {  Moves by (dx,dy) relative TO current position}
2070
2080 BEGIN
2090  move(gxr+dx,gyr+dy)
2100  END;
2110
2120 PROCEDURE rdraw(dx,dy:real;line:integer);
2130
2140 FORWARD;
2150
2160 PROCEDURE draw(x,y:real;line:integer);
2170
2180 {  Draws line from current position TO (x,y)
2190     line = 0 : unbroken line
2200     line > 0 : dashed line, higher number -> longer dashes}
2210
2220 VAR
2230  i,np,xp,yp:integer;
2240  dx,dy,xi,yi:real;
2250
2260 BEGIN
2270  IF line<1 THEN BEGIN
2280   transform(x,y,xp,yp);
2290   rde:=xp; rhl:=yp;
2300   user(#bbf6); {GRALINEABSOLUTE}
2310   gxr:=x; gyr:=y
2320   END
2330  ELSE BEGIN
2340   dx:=x-gxr; dy:=y-gyr;
2350   np:=round((sqrt(sqr(gxm*dx)+sqr(gym*dy)))/(8*line));
2360   xi:=0.5*dx/np; yi:=0.5*dy/np;
2370   FOR i:=1 TO np DO BEGIN
2380    rdraw(xi,yi,0);
2390    rmove(xi,yi)
2400    END;
2410   draw(x,y,0)
2420   END
2430  END;
2440
2450 PROCEDURE rdraw;
2460
2470 {  Draws line from current position TO (dx,dy)
2480    relative TO it
2490    See 'draw' FOR action OF 'line' }
2500
2510 BEGIN
2520  draw(gxr+dx,gyr+dy,line)
2530  END;
2540
2550 PROCEDURE gstart(xmin,ymin,xmax,ymax:real);
2560
2570 {  1.  Initialises such that, IN user's coordinates,
2580        (xmin,ymin) is bottom left OF graphics window
2590        AND (xmax,ymax) is top right
2600
2610     2.  Sets TO blue pen on yellow paper
2620
2630     3.  Clears graphics window
2640
2650     4.  Sets current point TO centre OF window}
2660
2670 VAR
2680  left,right,top,bottom:integer;
2690
2700 BEGIN
2710  setink(1,24,24); setink(0,1,1);
2720  gpen(0); gpaper(1); gclear;
2730  user(#bbd5); {GRAGETWWIDTH}
2740  left:=rde; right:=rhl;
2750  user(#bbd8); {GRAGETWHEIGHT}
2760  top:=rde; bottom:=rhl;
2770  gxm:=(right-left)/(xmax-xmin);
2780  gxc:=left-gxm*xmin;
2790  gym:=(top-bottom)/(ymax-ymin);
2800  gyc:=bottom-gym*ymin;
2810  move(0.5*(xmax+xmin),0.5*(ymax+ymin));
2820  gxmin:=xmin; gxmax:=xmax;
2830  gymin:=ymin; gymax:=ymax
2840  END;
2850
2860 PROCEDURE axes(xdiv,ydiv:real;
2870   xfield,xdecpl,yfield,ydecpl:integer);
2880
2890 {  Draws axes along x=0 AND y=0 WITH ticks at intervals
2900    OF xdiv AND ydiv along x AND y axes, respectively
2910
2920    Labels axes WITH x AND y values at each tick, written
2930    IN field OF xfield AND yfield, AND WITH xdecpl AND
2940    ydecpl decimal places, respectively
2950
2960    Note: field <1 disables labelling OF axis }
2970
2980 VAR
2990  mode,charsize,xshift,yshift:integer;
3000  ticklength:real;
3010
3020 BEGIN
3030  gtext(true);
3040  user(#bc11); {SCRGETMODE}
3050  mode:=ord(ra);
3060  CASE (mode) OF
3070   0: charsize:=32;
3080   1: charsize:=16;
3090   2: charsize:=8
3100   END;
3110  ticklength:=0.04*(gymax-gymin);
3120  xshift:=-(xfield*charsize) DIV 2; yshift:=-4;
3130  move(0,0);
3140  WHILE (gxr<gxmax) DO BEGIN
3150   rdraw(xdiv,0,0);
3160   rdraw(0,ticklength,0);
3170   rmove(0,-ticklength);
3180   IF (xfield>1) THEN BEGIN
3190    rde:=xshift; rhl:=yshift;
3200    user(#bbc3); {GRAMOVERELATIVE}
3210    write(gxr:xfield:xdecpl);
3220    move(gxr,gyr)
3230    END
3240   END;
3250  move(0,0);
3260  WHILE (gxr>gxmin) DO BEGIN
3270   rdraw(-xdiv,0,0);
3280   rdraw(0,ticklength,0);
3290   rmove(0,-ticklength);
3300   IF (xfield>1) THEN BEGIN
3310    rde:=xshift; rhl:=yshift;
3320    user(#bbc3); {GRAMOVERELATIVE}
3330    write(gxr:xfield:xdecpl);
3340    move(gxr,gyr)
3350    END
3360   END;
3370  ticklength:=0.04*(gxmax-gxmin);
3380  xshift:=-(yfield*charsize)-4; yshift:=8;
3390  move(0,0);
3400  WHILE (gyr<gymax) DO BEGIN
3410   rdraw(0,ydiv,0);
3420   rdraw(ticklength,0,0);
3430   rmove(-ticklength,0);
3440   IF (yfield>1) THEN BEGIN
3450    rde:=xshift; rhl:=yshift;
3460    user(#bbc3); {GRAMOVERELATIVE}
3470    write(gyr:yfield:ydecpl);
3480    move(gxr,gyr)
3490    END
3500   END;
3510  move(0,0);
3520  WHILE (gyr>gymin) DO BEGIN
3530   rdraw(0,-ydiv,0);
3540   rdraw(ticklength,0,0);
3550   rmove(-ticklength,0);
3560   IF (yfield>1) THEN BEGIN
3570    rde:=xshift; rhl:=yshift;
3580    user(#bbc3); {GRAMOVERELATIVE}
3590    write(gyr:yfield:ydecpl);
3600    move(gxr,gyr)
3610    END
3620   END;
3630  gtext(false)
3640  END;
3650
3660 PROCEDURE gchar(c:char);
3670 {  Writes single character at current position
3680
3690    Current position reset ready TO write next character}
3700
3710
3720 VAR
3730  xp,yp:integer;
3740
3750 BEGIN
3760  ra:=c;
3770  user(#bbfc); {GRAWCHAR}
3780  user(#bbc6); {GRAASKCURSOR}
3790  xp:=rde; yp:=rhl;
3800  gxr:=(xp-gxc)/gxm;
3810  gyr:=(yp-gyc)/gym
3820  END;
3830
3840 PROCEDURE plot(x,y:real;point:integer);
3850
3860 {  Plots point centered on (x,y)
3870    'point' is ascii value OF character TO be plotted
3880      (most useful values are 159,202,203,227,230,
3890        232,233,244,245)
3900
3910    Note: IF point<32, pixel is plotted}
3920
3930 VAR
3940  ascii,xp,yp,mode,offset:integer;
3950
```

```
BEGIN
 ascii:=point MOD 256;
 IF (ascii<32) THEN BEGIN
  gxr:=x; gyr:=y;
  transform(x,y,xp,yp);
  rde:=xp; rhl:=yp;
  user(#bbea)  {GRAPLOTABSOLUTE}
  END
 ELSE BEGIN
  move(x,y);
  user(#bc11); {SCRGETMODE}
  mode:=ord(ra);
  CASE (mode) OF
   0: offset:=-16;
   1: offset:=-8;
   2: offset:=-4
  END;
  rde:=offset; rhl:=8;
  user(#bbc3); {GRAMOVERELATIVE}
  gchar(chr(ascii));
  move(x,y)
  END
 END;

PROCEDURE rplot(dx,dy:real;point:integer);

{  Plots point at (dx,dy) relative TO current position

   See 'plot' FOR action OF 'point' }

BEGIN
 plot(gxr+dx,gyr+dy,point)
 END;

{$I+}

```

```
PROGRAM grdemo1;

{  Demonstration PROGRAM FOR Graphics Library

   Draws curves OF y=x^2 AND y=x^3

     (c) Robin Shipp 1985}

VAR
 xi,yi:real;

   {$f graphlib.pas}

BEGIN   {Main PROGRAM}
 mode(1);
 gwindow(150,639,399,0);
 gstart(-2.1,-10.5,2.1,10.5);
 setink(2,13,13); setink(3,26,26);
 gpaper(2); gclear;
 axes(1.0,2.0,2.0,3,0);
 move(1.8,-0.75); gchar('X');
 move(-0.25,9.5); gchar('Y');
 gpen(3); xi:=-2.0; yi:=4.0;
 move(xi,yi);
 WHILE (xi<2.0) DO BEGIN
  xi:=xi+0.25; yi:=sqr(xi);
  draw(xi,yi,0)
  END;
 move(-1.8,4.2);
 gtext(true); write('Y=X^2'); gtext(false);
 xi:=-2.0; yi:=-8.0;
 move(xi,yi);
 WHILE (xi<2.0) DO BEGIN
  xi:=xi+0.25; yi:=xi*sqr(xi);
  draw(xi,yi,3)
  END;
 move(-1.8,-8.0);
 gtext(true); write('Y=X^3'); gtext(false)
 END.
```

Commodore 64 Polyphonic 64

by Jonathan Wadie

MICROTEX 666

Still keying in programs? Forget it! This program is available for telesoftware downloading on Microtex 666 (page *6663#.)

All the instructions for this program are contained within it. Suffice it to say that the program is a polyphonic synthesiser program which allows three different notes to be played at once. It has eight pre-programmed instruments and also allows you to program your own.

Full instructions for use are contained within the program.

```
0 REM ***********************************
1 REM *          POLYPHONIC  64         *
2 REM *           BY JON WADIE          *
3 REM *     RED BARON SOFTWARE 1985     *
4 REM ***********************************
5 PRINT"■":AD=008:SR=035:PL=000:PH=008
6 GOTO120
9 PRINT"■"
10 PRINT"    _ _   _ _ _   _ _   _"
15 PRINT"   _ // ///_ // // ///_ ///_ ";
20 PRINT"   // // // // // // // // //"
25 PRINT"   // // // // // // // // // ";
30 PRINT"   |_|_|_| |_|_|_|_| |_|_|_| |_|_|_";
35 PRINT"  "
40 PRINT"  ";
45 PRINT"|  |  |  |  |  |  |  |  |  |  |  |  |";
50 PRINT"|Q |W |E |R |T |Y |U |I |O |P |@ |* |↑|";
55 PRINT"_______________________________________";
70 PRINT"■■■■■■■■■■F1: TRIANGLE WAVEFORM"
80 PRINT"■■■■■■■■■F3: SAWTOOTH WAVEFORM"
90 PRINT"■■■■■■■■■F5:. PULSE WAVEFORM"
100 PRINT"■■■■■■■■■F7: MENU"
101 PRINT
102 RETURN
120 IFPEEK(49152)=32ANDPEEK(49153)=93THENGOTO300
130 FORC=49152TO49655STEP8:CK=C-256*INT(C/256):FORCC=0TO7:READM
   :CK=(CK+M)AND255
 140 POKEC+CC,M:NEXT
 150 IF(PEEK(63)+256*PEEK(64))<>CTHENPRINT"■■LINE"C"■ MISSING"
   :POKE49152,96:END
160 READM:IFM<>CKTHEN210
170 NEXT:GOTO300
180 POKE49278,AD:POKE49299,SR
190 POKE49256,PL:POKE49267,PH
200 SYS49152:GOTO300
210 PRINT"■■ERROR IN LINE"PEEK(63)+256*PEEK(64):POKE49152,96:END
299 REM *********MAIN MENU**************
300 PRINT"■■■■■■■■■■■■■■■■■■■■■■┌────┐"
301 PRINT"■■■■■■■■■■■■■■■■■■■■■|MENU|"
302 PRINT"■■■■■■■■■■■■■■■■■■■■■└────┘"
310 PRINT"■■■■■■■■■■■■■■■■SELECT SOUND (1)"
312 PRINT"■■■■■■■■■■■■■■■CREATE SOUND (2)"
314 PRINT"■■■■■■■■■■■■■■■KEYBOARD     (3)"
316 PRINT"■■■■■■■■■■■■■■■INSTRUCTIONS (4)"
318 PRINT"■■■■■■■■■■■■■■■QUIT PROGRAM (5)"
330 INPUT"■■■■■■■■■■■■■SELECT NUMBER REQUIRED ";SN
335 IFSN<1ORSN>5THEN330
340 IFSN=1THENGOSUB500:GOSUB9:GOTO180
341 IFSN=2THENGOSUB400:GOSUB9:GOTO180
342 IFSN=3THENGOSUB9:GOTO180
343 IFSN=4THENGOTO600
344 IFSN=5THENGOTO345
345 INPUT"■■■■ QUIT PROGRAM : ARE YOU SURE (Y/N) ";Q$
349 PRINT"■"
350 IFQ$="Y"THENPRINT"■":NEW
351 IFQ$="N"THEN300
399 REM *********CREATE SOUND***********
400 PRINT"■■■■■■■■■■■■■■■■■■■■CREATE SOUND"
405 PRINT"■■  THE SETTINGS WERE:-■"
409 PRINT"■        ┌──────────────────┬──────┐"
410 PRINT"         | ATTACK/DECAY     |";AD;TAB(31);"|"
411 PRINT"         ├──────────────────┼──────┤"
412 PRINT"         | SUSTAIN/RELEASE  |";SR;TAB(31);"|"
413 PRINT"         ├──────────────────┼──────┤"
414 PRINT"         |    LOW PULSE     |";PL;TAB(31);"|"
415 PRINT"         ├──────────────────┼──────┤"
416 PRINT"         |    HI PULSE      |";PH;TAB(31);"|"
417 PRINT"         └──────────────────┴──────┘"
420 PRINT"■■  NOW ENTER NEW VALUES FOR:-■"
460 INPUT"■        ATTACK/DECAY (000-255) ";AD
461 IFAD<000ORAD>255THENPRINT"■■■":GOTO460
465 INPUT"■        SUSTAIN/RELEASE (000-255) ";SR
466 IFSR<000ORSR>255THENPRINT"■■■":GOTO465
470 INPUT"■        LO PULSE (000-200) ";PL
471 IFPL<000ORPL>200THENPRINT"■■■":GOTO470
475 INPUT"■        HI PULSE (000-200) ";PH
476 IFPH<000ORPH>200THENPRINT"■■■":GOTO475
498 RETURN
499 REM *********SOUND MENU*************
500 PRINT"■■■■■■■■■■■■■■■■■■■■■SOUND MENU"
501 PRINT"■■■INSTRUMENT            WAVEFORM    NO.■"
502 PRINT"■___________            ________    ___"
510 PRINT"■■■■PIANO               [■TRIANGLE■]   (1)"
511 PRINT"■■■■FLUTE               [■TRIANGLE■]   (2)"
512 PRINT"■■■■HARPSICHORD         [■SAWTOOTH■]   (3)"
513 PRINT"■■■■WOODWIND            [■TRIANGLE■]   (4)"
514 PRINT"■■■■ORGAN               [■TRIANGLE■]   (5)"
515 PRINT"■■■■BANJO               [■ PULSE  ■]   (6)"
516 PRINT"■■■■ACCORDIAN           [■TRIANGLE■]   (7)"
517 PRINT"■■■■TRUMPET             [■SAWTOOTH■]   (8)"
540 INPUT"■■■■■■■■■■■■■■■■■■■■■■ENTER CHOICE";C
541 IFC<0ORC>8THEN540
545 IFC=1THENAD=9:SR=2:PL=200:PH=0:RETURN
546 IFC=2THENAD=96:SR=15:PL=0:PH=1:RETURN
547 IFC=3THENAD=9:SR=3:PL=0:PH=1:RETURN
548 IFC=4THENAD=135:SR=9:PL=0:PH=1:RETURN
549 IFC=5THENAD=0:SR=240:PL=0:PH=1:RETURN
550 IFC=6THENAD=9:SR=5:PL=200:PH=2:RETURN
551 IFC=7THENAD=102:SR=0:PL=1:PH=0:RETURN
552 IFC=8THENAD=96:SR=10:PL=3:PH=4:RETURN
598 RETURN
```

```
599 REM ********INSTRUCTIONS**********
600 PRINT"           ヨTL I ♦ I/I I 64T'
601 PRINT"                  ________"
602 POKE53272,23
605 PRINT" |HIS PROGRAM ALLOWS YOU TO USE ONE OF  ";
606 PRINT" 8  PRE-DEFINED SOUNDS OR TO CREATE YOUR";
607 PRINT"OWN SOUNDS.|HE KEYBOARD IS POLYPHONIC.  ";
608 PRINT"THIS MEANS YOUR CAN PLAY UP TO 3 NOTES  ";
609 PRINT"AT A TIME"
610 PRINT
611 PRINT" |HERE ARE 3 MAIN OPTIONS:-             ";
612 PRINT:PRINT" (ヨ1) ♦ELECT ♦OUND.....|HIS ALLOWS YOU TO";
613 PRINT"CHOOSE ONE OF THE PRESET SOUNDS. |HE    ";
614 PRINT"WAVEFORM TELLS YOU WHICH KEY TO PRESS   ";
615 PRINT"WHEN YOU ARE IN THE KEYBOARD MODE.      ";
616 PRINT" (ヨ2) -REATE ♦OUND.....|HIS ALLOWS YOU TO";
617 PRINT"DEFINE YOUR OWN SOUNDS BY MODIFYING THE ";
618 PRINT"♠/‾/♦/_ AND THE WAVEFORMS.              ";
619 PRINT" (ヨ3) ‘EYBOARD.....|HIS ALLOWS YOU TO USE";
620 PRINT"NEARLY 2 OCTAVES.USING THE DISPLAYED    ";
621 PRINT"KEYS.|HE FUNCTION KEYS LET YOU CHANGE   ";
622 PRINT"WAVEFORM WHILE YOU ARE PLAYING."
623 PRINT:PRINT"       L‾♦♦ ♠ I /‾I I‾ ‾I ‾ I‾ ‾"
625 GETK$:IFK$=""THEN625
626 POKE53272,21:GOTO300
49151 REM ****MACHINE CODE DATA********
49152 DATA32,93,192,32,28,192,32,169,2
49160 DATA192,32,1,193,162,50,160,0,30
49168 DATA136,208,253,202,208,250,32,199,224
49176 DATA193,76,3,192,120,162,7,169,178
49184 DATA127,141,0,220,72,173,1,220,218
49192 DATA157,244,193,104,56,106,202,16,94
49200 DATA240,88,162,63,169,255,157,252,154
49208 DATA193,202,16,250,162,0,160,0,15
49216 DATA189,244,193,134,251,162,7,10,230
49224 DATA72,144,5,169,0,153,252,193,36
49232 DATA200,104,202,16,242,166,251,232,213
49240 DATA192,64,208,228,96,162,24,169,207
49248 DATA0,157,0,212,202,16,250,169,78
49256 DATA160,141,2,212,141,9,212,141,98
49264 DATA16,212,169,15,141,3,212,141,253
49272 DATA10,212,141,17,212,169,0,141,254
49280 DATA5,212,141,12,212,141,19,212,58
49288 DATA169,249,141,6,212,141,13,212,255
49296 DATA141,20,212,169,15,141,24,212,54
49304 DATA169,255,141,52,3,141,53,3,201
49312 DATA141,54,3,169,64,141,55,3,22
49320 DATA96,174,52,3,48,24,189,252,238
49328 DATA193,240,8,169,0,157,252,193,108
49336 DATA76,198,192,173,55,3,141,4,2
49344 DATA212,169,255,141,52,3,174,53,227
49352 DATA3,48,24,189,252,193,240,8,133
49360 DATA169,0,157,252,193,76,227,192,194
49368 DATA173,55,3,141,11,212,169,255,211
49376 DATA141,53,3,174,54,3,48,24,212
49384 DATA189,252,193,240,8,169,0,157,160
49392 DATA252,193,76,0,193,173,55,3,161
49400 DATA141,18,212,169,255,141,54,3,217
49408 DATA96,162,0,134,251,173,52,3,103
49416 DATA48,11,173,53,3,48,6,173,11
49424 DATA54,3,48,1,96,189,130,193,218
49432 DATA168,185,252,193,240,3,32,39,112
49440 DATA193,232,224,23,208,223,96,173,124
49448 DATA52,3,16,3,76,80,193,173,124
49456 DATA53,3,16,3,76,105,193,152,137
49464 DATA141,54,3,189,153,193,141,14,176
49472 DATA212,189,176,193,141,15,212,173,95
49480 DATA55,3,9,1,141,18,212,96,95
49488 DATA152,141,52,3,189,153,193,141,80
49496 DATA0,212,189,176,193,141,1,212,188
49504 DATA173,55,3,9,1,141,4,212,182
49512 DATA96,152,141,53,3,189,153,193,60
49520 DATA141,7,212,189,176,193,141,8,155
49528 DATA212,173,55,3,9,1,141,11,213
49536 DATA212,96,57,60,14,15,9,22,101
49544 DATA23,17,20,30,31,25,38,39,103
49552 DATA33,36,46,41,44,54,55,49,246
49560 DATA52,135,134,162,223,62,193,107,196
49568 DATA60,57,99,190,75,15,12,69,225
49576 DATA191,125,131,214,121,115,199,124,108
49584 DATA33,35,37,39,42,44,47,50,247
49592 DATA53,56,59,63,67,71,75,79,195
49600 DATA84,89,94,100,106,112,119,173,45
49608 DATA255,193,240,6,169,16,141,55,251
49616 DATA3,96,173,254,193,240,6,169,62
49624 DATA32,141,55,3,96,173,253,193,138
49632 DATA240,6,169,64,141,55,3,96,230
49640 DATA173,0,194,240,6,104,104,169,198
49648 DATA0,133,198,96,0,0,0,0,155
```

Amstrad Disk ROM Disabling

by JR Wozniak

If you want to run some of your old CPC464 software on your new CPC6128, a few programs will not load because the disk ROM takes up an extra 1.25k. The following program releases this block of RAM and re-boots the system.

Another problem that may arise when trying to run CPC464 programs on the CPC6128 is that various software houses have written programs that access routines in firmware directly, instead of via the jump block tables as recommended by Amstrad. There is little that can be done about this without access to the source code.

```
30 MEMORY &9FFF: a=&A000
40 READ d$
50 WHILE d$<>"0"
60    POKE a, VAL("&"+d$): a=a+1:READ d$
70 WEND
80 CALL &A000
90 DATA 21,08,A0,0E,00,CD,16,BD,3E,C9,32,CB,BC,C3,06,C0,0
```

Zipsort

by K Riordan

"Some time ago (see Sought of sorts, June 1985 *APC*, page 171) I exposed the relative advantage in speed of a combined distribution count/"smart" bubble sort technique over the more familiar Quicksort in the ordering of string arrays. The initials of the string elements themselves formed the basis of the distribution count, which grouped together all elements with the same initial; the bubble sort completed the task by ordering each of the resultant sub-groups in turn.

However, since I had a specific task in mind — to read an unordered index from disk relative file, sort it and write it back to disk again — I was not particularly satisfied with the improved routine, which still required more than five minutes to sort a 500-element array of randomly-generated strings. It appeared

MICROTEX 666

Still keying in programs? Forget it! This program is available for telesoftware downloading on Microtex 666 (page *6663#.)