Australian Personal Computer

PC GRAPHICS FEATURE

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

SEPTEMBER 1986 $3.50

AUSTRALIA'S TOP SELLING COMPUTER MAGAZINE

NEC STEPS INTO LINE

'Go-it-alone' policy abandoned on APC IV

Amstrad CPC Promenu

by Kevin Nixon

This program allows menus to be easily included in your own programs. Examples of how to create and use menus within programs are given.

Two lines could not be printed and must be added to the program:

```
50370 LOCATE 1,v:CALL &BD19:
  PRINT"[CONTROL—X]"+m$(v)+
  SPACE$((wi)—LEN(m$(v)))+"
  [CONTROL—X];
50750 IF mh$<>"" THEN WINDOW
  £w,ml—1,mr,mt—2,mb:LOCATE
  £w,1,1:PRINT £w"[CONTROL—X]
  [SPACE]"+mh$+"[SPACE]
  [CONTROL—X]";
```

In these lines, type the bits between square brackets as single characters.

To create windows, certain variables have to be set and then a particular subroutine called. An example is line 130. The variables are:

ml — left position of window
mt — top of window
am — height of window
wi — width of window
w — the window number (0–7)
mh$ — the title of the window

GOSUB 50420 is used to create a window with a normal frame, GOSUB 50620 for one with a pull-down menu frame. If a window is accidentally positioned off-screen, it will be moved back on at the nearest point.

To create a menu, the options have to be in data statements as in line 180. The data is held as follows:

DATA menuno,type,title,xpos,ypos,option,option,. . .,*

menuno — used in conjunction with off(menuno,optionno)
type — 1 for ordinary, 2 for pull-down menu
title — any string, can be blank
xpos — left position for menu, 0 to centre on x plane
ypos — top position for menu, 0 to centre on y plane
option — the option name; if left blank, a line of separators will appear in its place
* — marks the end of the list of options

The maximum number of options in a menu is 23, or 22 if a title is used. Options within a menu can be switched off at any time using the off function:

off(menuno,optionno)=1

The result is to display the option in light type and prevent it from being selected until it's set to 0, which switches it on again.

The program only works on the 464 and 664, but to convert it for the 6128, delete lines 70 and 51110 to 51200. Then replace the RSX calls in lines 50030 and 50260 with the equivalent on the 6128 using the bank manager RSXs supplied with the machine.

To use the menus, use the cursor and copy keys. To incorporate them into your own programs, lines 50200–50260 do the selecting and v holds the value of the position of the selection bar at the end, so this can be used to determine the course of action to be taken.

The variables holding the key codes for up, down and select are ku, kd and kf respectively.

```
10 MODE 2:RESTORE 51110:GOSUB 51110
20 INK 0,26:INK 1,0:BORDER 20
30 FOR a=1 TO 25:PRINT STRING$(80,207);:NEXT
40 DRAW 639,0,1:DRAW 639,398:DRAW 0,398:DRAW 0,0
50 DIM m$(24),off(25,24)
60 ku=0:kd=2:kf=9
70 RESTORE 51110:GOSUB 51110
80  REM ****************
90  REM * EXAMPLE DATA *
100 REM *  NO NEED TO  *
110 REM *  TO TYPE IN  *
120 REM ****************
130 ml=6:mt=10:am=10:wi=38:w=1:GOSUB 50420
140 LOCATE £1,16,3:PRINT £1,"PRO MENU":LOCATE £1,13,5:PRINT £1,"By Kevin Nixon":
LOCATE £1,4,9:PRINT £1,"For the Amstrad CPC464 computer."
150 off(1,2)=1:REM ** Option 2 ( complicted ) is unnavailable.
160 REM ** DATA FOR EXAMPLE MENU **
170 CALL &BB18:RESTORE 180
180 DATA 1,2,Example,0,0,Creating,complicated,menus,,is easy,with,Pro menu,*
190 GOSUB 50030:GOTO 170
50000 REM ***************
50010 REM ** PRO MENU **
50020 REM ***************
50030 |COPY:w=0:v=0:wi=0:n=0:opt=1
50040 READ menu,type,mh$,ml,mt:IF type<1 THEN type=1:ELSE IF type>2 THEN type=2
50050 WHILE m$(n)<>"*":n=n+1:READ m$(n):WEND:am=n-1
50060 IF am<1 THEN am=1
50070 GOSUB 51080
50080 FOR n=1 TO am:IF LEN(m$(n))>wi THEN wi=LEN(m$(n))
50090 NEXT
50100 IF LEN(mh$)>wi THEN wi=LEN(mh$)
50110 IF ml=0 THEN ml=41-INT(wi/2)
50120 IF mt=0 THEN mt=13-INT(am/2)
50130 ON type GOSUB 50420,50620
50140 GOSUB 50990:REM ** PRINT OPTIONS **
50150 GOSUB 50200
50160 WINDOW £w,1,80,1,25:RETURN
50170 REM ***************
50180 REM * UP 'N' DOWN *
50190 REM ***************
50200 GOSUB 50360
50210 WHILE INKEY(kf)=-1
50220 IF INKEY(ku)=0 AND v>1 THEN GOSUB 50300:GOSUB 50810:GOSUB 50360
50230 IF INKEY(kd)=0 AND v<am THEN GOSUB 50300:GOSUB 50900:GOSUB 50360
50240 WEND
50250 CALL &BFF9,&C9,&1CED,&CF00:REM ** CLEAR KEYBOARD BUFFER **
50260 |SHOW:RETURN
50270 REM ********************
50280 REM * DELETE HIGHLIGHT *
50290 REM ********************
50300 IF v<1 THEN v=1 ELSE IF v>am THEN v=am
50310 LOCATE 1,v:CALL &BD19:PRINT m$(v)+SPACE$((wi)-LEN(m$(v)));
50320 RETURN
50330 REM ********************
50340 REM * PRINT HIGHLIGHT *
50350 REM ********************
50360 IF v<1 THEN v=1 ELSE IF v>am THEN v=am
";
50380 SOUND 1,50,2:RETURN
50390 REM ******************
50400 REM * ORDINARY MENU *
50410 REM ******************
50420 IF mh$="" AND am>23 THEN am=23:ELSE IF am>22 THEN am=22
50430 IF wi>70 THEN wi=70
50440 IF ml<2 THEN ml=2:mr=ml+wi
50450 IF (mt<3 AND mh$<>"") THEN mt=3:mb=mt+am
50460 IF (mt<2 AND mh$="") THEN mt=2:mb=mt+am
50470 IF (mt=1 AND mh$="") THEN mt=2:mb=mt+am
50480 mr=ml+wi-1:mb=mt+am-1
50490 IF mr>79 THEN mr=79:ml=mr-wi
50500 IF mb>24 THEN mb=24:mt=mb-am+1
50510 gl=ml*8-12:gr=(mr*8)+3:gb=398-mb*16-2
50520 IF mh$<>"" THEN gt=398-mt*16+44:ELSE gt=398-mt*16+20
50530 ORIGIN 0,0,gl-4,gr+4,gt+4,gb-4:CLG
50540 MOVE gl,gt:DRAW gr,gt,1:DRAW gr,gb:DRAW gl,gb:DRAW gl,gt
50550 MOVE gl-4,gt+4:DRAW gr+4,gt+4:DRAW gr+4,gb-4:DRAW gl-4,gb-4:DRAW gl-4,gt+4
50560 IF mh$<>"" THEN TAG:MOVE gl+(gr-gl)/2-LEN(mh$)*8/2,gt-6:PRINT mh$;:TAGOFF:
MOVE gl,gt-24:DRAW gr,gt-24
50570 WINDOW £w,ml,mr,mt,mb
50580 RETURN
50590 REM ******************
50600 REM * PULL DOWN MENU *
50610 REM ******************
```

```
50620 IF mh$<>"" AND am>22 THEN am=22
50630 IF mh$="" AND am>23 THEN am=23
50640 IF ml<2 THEN ml=2:mr=ml+wi
50650 IF (mt<3 AND mh$<>"") THEN mt=3:mb=mt+am
50660 IF (mt<2 AND mh$="") THEN mt=2:mb=mt+am
50670 IF mh$<>"" AND LEN(mh$)>wi-1 THEN wi=wi+1
50680 mr=ml+wi-1:mb=mt+am-1
50690 IF mr>79 THEN mr=79:ml=mr-wi+1
50700 IF mb>24 THEN mb=24:mt=mb-am+1
50710 gl=ml*8-12:gr=(mr*8)+3:gb=398-mb*16-2
50720 IF mh$<>"" THEN gt=398-mt*16+28:ELSE gt=398-mt*16+20
50730 ORIGIN 0,0,gl-4,gr+4,gt+4,gb-4:CLG
50740 MOVE gl-4,gt+4:DRAW gr+4,gt+4:DRAW gr+4,gb-4:DRAW gl-4,gb-4:DRAW gl-4,gt+4
 "+mh$+"
";
50760 WINDOW £w,ml,mr,mt,mb
50770 RETURN
50780 REM ************************************
50790 REM * SEARCH UP FOR NEAREST LEGAL OPTION *
50800 REM ************************************
50810 opt=v
50820 opt=opt-1
50830 IF opt=0 THEN 50850
50840 IF m$(opt)="" OR off(menu,opt)=1 THEN GOTO 50820
50850 IF opt>0 THEN v=opt
50860 RETURN
50870 REM ************************************
50880 REM * SEARCH DOWN FOR NEAREST LEGAL OPTION *
50890 REM ************************************
50900 opt=v
50910 opt=opt+1
50920 IF opt>am THEN RETURN:REM 51120
50930 IF m$(opt)="" OR off(menu,opt)=1 THEN GOTO 50910
50940 IF opt<am+1 THEN v=opt
50950 RETURN
50960 REM *****************
50970 REM * PRINT OPTIONS *
50980 REM *****************
50990 FOR n=1 TO am
51000 LOCATE 1,n
51010 IF m$(n)="" THEN PRINT STRING$(wi,45);:GOTO 51040
51020 PRINT m$(n);
51030 IF off(menu,n)=1 THEN PRINT CHR$(22);CHR$(1):PEN 0:LOCATE 1,n:PRINT STRING
$(wi,207);:PEN 1:PRINT CHR$(22);CHR$(0);
51040 NEXT:RETURN
51050 REM *******************************
51060 REM * FIND START POSITION FOR BAR *
51070 REM *******************************
51080 v=v+1:IF v=am THEN RETURN
51090 IF m$(v)="" OR off(menu,v)=1 THEN GOTO 51080
51100 RETURN
51110 REM RSX's
51120 REM |TURBO,|COPY,|SHOW
51130 MEMORY &3999
51140 FOR f=40000 TO 40071:READ a$:POKE f,VAL("&"+a$):NEXT
51150 CALL 40000
51160 DATA 01,4a,9c,21,84,9c,cd,d1,bc,c9,55,9c,c3,63,9c,c3,6c,9c
51170 DATA c3,78,9c,54,55,52,42,cf,43,4f,50,d9,53,48,4f,d7,00,21
51180 DATA 5f,00,3e,05,cd,68,bc,c9,21,00,c0,11,00,40,01,00,40,ed
51190 DATA b0,c9,21,00,40,11,00,c0,01,00,40,ed,b0,c9,00,00,00,00
51200 RETURN
```

MBasic Daisywheel Graphics

by Christopher Lawton

This program simulates graphics printing on a daisywheel. It has been written to work with a Brother HR25, but can be rewritten to work with other daisywheels.

The program uses the full stop as a notional pixel, and uses microjustification to move the print head by sufficiently small amounts. The printer head can be moved in 1/60ths of an inch horizontally and 1/16ths of an inch vertically. As the program moves the print head, it scans the computer screen. If a pixel on the screen is on, then it prints a full stop; otherwise it prints nothing.

The program takes a long time, however, because to print one screen of graphics it needs to effectively print 128,000 characters.

The printing routine is in lines 90–380, and lines 1000 onward should hold a routine to display the graphics to be drawn on screen two. A BLOAD statement could be used to load a previously-created graphics screen.

The program needs to use line lengths of 640 characters because it's printing a full stop every sixtieth of an inch. This is done by typing BASICA/S:640 from DOS.

The printer control codes used are:
line 130 — CHR$(27)+CHR$(&H1E)+CHR$(4) — set VMI to 3/48in
line 140 — CHR$(27)+CHR$(&H1F)+CHR$(3) — set HMI to 2/120in
line 200 — CHR$(32) — print a space
line 240 — CHR$(&H2E) — print a full stop
line 260 — CHR$(27)+CHR$(&H55) — advance paper by VMI
line 310 — CHR$(27)+CHR$(34) — reset auto line feed
line 320 — CHR$(27)+CHR$(83) — reset HMI and VMI

These codes can be used to run the program with other printers.

```
10 CLS:KEY OFF
20 SCREEN 2
30 GOSUB 390:                                   '   JUMP TO DRAW PICTURE.
40 '
50 '
60 '            BEGIN PRINT ROUTINE
70 '
80 '
90 OPEN "LPT1:" AS #1 LEN=640                   '   OPEN FILE TO PRINTER WITH
100 WIDTH#1,255 '                               '   LINE LENGTH OF 640 CHARACTERS
110 '
120 '
130 PRINT#1,CHR$(27)+CHR$(&H1E)+CHR$(3)         '   SET VERTICAL SPACING
140 PRINT#1,CHR$(27)+CHR$(&H1F)+CHR$(2)         '   SET HORIZONTAL SPACING
150 '
160 '
170 '
180 FOR T=0 TO 199                              '   BEGIN VERTICAL SCANNING
190 FOR TT=0 TO 639                             '   BEGIN HORIZONTAL SCANNING
200         IF POINT (TT,T) = 0 THEN
                    PRINT#1,CHR$(32);
                    :GOTO 250                   '   PRINT SPACE IF PIXEL IS OFF
210 '
220 '
230 '
240         PRINT#1,CHR$(&H2E);                 '   OTHERWISE PRINT FULL STOP
250     NEXT TT
260     PRINT#1,CHR$(27)+CHR$(&H55)             '   MOVE ROLLER UP ONE LINE
270 NEXT T
280 '
290 '
300 '
310 PRINT#1,CHR$(27)+CHR$(34)                   '  RESET PRINTER TO INITIAL
320 PRINT#1,CHR$(27)+CHR$(83)                   '        SETTINGS
330 '
340 '
350 CLOSE#1                                     '  CLOSE PRINTER FILE
360 '
370 '
380 END
390 '
400 '
410 '           ****   D R A W   P I C T U R E   ****
420 '
430 '
440 PRINT
450 LOCATE 25,1:PRINT "   Daisywheel
    Graphics"
460 PI=3.14159:SC=5/12                                  ASSIGN PI & SET SCREEN RATIO
470 X=250 :Y=0:CX=320:CY=100:SF=.95
480 C=COS(PI/5):S=SIN(PI/5)
490 C1=COS(PI/36):S1=SIN(PI/36)
500 FOR J=1 TO 40
```