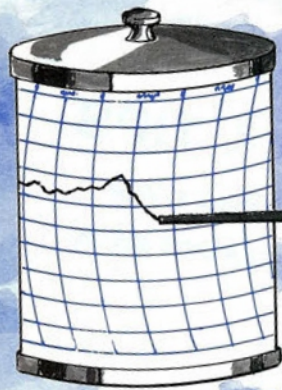
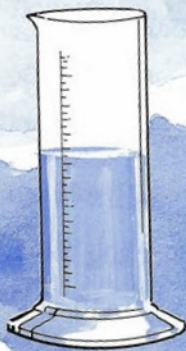
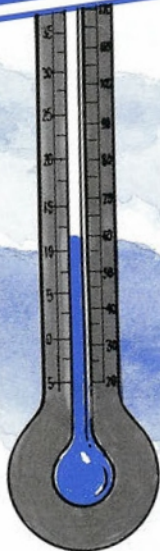


Vol.11 No.2 June 1992

BEEBUG

FOR THE
BBC MICRO &
MASTER SERIES



WEATHER STATION

• INPUT ROM FOR SWR

• EXPLORE: ADVENTURE GAME

• HYBRID'S MUSIC PUBLISHER REVIEW

• ELECTRIC FIELD LINES

FEATURES

Weather Station	
Input ROM	
Days and Nights in School	
DataSheet	
Fiddling with Field Lines	
Mr Toad's Machine Code Corner (3)	
BEEBUG Workshop: Replicating Polygonal Patterns	29
512 Forum	
Smart Renumber	
First Course: Direct Memory Access (2)	
Public Domain Software (5)	
BEEBUG Function/ Procedure Library (12)	

REVIEWS

7	The Hybrid Music System for the BBC Micro	47
9	Games Review: Explorer	51

REGULAR ITEMS

13	Editor's Jottings	4
15	News	5
21	RISC User	6
	Points Arising	42
26	Postbag	57
	Hints and Tips	59
29	Personal Ads	60
34	BEEBUG Membership	62
38	Magazine Disc	63

HINTS & TIPS

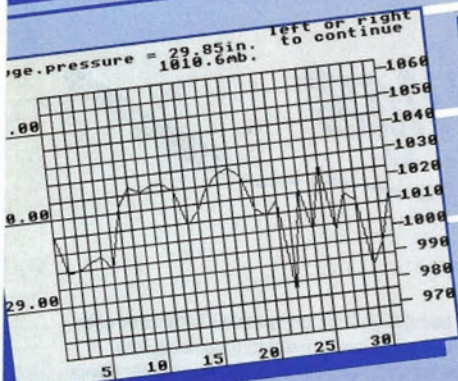
43	Hard Space and Rivers in View
52	BEEBUG Magazine Discs and the Master
54	Four into One

PROGRAM INFORMATION

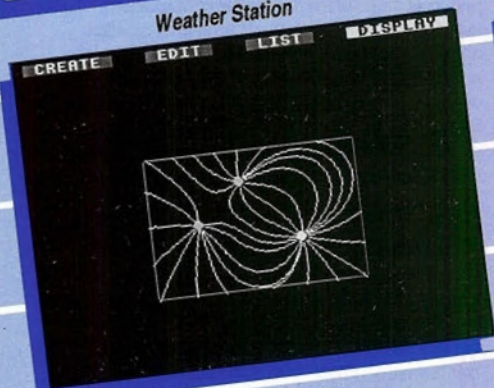
All listings published in BEEBUG magazine are produced directly from working programs. They are formatted using LISTO 1 and WIDTH 40. The space following the line number is to aid readability only, and may be omitted when the program is typed in. However, the rest of each line should be entered exactly as printed, and checked carefully. When entering a listing, pay special attention to the

difference between the digit one and a lower case l (L). Also note that the vertical bar character (Shift \) is reproduced in listings as |.

All programs in BEEBUG magazine will run on any BBC micro with Basic II or later, unless otherwise indicated. Members with Basic I are referred to the article on page 44 of BEEBUG Vol.7 No.2 (reprints



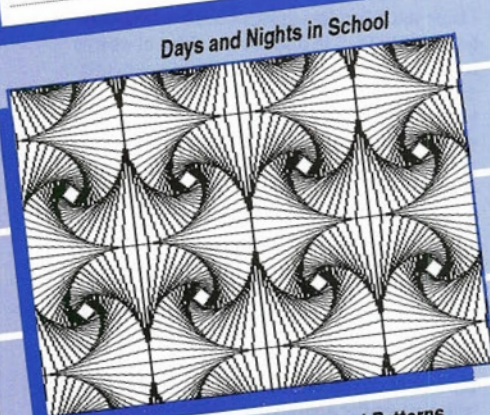
Weather Station



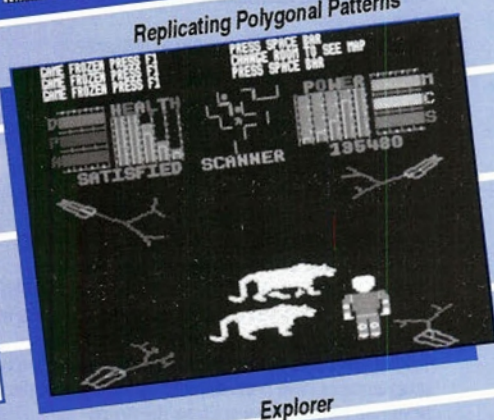
Fiddling with Field Lines

Part 1, bar 1. Exit: ESC, Update: TAB.
(Bach Two Part Invention No.6
K (q.=48) (+4) (3/8)
#1
[e3SB e5ED s=D^ ;
e3EFG]
#2
[s5D eC.B sA^ ;
e3RB C']
#3
[s4R e6F tGA ;

Hybrid Music Publisher



Replicating Polygonal Patterns



Explorer

available on receipt of an A5 SAE), and are strongly advised to upgrade to Basic II. Any second processor fitted to the computer should be turned off before the programs are run.

Where a program requires a certain configuration, this is indicated by symbols at the beginning of the article (as shown opposite). Any other requirements are referred to explicitly in the text of the article.

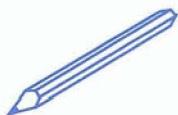


Program needs at least one bank of sideways RAM.



Program is for Master 128 and Compact only.

Editor's Jottings



I hope you found last month's tenth anniversary issue of BEEBUG of interest. For those of us who have been involved with BEEBUG from the early days, as contributors or members, it is quite fascinating looking back and browsing through previous issues. Of course, the style and presentation of the magazine changed as time went by, and we were able to introduce a more and more professional looking product.

One aspect though, of perhaps even greater importance, is the sheer variety and quality of the software which we have published through the pages of the magazines over the years. It really is amazing to see what can be achieved on a BBC micro, and the best of this software can often stand comparison with that for the Archimedes range.

Where the BBC micro loses out is primarily in memory capacity and in raw processing power, just the attributes which are essential to support the Wimp-based desktop approach used first by the Apple Macintosh, implemented on the Archimedes via RISC OS, and available now for PCs through Microsoft Windows.

In my view, much of the best software for the BBC micro appeared in the mid years - that is of the order of four to six years ago. In those days, the BBC micro was at the forefront of home micros and many clever and highly innovative programmers were writing programs for the Beeb. In recent years, such programmers have been seduced by the likes of the newer 32 bit machines such as the Acorn Archimedes range, and it is true to say that we receive fewer technically excellent contributions than we used to. Indeed it is interesting to see the wheel turning more than full circle, as users who have only recently acquired a BBC micro for the first time send in the results of their programming endeavours unknowingly duplicating the efforts of earlier contributors from years before.

With such a wealth of material published over the years, and now (from earlier issues at least) largely

unobtainable (but see our summer back issue offer with this magazine), we feel that updating and reprinting some of the best items from earlier issues performs a useful service for users. It makes available to newer readers a selection of excellent applications, and for those who have been subscribing longer we are able to update a program where appropriate (for example, to make it work on a Master 128 as well as the model B for which it was originally written) and incorporate any subsequent updates or amendments which have come to our attention.

We do still very much welcome new and original contributions for BEEBUG. Without your help in this way we would not now be celebrating ten years of successful publication of BEEBUG. And if you are not sure in what form to submit a contribution, we have recently revised and rewritten our *Notes of Guidance for Contributors*, which is available free of charge on receipt of an A5 size (or larger) SAE, and we do pay for anything published.

TENTH ANNIVERSARY DISC

The BEEBUG *Tenth Anniversary Disc* is still available for the special members' price of £4.95 (plus £1.00 p&p). This contains a selection of some of the best BEEBUG programs over the years (as detailed elsewhere). Other collections of previous BEEBUG programs are also available (see centre pages).

BEEBUG OPENING HOURS

Please note that during the summer months (for the period to 31 August 1992) our showroom hours are 9.00am to 6.00pm Monday to Saturday. Note there will be no late Thursday opening - we will be opening late on Thursday evenings again from 1st September. Telesales hours are 9.00am to 6.00pm Monday to Friday and admin and technical support hours are 9.00am to 5.00pm Monday to Friday.

M.W.

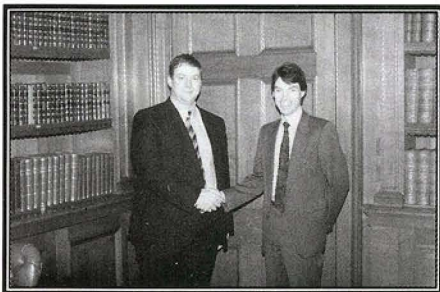
ACORN MAKES A PROFIT

Despite a slightly reduced turnover at £40.9M (down from £45.1M in 1990), Acorn has announced an improved financial position for 1991, with a second half net profit of £694,000, and a full-year profit of £274,000. The company's cash position has improved markedly, and this should be reflected in 1992's results. Acorn claims that, according to independent research, its brand share of 53% across all schools was unchanged on the previous year, and that Acorn computers are now found in 85% of all schools in the UK.

Acorn also announced its receipt of the Queen's Award for Industry, for the design and development of the ARM 32-bit RISC processor.

For more information contact Acorn Computers Ltd., Fulbourn Road, Cambridge CB1 4JN, tel. (0223) 245200.

COMPUTER CONCEPTS HAS THE VISION



Readers may be interested to learn that well known software house, Computer Concepts, has announced the acquisition of a majority stake in Newcastle-based hardware developers Wild Vision. Wild Vision design and supply video digitiser boards for the Archimedes, and also manufacture boards for Computer Concepts (for its Laser Direct printer for example). The two companies will continue to operate separately, but will be joining forces in the future on selected ventures.

Computer Concepts is at Gaddesden Place, Hemel Hempstead, Herts HP2 6EX, tel. (0442) 63933; Wild Vision is at 15 Witney Way, Boldon Colliery, Tyne and Wear NE35 9PE, tel. 091-519 1455.

LONGMAN LOGOTRON BBC CATALOGUE

Longman Logotron has released a new BBC Software for Schools catalogue. This covers software developed for schools jointly by BBC Educational Publishing and Longman Logotron. This continues to feature software for the BBC micro, including old favourites like Through the Dragon's Eye and Geordie Racer. A new title for the Look and Read Series, SkyHunter, is now available in both BBC and Archimedes formats. All in all, some eleven of the twenty-one titles listed are available for the BBC micro, and some only for that system.

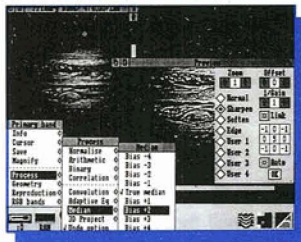
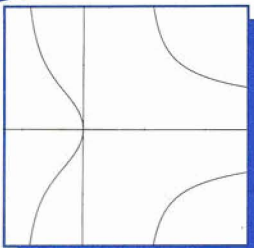
Copies of the BBC Software for Schools brochure are available from BBC Educational Information, Room A3155, Woodlands, 80 Wood Lane, London W12 0TT, tel. 081-746 1111. Details of Audio and Video resources for schools are available from the same source.

SCOTLAND CONTINUES SUPPORT FOR BBC MICRO

The Scottish Council for Educational Technology (SCET) is a research and development agency which focuses on education and training in Scotland. It also generates income from the sale of services including a range of software for the BBC micro (see also last month's BEEBUG Education). SCET has just released a new catalogue detailing the resources available, including software, videos and publications. The majority of the software is suitable for the BBC micro (over 70 different titles), and new items are still under development.

The catalogue is available from SCET, Downahill, 74 Victoria Crescent Road, Glasgow G12 9JN, tel. 041-334 9314, fax 041-334 6519.

RISC USER



The Archimedes Magazine & Support Group

RISC User continues to enjoy the largest circulation of any subscription magazine devoted solely to the Acorn Archimedes range of computers. Its in-depth, authoritative approach appeals to all users, whatever their interest and level of expertise, and the lively mixture of articles, programs, reviews and news is carefully selected to cater for all Archimedes owners from beginner to expert.

Existing BEEBUG members, who want to find out more about the Archimedes range, may either transfer their existing subscription to RISC User (at no extra charge), or extend their subscription to include both magazines. A joint subscription will enable you to keep completely up-to-date with all innovations, and the latest information from Acorn and other developers for both the BBC micro and the Archimedes range. RISC User is the magazine for enthusiasts and professionals at all levels.

The Archimedes Magazine & Support Group

Here are some articles and series published in the most recent issues of RISC User:

OPTICAL CHARACTER RECOGNITION
A preview of an exciting new software development from Irlam Instruments for converting scanned images of text into ASCII files.

PDSVIEW IMAGE PROCESSOR
A review of software from Spacetech Ltd for processing a wide range of satellite images supplied on CD-ROM.

CARTESIAN/POLAR GRAPH PLOTTER
Two applications for the graphical display of mathematical functions using either Cartesian co-ordinates or polar co-ordinates.

IMPROVING SCANNER IMAGES
An application for improving the quality of grey tones used in displaying and printing handheld scanner images.

SELECT AND COLLECT: PD SOFTWARE
A survey of some of the best Public Domain software currently available (and now included in the RISC User PD Library).

WIMP FUNCTION/PROCEDURE LIBRARY
A collection of functions and procedures for your own Wimp programs. all the necessary building blocks for your own Wimp programs.

ARMED AND READY FOR BATTLE
A fascinating investigation of the latest developments in RISC technology and the ARM processor.

CREATING DRAW FILES IN BASIC
A two-part article on how to use Basic to create Draw files.

WRITE-BACK
New readers' section of RISC User for comment, information, help - a magazine version of a bulletin board.

USING ANSI C
A series of articles on programming Desktop applications in C.

WP/DTP
A regular column on using different DTP and WP packages.

INTO THE ARC
A regular series for beginners currently treating the subject of using shared resources.

SUBSCRIPTION DETAILS

As a member of BEEBUG you may extend your subscription to include RISC User for only:

Destination	Additional Cost
UK, BFPO & Ch Is	£ 10.50
Rest of Europe and Eire	£ 15.40
Middle East	£ 19.60
Americas and Africa	£ 21.90
Elsewhere	£ 33.00

Weather Station (Part 1)

Feeling under pressure? Nick Case helps you keep track of it.

Have you ever wondered what the weather was like this time last year, as you tap the barometer in the hall on your way out of the house? Perhaps you wished you had one of those barograph machines which records the pressure on a paper strip, so that you could file it away for future reference.

With this program you can use your Beeb to display the graph and jump from month to month or year to year, then dump your choice to a printer, if required.

Type in the program and save it as BAROM. If it is all correct then when you run it you should get the display for my home town for January 1988. "Not much use to me", I hear you say. Do not despair, just start noting down those barometer readings every day and soon you will have your first month's worth. Edit them into the data lines (2400 to 2430) of this program, save the new program as <month>'<year> e.g. MAR'92 and away you go.

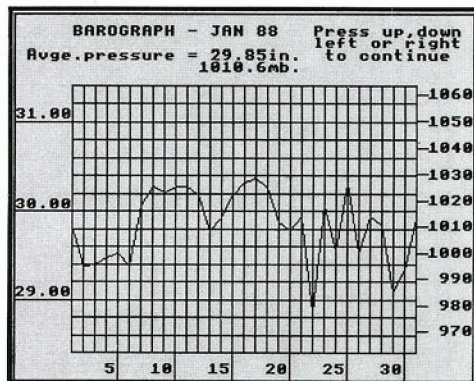
If you want quicker results, then look in back numbers of your local newspaper, as many have weather data in them; I know that mine does. Alternatively check your local library.

To continue from there requires the appropriate month and year to be present on the disc with the name and year in the file name as above. The cursor keys are set up to load other the months: Down - same month last year, Up - same month next year, Left - last month, Right - next month. Until you have built up the necessary data put a REM at the start of lines 190 to 220 as required to stop the display locking up.

PROGRAM NOTES

PROCinit and PROCtitle set up as expected, followed by PROCgr1, gr2, gr3 which draw and number the grid.

PROCreadplot reads the data and calls PROCplot to draw the graph. PROCsum will then print the month's average.



Barometric pressure in Winchester

Finally, if you have problems, put a REM at the start of line 170 to prevent VDU 21 from stopping text output to the screen.

The line numbering employed is to enable this program to be used (with a few additional lines and changes) for a maximum and minimum temperature graph and that's just what we'll be looking at next month.

```
10 REM Program BAROM
20 REM Version B 2.0
30 REM Author N.J. Case
40 REM BEEBUG June 1992
50 REM Program subject to copyright
60 :
100 ON ERROR MODE7:PROCerr
110 *FX144,0,1
120 MODE1:PROCinit
130 PROCtitle
140 PROCgr1:PROCgr2:PROCgr3
150 PROCreadplot
160 PROCsum
170 VDU21
180 REPEAT UNTIL INKEY(-42)OR INKEY(-5)
```

Weather Station

```
8)OR INKEY(-122)OR INKEY(-26)OR INKEY(-5
1):*FX15,1
190 IF INKEY(-42) PROClastyr
200 IF INKEY(-58) PROCnextyr
210 IF INKEY(-122) PROCnextmth
220 IF INKEY(-26) PROClastmth
230 IF INKEY(-51) VDU6:PROCdump
240 END
250 :
1000 DEF PROCinit
1010 DIM X(31),Y(31)
1020 VDU6,16,23;8202;0;0;0;
1030 VDU19,0,7;0;19,2,4;0;19,3,0;0;
1040 hght%=50:wdth%=32
1050 M$="JANFEBMARAPRMAJUNJULAUGSEPOCT
NOVDEC"
1060 ENDPROC
1070 :
1100 DEF PROCtitle
1110 COLOUR2:PRINTTAB(5,1)"BAROGRAPH"
1120 COLOUR3:PRINTTAB(26,1)"Press up,do
wn":PRINTTAB(26,2)"left or right":PRINTT
AB(27,3)"to continue"
1130 RESTORE:READ month$,yr%
1140 COLOUR1:PRINTTAB(15,1)"- ";month$;
" ";yr%
1150 ENDPROC
1160 :
1200 DEF PROCgr1
1210 FOR yr%=0 TO 15:MOVE 160,80+(yr%*hgh
t%):DRAW 160+(30*wdth%),80+(yr%*hght%):NE
XT
1220 FOR x%=0 TO 30:MOVE 160+(x%*wdth%)
,80:DRAW 160+(x%*wdth%),80+(15*hght%):NE
XT
1230 ENDPROC
1240 :
1300 DEF PROCgr2
1310 FOR yr%=3 TO 13 STEP5:MOVE 0,80+(yr%
*hght%):DRAW 160,80+(yr%*hght%):NEXT
1320 FOR x%=4 TO 30 STEP5:MOVE 160+(x%*
wdth%),0:DRAW 160+(x%*wdth%),80:NEXT
1330 ENDPROC
1340 :
1400 DEF PROCgr3
1410 VDU5:FOR x%=1 TO 30 STEP5:MOVE (x%
*wdth%)-65,50:PRINT x%+4:NEXT
1420 @%=&20202:FOR yr%=0 TO 10 STEP5:MOV
E 0,80+((yr%+3)*hght%)+35:PRINT (yr%/5)+29
,:NEXT:@%=&90A
```

```
1430 FOR mb%=970 TO 1060 STEP10:MOVE 16
0+(30*wdth%),80+(((mb%/33.86)-28.40)*250
):DRAW 160+(31*wdth%),80+(((mb%/33.86)-2
8.40)*250)
1440 MOVE 160+(25*wdth%),95+(((mb%/33.8
6)-28.40)*250):PRINT mb%:;NEXT
1450 ENDPROC
1460 :
1500 DEF PROCreadplot
1510 RESTORE 2410
1520 READ end:N=0
1530 REPEAT:N=N+1:READ X(N),Y(N)
1540 UNTIL N=end
1550 MOVE 160+((X(1)-1)*wdth%),80+((Y(1
)-28.4)*250):FOR N=2 TO end:GCOLOR,1
1560 PROCplot(X(N),Y(N)):NEXT
1570 ENDPROC
1580 :
1600 DEF PROCplot(X(N),Y(N))
1610 VDU5:PLOT 5,160+((X(N)-1)*wdth%),8
0+((Y(N)-28.4)*250):VDU4
1620 ENDPROC
1630 :
1700 DEF PROCsum
1710 @%=&20202:totalpress=0:FOR N=1 TO
end:totalpress=totalpress+(Y(N)):NEXT
1720 meanpress=totalpress/end:PRINTTAB(
1,3)"Avge.pressure = ";meanpress;:PRINT"
in.":@%=&20101:PRINTTAB(16,4)meanpress*3
3.86;"mb.":@%=&90A
1730 ENDPROC
1740 :
1800 DEF PROClastyr
1810 yr%=yr%-1:lastyr$=month$+" "+STR$(
yr%):*K.9 CH.lastyr$|M
1820 *FX138,0,137
1830 ENDPROC
1840 :
1900 DEF PROCnextyr
1910 yr%=yr%+1:nextyr$=month$+" "+STR$(
yr%):*K.9 CH.nextyr$|M
1920 *FX138,0,137
1930 ENDPROC
1940 :
2000 DEF PROCnextmth
2010 IF month$="DEC"THEN newmonth$="JAN
":yr%=yr%+1:GOTO 2050
2020 month%=INSTR(M$,month$)
2030 month%=month%+3
```

continued on page 37

Input ROM



Ian Palmer adds some useful features to OSWORD 0.

The Input ROM described here supplies an alternative version of the OSWORD 0 input routine. This OSWORD call is used by most programs, and languages to input text from the keyboard, but the standard routine lacks some useful features.

This new routine takes the place of the operating system routine, allowing full cursor control over your input, access to the extended character set and an extended Ctrl-U function.

The program will create a ROM image to place in sideways RAM, or blow into an EPROM. Due to its small size (less than 1K) it is only worth blowing it into an EPROM if you add further routines.

First you need to type the program in, save it and then run it. The layout used in the listing is not important, I have used this particular layout because it is easy to follow. It includes a simple check that the code is correct, and if not the program will tell you. If the program is correct further instructions are given to tell you how to install the routine.

MASTERS ONLY

It is worth mentioning at this stage that the program, as it stands, only works on the Master range of computers. The reason for this is twofold: first that the service call used (&27 on line 250) does not exist on the standard BBCs; secondly that the program uses 65C02 specific commands (for example PHY) and removing these would make the code larger and slower.

Once the routine has been set up, the first thing to notice is that pressing the cursor keys at Basic's prompt no longer move the cursor around the screen, and pressing Copy no longer produces a beep.

As usual, text is entered at the cursor position when typed at the keyboard. Pressing the left cursor key moves the cursor back through the text, and the right cursor key moves the cursor forward again. This allows you to insert text anywhere in the input text, rather like in a word processor.

The up cursor key and down cursor key allow rapid movement to the start and end of the text respectively. Delete will delete the character behind the cursor, while Copy will delete the character at the cursor.

You may, at this stage, be wondering about the case when you wish to copy text from elsewhere on the screen. The cursor keys no longer allow this, or so it would seem at first, but pressing Tab toggles between this new cursor control and the old cursor control where the cursor keys and Copy have their traditional functions.

One thing missing from the Master 128, but not the Compact, is the ability to access the extended character set via the keyboard. This new input routine allows you to input these characters by use of Ctrl-@. Pressing Ctrl-@ followed by another key produces the character with ASCII code 128 plus the value of the second key. For example pressing Ctrl-@

then Delete will produce character 255 (i.e. 128+127), and pressing Ctrl-@ Ctrl-G produces the copyright symbol (128+7).

Ctrl-U has been improved to allow you to delete quickly all text before the cursor, rather than the whole input text. To delete all the text simply press the cursor down key before Ctrl-U.

As far as I can tell the routine works under all conditions. I've tested View, Viewsheet, Interword, Spellmaster, Basic and Edit and all the places where these call OSWORD 0 work perfectly.

TECHNICAL NOTES

The ROM image produced only traps one service call, &27, to allow it to set up the vectors on a reset. This is done by *init* (lines 280 to 400).

All OSWORD calls are then trapped by the routine, and those which are not meant for call number zero are passed to the MOS's OSWORD routine. When graphic and text cursors are combined (via VDU 5) the MOS's OSWORD routine is also used, this is tested in lines 460 & 470 by OSBYTE call 117.

The main loop (lines 610 to 800) simply gets characters from OSRDCH and decides what action should take place according to the character read.

One problem which needed to be overcome was in fetching the parameter block. In most cases there is no problem, but some ROMs (for example Interword and Edit) place their parameter block in the ROM itself. This is overcome by calling *bad* (lines 1580 to 1680) to download the parameter block into main memory (locations &AA to &AE) and

then the parameter handler (lines 1410 to 1500) can carry on as normal. The ROM number of the caller is taken directly from the stack (location &108,S where S is the stack pointer.)

Line 840 sets up the accumulator, X and Y registers and the flags for exit. The Y register contains the length of the string and the carry bit of the flag register is set if the input was terminated by Escape. A problem with writing this routine was that Basic would hang up after issuing an input command, and it took much searching to find that the reason for this is that Basic relies on an undocumented feature of the in-built routine, namely that the accumulator and X register should contain zero on exit. This too is performed by line 840.

Another factor which had to be taken into account is that if you perform VDU functions in the middle of entering text by using Ctrl then any characters typed must not be placed in the input buffer. This is done by use of OSBYTE call 218 in lines 620 to 640, which returns the number of characters needed to complete the current command.

The program uses up to 17 main memory locations, 7 of which are zero page locations. The zero page locations are from &A8 to &AE which are designated as MOS scratch space, so there should be no problems there as Acorn have specified that these locations can be used.

The other ten locations (from &D92 to &D9B) are specified as reserved for a trackerball, so unless you possess a trackerball there should be no problems there either. If you do possess a trackerball then you should change the

assignments in lines 1720 to 1800 to point to convenient locations, making sure that *wvec* (line 1770) gets two memory locations. Other possible locations are from &D60 to &D7F if you don't possess Econet, or &D80 to &D91 which according to the User Guide are currently unallocated. That doesn't mean there aren't other potential problems and you will need to watch out for software which also tries to use these locations.

DIFFERENCES

The only functional differences between this new routine and Acorn's routine, other than those described above, are as follows:

ASCII character zero (which does nothing in any case) cannot be placed in an input string (but most programs/languages don't allow it anyway), nor can ASCII character 9 as this is used (via the Tab key) to toggle the cursor keys' characteristics.

Other than ASCII 1 to 31, characters typed at the keyboard which are not allowed by the input parameters will not be displayed, unlike Acorn's routine. This is because these erroneous characters would cause problems when deleting or inserting text in the middle of existing text.

This enhanced call will immediately make many applications more flexible and could also contribute to the friendliness of your own programs.

```

10 REM Program Input Rom
20 REM Version B 3.4
30 REM Author I.J.Palmer
40 REM BEEBUG June 1992
50 REM Program subject to copyright
60 :
```

```

100 buffer=&AA:word=&20C:byte=&FFF4
110 param=&A8:oswch=&FFEE:osa=&FFE3
120 osrdch=&FFE0:rdsc=&FFB9:ass=&5000
130 FOR p=4 TO 6 STEP2
140 P%=&8000:O%=ass
150 [OPTp
160 NOP:NOP:NOP
170 JMP enter
180 EQUB 130
190 EQUB &12
200 EQUB 1
210 EQUS "Input Rom":EQUB 0
220 .mess EQUS"(C) BEEBUG 1992."
230 EQUB 0:EQUB 0
240 .enter PHP:PHA:PHX:PHY
250 CMP#&27:BEQ init
260 .mexit PLY:PLX:PLA:PLP
270 RTS
280 .init STX rom:LDA word:STA wvec
290 LDA word+1:STA wvec+1
300 LDA#&12:STA word
310 LDA#&FF:STA word+1
320 LDA#168:LDX#0:LDY#255
330 JSR byte:STX param
340 STY param+1:LDY#18
350 LDA#entry MOD 256
360 STA (param),Y:INY
370 LDA#entry DIV 256
380 STA (param),Y:INY
390 LDA rom:STA (param),Y
400 BRA mexit
410 .entry PHP:CMP#0:BEQ input
420 PLP:JMP (wvec)
430 .eek LDX param:LDY param+1
440 LDA#0:JMP (wvec)
450 .input PLP:STX param:STY param+1
460 LDA#117:JSR byte:TXA
470 AND#32:BNE eek
480 JSR getpar:LDA#1:STA tb
490 LDA#4:LDY#0:LDX#1:JSR byte
500 STX oldcur
510 LDX#0:LDY#0:BRA loop
520 .ctrlUP JMP ctrlU
530 .leftP JMP left
540 .rightP JMP right
550 .upP JMP up
```

Input ROM

```
560 .downP JMP down
570 .asciiP JMP ascii
580 .alt JMP alter
590 .tabP JMP tab
600 .delP JSR delete
610 .loop JSR osrdch:BCS exit
620 PHX:PHY:PHA:LDA#218:LDX#0
630 LDY#255:JSR byte:PLA:PLY
640 CPX#0:BNE vduit:PLX
650 CMP#127:BEQ delP
660 CMP#135:BEQ copy
670 CMP#137:BEQ rightP
680 CMP#21 :BEQ ctrlUP
690 CMP#136:BEQ leftP
700 CMP#139:BEQ upP
710 CMP#138:BEQ downP
720 CMP#0 :BEQ alt
730 .altret CMP#13 :BEQ jump
740 CMP#9 :BEQ tabP
750 CMP low:BCS asciiP
760 CMP#0 :BMI loop
770 CMP#31 :BPL loop:BRA vdu
780 .vduit PLX
790 .vdu JSR osa
800 BRA loop
810 .jump JSR osa:STA (buffer),Y:CLC
820 .exit PHP:PHY:LDA#4:LDY#0
830 LDX oldcur:JSR byte
840 PLY:PLP:LDA#0:LDX#0:RTS
850 .delete CPX#0:BNE dcont:RTS
860 .dcont STY temp:TXA:TAY
870 .deloop CPY temp:BEQ dexit
880 LDA(buffer),Y:DEY
890 STA(buffer),Y
900 INY:INY:BRA deloop
910 .dexit DEX:DEY:LDA#8:JSR oswch
920 LDA#1:JSR refresh:RTS
930 .copy STY temp:TXA:TAY
940 CPY temp:BEQ nocopy
950 .cloop INY:CPY temp:BEQ cexit
960 LDA(buffer),Y:DEY
970 STA(buffer),Y
980 INY:BRA cloop
990 .cexit DEY:LDA#1:JSR refresh
1000 .nocopy JMP loop
1010 .right STY temp:CPX temp
```

```
1020 BEQ norght
1030 INX:LDA#9:JSR oswch
1040 .norght JMP loop
1050 .left CPX#0:BEQ noleft
1060 DEX:LDA#8:JSR oswch
1070 .noleft JMP loop
1080 .ctrlU CPX#0:BEQ endU
1090 JSR delete
1100 BRA ctrlU
1110 .endU JMP loop
1120 .up LDA#8
1130 .uloop CPX#0:BEQ endU:JSR oswch
1140 DEX:BRA uloop
1150 .down STY temp:LDA#9
1160 .dloop CPX temp:BEQ endU
1170 JSR oswch:INX:BRA dloop
1180 .ascii PHA:SBC low:CMP high
1190 PLA:BCS endU:CMP#13
1200 BEQ endU:CPY max:BEQ beep
1210 PHY:PHA:STX temp
1220 .aloop CPY temp:BEQ aexit
1230 DEY:LDA(buffer),Y:INY
1240 STA(buffer),Y:DEY
1250 BRA aloop
1260 .aexit PLA:STA(buffer),Y:PLY:INY
1270 LDA#0:JSR refresh:LDA#9
1280 JSR oswch:INX:JMP loop
1290 .beep LDA#7:JSR oswch:JMP loop
1300 .refresh
1310 STA rsp:STY temp:TXA:TAY
1320 .rloop CPY temp:BEQ rexit
1330 LDA(buffer),Y:JSR oswch
1340 INY:BRA rloop
1350 .rexite PHY:LDA rsp:BEQ noins
1360 LDA#32:JSR oswch:INY
1370 .noins STX temp:LDA#8
1380 .bloop CPY temp:BEQ bexit
1390 JSR oswch:DEY:BRA bloop
1400 .bexit PLY:RTS
1410 .getpar LDA param+1:CMP#&&80
1420 BMI gpar:JSR bad
1430 .gpar LDY#0:LDA(param),Y
1440 STA buffer:INY
1450 LDA(param),Y:STA buffer+1
1460 INY:LDA(param),Y:STA max
```

Continued on page 46

Days and Nights in School

Byron Blessed explains how he has been using a BEEBUG classic in his classroom.

Artificial satellites have been around for many years now, but it is only recently that we have had a chance to teach about them in the classroom in the context of Information Technology. This article gives a few ideas about how one of the BEEBUG BBC/Archimedes computer programs can be used to highlight important features about the Earth, Sun, and man-made satellites.

THE WORLD BY DAY AND NIGHT

This program was published first in BEEBUG June 1987 (Vol.6 No.2), and has now also been modified and an Archimedes version appeared on the RISC User Volume 4 Special Disc (published November 1991). It is an excellent program, which has great potential, opening up many thought provoking avenues to explore. It allows the user to specify a date and time, and then displays a map of the world, showing which areas are in darkness and which in daylight. Other display features include overlay lines depicting the Equator, the tropics, and lines of longitude. Don't worry if you don't have a copy; the BBC version is available on the BEEBUG 10th Anniversary disc.

VISUAL AIDS

As the VDU display of the Earth is two dimensional, it is of considerable help

The World by Day and Night.

Because the Earth spins once every 24 hours, different countries face the Sun at different times; this means that it can be daylight in one part of the Earth, at the same time as night in another.

Load into the computer the program called "WorldDN".
When asked, type in the date and time as it is now. The screen now shows which areas of the Earth are in daylight and darkness.

Use the computer to answer these questions:-

Where is Great Britain?	Where is the Equator?	Where is the Atlantic Ocean?
Whereabouts is today just starting?		Whereabouts is the day just ending?
Which are the lines of latitude?	Where is the Tropic of Cancer?	Which are the lines of longitude?
How many hours of darkness will we have today?		Where is the Tropic of Capricorn?
What is the name of the country where the Sun is overhead now?		How many hours of daylight will we have today?
When is the 'shortest' day?	Where was the sun overhead at the time you were born?	If you were there now, where in the sky would you have to look to see the Sun?
When is the 'longest' day?	What proportion of the Earth is covered with water?	When is Midsummers day (Equinox)?
Where will the sun be overhead on Christmas Day?	How long is the day in Australia on 25th December?	

with explanations to have a large globe available, and a torch to represent the Sun.

THE WORKSHEET

Reproduced here is a sample worksheet which has been developed for use with the program, and this can be photocopied and used by pupils to provide a written record.

The first thing to take advantage of is an opportunity to recall a little geography, simply asking pupils to point a finger at

Days and Nights in School

such things as the Equator, the tropics, Great Britain, the oceans and continents. It is also useful to reinforce the meaning and difference between lines of longitude (for east-west measurement, and for the relative time of day), and lines of latitude (for north-south measurement, seasons, and length of daylight period).

SUNRISE AND SUNSET

This section introduces the concept of the Earth's rotation every 24 hours, and the direction of spin must be established. As you look at the map, the Sun would progress to the LEFT (this can be checked by typing in two times, 1 hour apart).

Using the present date and time as an example, the computer screen will show a full 24 hour period from left to right. The band of yellow on the map represents areas in daylight, but the edge of this band also shows where the sun is rising, and where it is setting. If we're at school now, where might people just be getting up? Where might they just be getting to bed?

TIME OF DAY

Knowing the time in Great Britain, what time is it now in New York, Los Angeles, Tokyo, Jerusalem? For this question, pupils will need to know where the places are on the map, and that times change 1 hour for every 15° of longitude exist.

HOURS OF DAYLIGHT

This is an interesting topic to investigate, and the reasons for the variation should be discussed. It is worth explaining what people mean when they say that 'the days are longer

in Summer than in Winter', maybe by quoting this statement, and asking for responses. Some may find the idea of proportions difficult, but no opportunity should be lost - back to your torch and globe!

Using the present date and time again, focus attention on your home town, and you will be able to work out how many hours of daylight to expect. This is done by measuring the whole width of the picture (call this 'L'), measuring the total width of the yellow (daytime) areas across the line of latitude passing through your town (call this 'D'), and finding the value of $24 \times D/L$. This exercise can be repeated for different latitudes or cities using the same date.

THE LAND OF THE MIDNIGHT SUN

Here's a chance to bring Eskimos into the discussion! Where is this region, and why is it called this?

HOT COUNTRIES

It is quite enlightening, and amusing, to ask what it would be like at the point on the Earth where the sun 'blob' is shown on the VDU. The Sun, of course, is directly overhead, and it will probably be very, very hot! There are countless things which could be brought up at this point, but climate is the obvious one. Do children who live there know what snow is, for example?

INVESTIGATIONS

As pupils gain experience using the program, short investigations can be given. For example, where will the Sun be overhead at a certain time on

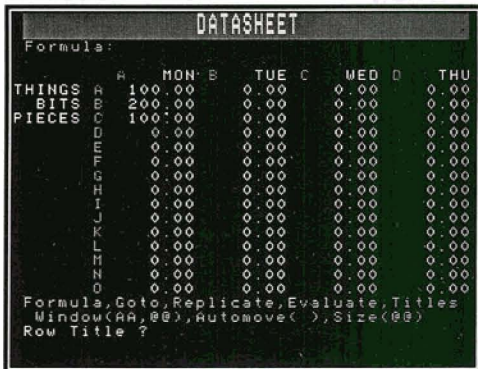
Continued on page 28

DataSheet (Part 2)

In which Stephen Colebourne completes this powerful application in Basic.

Last month the main outline of the spreadsheet was presented. This month's listing will complete the program providing valuable new functions.

Providing the line numbers were followed last month, you should be able to enter this month's program directly on top. Before entering the listing given here, load in SHEET1 from last month. Then when you type in part 2, using the line numbers given, you will add the correct lines to part 1. The completed program should be saved as DSHEET. Alternatively you could type in this month's listing, *SPOOL it, load SHEET1 then *EXEC this listing on top of it.



Formulas:

	MON	TUE	MED	THU
THINGS	100.00	0.00	0.00	0.00
BITS	200.00	0.00	0.00	0.00
PIECES	100.00	0.00	0.00	0.00
A	0.00	0.00	0.00	0.00
B	0.00	0.00	0.00	0.00
C	0.00	0.00	0.00	0.00
D	0.00	0.00	0.00	0.00
E	0.00	0.00	0.00	0.00
F	0.00	0.00	0.00	0.00
G	0.00	0.00	0.00	0.00
H	0.00	0.00	0.00	0.00
I	0.00	0.00	0.00	0.00
J	0.00	0.00	0.00	0.00
K	0.00	0.00	0.00	0.00
L	0.00	0.00	0.00	0.00
M	0.00	0.00	0.00	0.00
N	0.00	0.00	0.00	0.00
O	0.00	0.00	0.00	0.00

Formulas, Goto, Replicate, Evaluate, Titles
Window(AR, @), Automove (<), Size(@@)
Row Title ?

Adding Titles to Rows and Columns

NEW OPTIONS

All the options on the main menu are now available, including *Load* and *Save*. On entry to either option you will be asked to supply a filename. When saving a check will be made to see if the file already exists. Assuming no problems, the spreadsheet will be saved, or a new one loaded.

The remaining menu options are all related to printing, including option 5, *Spool*. The main use for a spooled spreadsheet is to load the output into a suitable word processor

where it could be formatted with text. You should note that the spooled file may contain some extra characters at the end of each line. In Wordwise these appear as pad characters and can easily be removed using *Search & Replace*.

Option 3, *Print Whole Sheet*, provides the standard format for options 4 and 5 (Windows are explained below). On pressing 3 the screen will clear and printing will immediately begin. Because a printer has a limited width the spreadsheet is broken up into as many pieces as is necessary to complete the job. Note that all the rows will be printed before going on to the next set of columns.

EDIT FUNCTION

A great many new functions have been added to the main editing screen of the spreadsheet. To remember which function keys are which, use the keystrip provided. Remember that all references to cells are made by using the two-letter codes.

REPLICATION

This is a powerful function which saves repetition in entering a number of similar formulae. Imagine you enter into cell AD the formula '(AA-AB)*AC'. Note that the formula is in the same column as the variables (numbers) in the formula. If you wanted to do the same calculation on three other sets of data, in columns B,C, and D. you might spend a long time typing in similar formulae. You could however replicate the formula in AD and get '(BA-BB)*BC' in BD, '(CA-CB)*CC' in CD, and '(DA-DB)*DC' in DD. Note that replication deals with rows in the same way.

To replicate a cell press key f1. This will prompt you for the cell to be replicated. Once entered you will be prompted for the location of the final cell the initial cell should

Data Sheet

be replicated to. In the example above it would be from cell AD to cell DD so that AD is replicated in BD, CD and DD. Replication will only take place if these two codes are in either the same row or the same column. Note that there might be a slight pause as the computer does its calculations.

FORMULAE

Last month I mentioned one extra function which could be achieved by pressing f0 followed by another key. The extra key is f1, and it allows the replication of any cell into the present location. You are prompted for the location of the cell to be replicated, again in the either same row or column. The aim of this is to provide quick access to replication if you are on the correct cell.

A second useful function has been added to the main formula entry routine. Instead of entering 'AA+AB+AC+AD', you can now enter 'S(AA,AD)'. The 'S' represents 'Sum' and the AA and AD define the inclusive range of cells to be summed. The two cells must either be in the same row or column. The whole function is automatically surrounded by brackets when it is converted, so it may be multiplied or divided directly. Ensure that the format for Sum is followed carefully, otherwise it might be rejected and removed from the formula.

WINDOWS

Windows are one of the most important means of manipulating data in the spreadsheet. A window consists of part of the sheet which is singled out from the rest. It can be any size up to the maximum sheet size. A trio of commands is provided, as well as those in the main menu, which work only on the current window.

The window is defined by pressing f4. At the prompt you must enter the top left corner cell of the new window, then the bottom right corner cell. If successful, the menu bar at the bottom of the screen should change to show *Windows(AB,GH)* or whatever you typed.

Also, you should notice that the reference letters around the screen which locate the cells are now white outside the new window (they were green before because the window was over the whole sheet).

The window can be reset to cover the whole of the sheet, its normal position, by pressing f4 twice. The first special window command allows you to clear a window. This will set every cell within the window to zero. Because of its power this option has a safety check on it. To use it press f4 followed by f3. You can use this to clear the entire spreadsheet.

The screenshot shows a spreadsheet window titled "DATASHEET". The formula bar at the top displays "Formula: AA+AB+AC". The spreadsheet grid has columns labeled A through D and rows labeled THINGS, BITS, PIECES, and a question mark. The data values are as follows:

	A	B	C	D
THINGS	100.00	30.00	23.00	12.00
BITS	200.00	20.00	34.00	34.00
PIECES	100.00	10.00	9.00	45.00
?	0.00	0.00	0.00	0.00
	400.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00

At the bottom of the screen, a menu bar is visible with the following options: "Formula, Goto, Replicate, Evaluate, Titles Window(AA,@@), Automove(), Size(@@) Your Choice ?".

Putting in a Formula

The second special command copies a window. As you would expect, this allows small windows to be copied from one part of the screen to another. The copy may take place in any direction, but must not overlap the original. To use this function press f4 followed by f5. You will be prompted for the cell where the top left hand corner of the copy will be located. If the location is not suitable, no copy will take place.

The third command is, in fact, a repeat of the *Print Window* option in the main menu. It is accessed by pressing f4 followed by f6. This will print the window using the same technique as the main print routine, i.e. all of the rows for the first few columns, then the rows for the next few columns, etc. Note that both window commands from the main

menu work using the current window, and that to spool the entire sheet you must reset the window to full size.

OTHER COMMANDS

Goto is used by pressing f2. This will transport you to whatever cell you give at the prompt provided. The evaluation of a spreadsheet can be achieved by pressing f3 or Copy. Key f9 provides an alternative method of returning to the Menu. It is also a buffer key between the keys in use and Break, which is not trapped, so be careful!

Formulas: AA+AB+AC		A	MON B	TUE C	WED D	THU
THINGS	A	100.00	30.00	23.00	12.00	
BITS	B	200.00	20.00	34.00	34.00	
PIECES	C	100.00	10.00	9.00	45.00	
?	D	400.00	60.00	66.00	91.00	
A		0.00	0.00	0.00	0.00	
B		0.00	0.00	0.00	0.00	
C		0.00	0.00	0.00	0.00	
D		0.00	0.00	0.00	0.00	
E		0.00	0.00	0.00	0.00	
F		0.00	0.00	0.00	0.00	
G		0.00	0.00	0.00	0.00	
H		0.00	0.00	0.00	0.00	
I		0.00	0.00	0.00	0.00	
J		0.00	0.00	0.00	0.00	
K		0.00	0.00	0.00	0.00	
L		0.00	0.00	0.00	0.00	
M		0.00	0.00	0.00	0.00	
N		0.00	0.00	0.00	0.00	
O		0.00	0.00	0.00	0.00	

Formula, Goto, Replicate, Evaluate, Titles
Window(AA,@@), Automove(< >), Size(@@)
Your Choice ?

The Formula Replicated Horizontally

When entering a long list of data, the command *Automove* can be useful. This works by automatically moving the cursor either across or down every time you enter a number. It works as a three way toggle. Press f5 once, and the menu bar at the bottom of the screen will show a right arrow for right-hand movement. Press f5 again and the menu bar will show a down arrow, for downward movement. Pressing f5 again will return the system to normal with no movement, as shown by no arrow after *Automove* on the menu bar.

The command *Size* allows you to limit the active size of the spreadsheet. To do this press f6 and enter the new maximum size (bottom right corner cell to use). Only the portion of the spreadsheet within this new maximum size is then active. Data outside

will not be harmed, but it will be ignored. Evaluate will work faster if it doesn't have to work through lots of zero cells. The *Save* command will also save only the active part of the sheet, thus reducing disc space taken. Remember, any data outside the active region will not be saved. Note that setting a maximum size like this will always reset the windows.

Each row and column can have a title. Row titles are set by key f7, column titles by key f8. Pressing either key allows you to enter the title for the row or column that you are currently on. The title is not usually very long, its length determined by the column width selected. The computer will beep if you attempt to enter too many letters. If accepted, the title will be immediately shown on screen. Titles are also saved on disc.

USER MODIFICATIONS

PROCUSER at the end of the program allows some customisation of the program. As mentioned before, the default column width can be altered. As well as this, the number of characters per line can be altered, and printer codes inserted. However, it is up to you to ensure that they work together. Alterations might typically allow condensed text, with perhaps 3 or 4 more columns printed.

Possible extensions to this program include new routines in the formulae similar to Sum, and an improved printer handler. Who knows what BEEBUG readers will come up with!

```

10 REM Program Datasheet (Part 2)
20 REM BEEBUG June 1992
30 :
310 IF C$="1" PROCSAVE
320 IF C$="2" PROCLoad
330 IF C$="3" PROCPRINT(1,1,MX%,MY%,1)
340 IF C$="4" PROCPRINT(WA%,WB%,WC%,WD
%,1)
350 IF C$="5" PROCPRINT(WA%,WB%,WC%,WD

```

Data Sheet

```
% , 0)
550 IF C$="R" PROCMULTIREP
560 IF C$="G" PROCGOTO
570 IF C$="W" PROCWINDOW
580 IF C$="A" PROCAUTOMOVE
590 IF C$="S" PROCMAZSIZE
600 IF C$=":" PROCTITLE(0,X%, "Column")
610 IF C$="/" PROCTITLE(1,Y%, "Row")
2000 DEF PROCNUM
2010 IF MID$(F$,P%+7,1)<>" " PROCFCD(1)
:ENDPROC
2020 IF FNPAIR(MID$(F$,P%+5,2)) PROCFCD
(8):ENDPROC
2030 PX2%=PX%:PY2%=PY%
2040 IF FNPAIR(MID$(F$,P%+2,2)) PROCFCD
(8):ENDPROC
2050 T%=(PX%>PX2%)-(PX2%>PX%)
2060 U%=(PY%>PY2%)-(PY2%>PY%)
2070 IF NOT(T%=0EORU%=0) VDU7:PROCFCD(8)
):ENDPROC
2080 E$=E$+" (D("+STR$(PX%)+", "+STR$(PY%
)+")"
2090 REPEAT:PX%=PX%+T%:PY%=PY%+U%
2100 E$=E$+"+D("+STR$(PX%)+", "+STR$(PY%
)+")"
2110 UNTIL(PX%=PX2%ANDPY%=PY2%)ORLEN(E$
)>247
2120 E$=E$+" " :P%=P%+7
2130 ENDPROC
2140 :
2150 DEF PROCSTREP
2160 IF FNPAIR(FNIN("Replicate Square",
2)) VDU7:ENDPROC
2170 IF PX%=X%EORPY%=Y% PROCREP(PX%,PY%
):ELSEVDU7
2180 ENDPROC
2190 :
2200 DEF PROCREP(RX%,RY%)
2210 F$=F$(RX%,RY%)
2220 IFRX%=X% O$=A$(RY%):N$=A$(Y%):B%=1
2230 IFRY%=Y% O$=A$(RX%):N$=A$(X%):B%=0
2240 P%=0:C%=0
2250 REPEAT:P%=P%+1:E%=FNPAIR(MID$(F$,P
%,2))
2260 IF PX$="[" C%=3
2270 IF PX$="]" C%=1
```

```
2280 IF E%=0 IFC%=0 IFMID$(F$,P%+B%,1)=
O$ F$=LEFT$(F$,P%+B%-1)+N$+MID$(F$,P%+B%
+1)
2290 C%=C%-C%MOD2
2300 UNTIL E%=1 ORP%+1=LEN(F$)
2310 ENDPROC
2320 :
2330 DEF PROCMULTIREP
2340 IF FNPAIR(FNIN("Replicate Square",
2)) VDU7:ENDPROC
2350 RA%=PX%:RB%=PY%
2360 IF FNPAIR(FNIN("From "+PX%+PY%+" t
o Square",2)) VDU7:ENDPROC
2370 RC%=PX%:RD%=PY%:V%=(RA%>RC%)-(RC%>
RA%)
2380 W%=(RB%>RD%)-(RD%>RB%)
2390 IF NOT(V%=0EORW%=0) VDU7:ENDPROC
2400 TX%=X%:TY%=Y%:X%=RA%:Y%=RB%
2410 REPEAT:X%=X%+V%:Y%=Y%+W%
2420 PROCREP(RA%,RB%):PROCFCN
2430 UNTIL X%=RC% AND Y%=RD%
2440 X%=TX%:Y%=TY%:PROCPOS
2450 ENDPROC
2460 :
2470 DEF PROCWINDOW
2480 IF FNPAIR(FNIN("Window, Top Left S
quare",2))=2 VDU7:ENDPROC
2490 IF PY$="" PROCWN2:ENDPROC
2500 W1$=PX$+PY$:WA%=PX%:WB%=PY%
2510 IF FNPAIR(FNIN("Bottom Right Squar
e",2))=0 W2$=PX$+PY$:WC%=PX%:WD%=PY%
2520 IF WA%>WC%ORWB%>WD% VDU7:PROCWNSET
2530 PROCSCREEN:PROCNUMBAR
2540 ENDPROC
2550 :
2560 DEF PROCWN2
2570 IF PX$="" ENDPROC
2580 IF PX$="E" PROCWNZERO:ENDPROC
2590 IF PX$="A" PROCWNCOPY:ENDPROC
2600 IF PX$="W" PROCWNSET:PROCSCREEN:PR
OCMENUBAR
2610 IF PX$="S" PROCPRINT(WA%,WB%,WC%,W
D%,1):PROCSCREEN:PROCNUMBAR
2620 ENDPROC
2630 :
2640 DEF PROCWNZERO
```



```

2650 IF FNIN("Clear Window, Are You Sur
e",1)<>"Y" ENDPROC
2660 FOR W%=WB% TO WD%:FOR V%=WA% TOWC%
2670 D(V%,W%)=0:F$(V%,W%)="" :E$(V%,W%)=
""
2680 NEXT, :PROCScreen
2690 ENDPROC
2700 :
2710 DEF PROCWNCOPY
2720 IF FNPAIR(FNIN("Copy Window to",2)
) VDU7:ENDPROC
2730 RX%=PX%+WC%-WA%:RY%=PY%+WD%-WB%
2740 IF RX%>MX%ORRY%>MY%ORFNWNC(PX%,PY%
) VDU7:ENDPROC
2750 IF FNWNC(RX%,PY%)ORFNWNC(PX%,RY%)O
RFNWNC(RX%,RY%) VDU7:ENDPROC
2760 FOR W%=0 TO RY%-PY%:FOR V%=0 TO RX
%-PX%
2770 D(V%+PX%,W%+PY%)=D(V%+WA%,W%+WB%)
2780 F$(V%+PX%,W%+PY%)=F$(V%+WA%,W%+WB%
)
2790 E$(V%+PX%,W%+PY%)=E$(V%+WA%,W%+WB%
)
2800 NEXT, :PROCScreen
2810 ENDPROC
2820 :
2830 DEF FNWNC(A%,B%)
2840 IF A%>=WA% AND A%<=WC% IFB%>=WB% A
ND B%<=WD% THEN=1
2850 =0
2860 :
2870 DEF PROCGOTO
2880 IF FNPAIR(FNIN("Goto Square",2)) V
DU7:ENDPROC
2890 X%=PX%:Y%=PY%:SX%=X%-1:SY%=Y%-7
2900 IF SX%>MX%-CN% SX%=MX%-CN%
2910 IF SX%<1 SX%=1
2920 IF SY%>MY%-14 SY%=MY%-14
2930 IF SY%<1 SY%=1
2940 PROCSCREEN
2950 ENDPROC
2960 :
2970 DEF PROCAUTOMOVE
2980 AM%=(AM%+1)MOD3:M%=(AM%=1):N%=(AM%
=2)
2990 PROCMENUBAR

```

```

3000 ENDPROC
3010 :
3020 DEF PROCMAxSIZE
3030 TX%=MX%:TY%=MY%:MX%=MA%:MY%=MA%
3040 A%=FNPAIR(FNIN("Bottom Right Squar
e in Use",2))
3050 MX%=TX%:MY%=TY%:IFA%ORPX%<5 ORPY%<
5 VDU7:ENDPROC
3060 MX%=PX%:MY%=PY%:MS$=PX$+PY$
3070 IF X%>MX% OR Y%>MY% X%=1:SY%=1:Y%
=1:SY%=1
3080 PROCWNCSET:PROCScreen:PROCMENUBAR
3090 ENDPROC
3100 :
3110 DEF PROCTITLE(A%,B%,C%)
3120 C%=CW%-3+A%:F$=FNIN(C$+" Title",C%
)
3130 IF F$="" ENDPROC
3140 T$(A%,B%)=RIGHT$(S$+F$,C%)
3150 PROCSCREEN
3160 ENDPROC
3170 :
3180 :
3190 DEF PROCLOAD
3200 PROCCLS:PRINTTAB(5,9)CHR$131"Load
Filename ";:INPUT"? "F$:IFF$="" ENDPROC
3210 A%=0:Z=OPENUP(F$):IFZ>0 INPUT#Z,C$
,A%
3220 IF C$<>"DS" ORA%<6 ORA%>10 CLOSE#Z
:VDU7:ENDPROC
3230 CW%=A%:INPUT#Z,MX%,MY%
3240 FOR W%=1 TO MY%:FOR V%=1 TO MX%
3250 INPUT#Z,D(V%,W%),F$(V%,W%),E$(V%,W
%)
3260 NEXT,
3270 FORV%=1TOMX%:INPUT#Z,T$(0,V%):NEXT
3280 FORW%=1TOMY%:INPUT#Z,T$(1,W%):NEXT
3290 CLOSE#Z
3300 MS$=A$(MX%)+A$(MY%):PROCCWSET:@%=A
T%
3310 ENDPROC
3320 :
3330 DEF PROCsave
3340 PROCCLS:PRINTTAB(5,9)CHR$131"Save
Filename ";:INPUT"? "F$:IFF$="" ENDPROC
3350 Z=OPENUP(F$):CLOSE#Z:IFZ>0 IFFNCON

```

```
T ENDPROC
3360 Z=OPENOUT(F$):PRINT#Z,"DS",Cw%,MX%
,MY%
3370 FOR W%=1 TO MY%:FOR V%=1 TO MX%
3380 PRINT#Z,D(V%,W%),F$(V%,W%),E$(V%,W
%)
3390 NEXT,
3400 FORV%=1TOMX%:PRINT#Z,T$(0,V%):NEXT
3410 FORW%=1TOMY%:PRINT#Z,T$(1,W%):NEXT
3420 CLOSE#Z
3430 ENDPROC
3440 :
3450 DEF FNCONT
3460 PRINTTAB(4,11)CHR$131"File Exists,
Continue ";:INPUT"? "C$
3470 IF LEFT$(C$,1)<>"Y" =1
3480 =0
3490 :
3500 DEF PROCPRINT(A%,B%,C%,D%,E%)
3510 VDU28,0,21,39,4,12
```

```
3520 IF E% VDU2:ELSEPROCSPPOOL
3530 N%=(C%-A%+1)DIVPC%-((C%-A%+1)MODPC
%>0)
3540 FOR Z%=0 TO N%-1
3550 L%=A%+Z%*PC%:M%=L%+PC%-1
3560 IF M%>C% M%=C%
3570 PROCSCR(L%,B%,M%,D%," "," ")
3580 NEXT
3590 IF E% VDU3:ELSE OSCLI("SPOOL")
3600 VDU26
3610 ENDPROC
3620 :
3630 DEF PROCSPPOOL
3640 PRINTTAB(5,5)CHR$131"Spool Filenam
e";:INPUT"? "F$
3650 OSCLI("SPOOL "+F$)
3660 ENDPROC
3670 :
3680 :
```

B

Desktop Publishing on Acorn Systems

- What are the component parts of a DTP system?
- How can I do DTP using Acorn computer systems?
- How good are they compared with Mac's and PC's?
- How much will it all cost?
- Where can I go for expert advice?

All these questions and more are answered in the booklet, "Desktop Publishing on Acorn Systems", published by Norwich Computer Services, price 75p (inc p&p).

To get one copy, *free of charge*, write to us stating "I saw your advertisement in Beebug magazine. Please send me a free copy of your DTP booklet". Alternatively, just fill in the coupon opposite and send it to...

Norwich Computer Services
96a Vauxhall Street, Norwich NR2 2SD.
Phone 0603-766592, Fax 0603-764011

Please send me a free copy of "Desktop Publishing on Acorn Systems".

Name.....

Address.....

.....

.....

BB Postcode.....

B

Fiddling with Field Lines

David Lowndes Williams presents a program that will be useful in the classroom, but will also interest other readers.

INTRODUCTION

Most people have sprinkled iron filings on a bar magnet and seen the magnetic field lines revealed. This program calculates the pattern of electric field lines surrounding point charges. These too can be revealed physically, usually by dipping electrodes (pieces of wire) into a shallow dish of oil with semolina particles sprinkled in it, and placing quite high voltages across the electrodes. This program allows you to enter a number of point charges, and you can specify the position and charge of each point; the field lines can then be displayed. To display a field line you choose a point on the screen and the field line passing through this point is drawn.

ENTERING THE PROGRAM

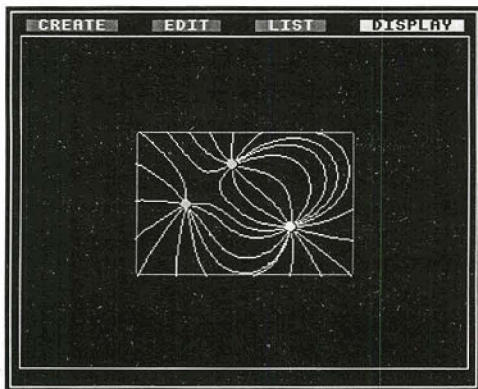
The program will work on a standard Model B as it is, but before it will run on a disc system PAGE has to be lowered. If you have a disc system on a BBC B, type in the program and save it to disc under the filename *Charges*. Then type:

```
PAGE=&1100  
CHAIN "Charges"
```

Master owners can simply CHAIN the program from disc.

LIMITATIONS

With PAGE set to &1100 the maximum number of points that can be entered is 9. Lowering PAGE will allow you to increase this number, but you may then have to sacrifice the use of some of the disc commands while using the program. If you have a tape system (or you are using a Master) you will be able to increase the maximum number of points if you wish.



USING THE PROGRAM

When you run the program, the four words CREATE, EDIT, LIST and DISPLAY are shown, and the cursor keys and Return can be used to select one of these.

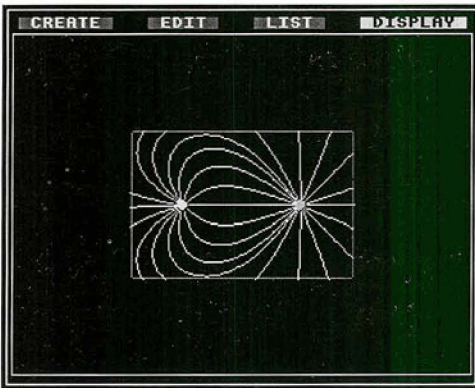
If CREATE is selected, a cross will appear which can be moved around using the cursor keys - for the moment keep it within the box at the centre of the screen. Pressing Return will select this point, and you will be asked to enter '+' or '-' to make the charge positive or negative. Entering '+' will make the point have a charge of '+1' (compared to the other charges). The magnitude of the charge can be redefined before entering CREATE by pressing Tab, upon which a window will appear and the new charge magnitude can be entered; i.e. if the magnitude is set to 4 then on pressing '+' the charge will be set to '+4'

Selecting EDIT allows you to edit points already entered. This can be done by positioning the cursor on (or very close

Fiddling with Field Lines

t) the point you wish to change. Press Return and the point will be 'picked up', and then move it to wherever you like and press Return again, before pressing '+' or '-' as with editing. The magnitude of the charge will be that previously defined (default value is 1). In this way the charge of any previously entered points can be altered.

Selecting LIST produces a list of the charges, and their positions on the screen. This can be especially useful before editing points.



DISPLAY will draw the charges as discs instead of points. You are then prompted to enter a point on a field line. Position the cursor anywhere you like (except on a charge) and press Return; a field line will then be drawn. This will be repeated until you press Space instead of Return.

Shift-Escape will allow you to quit the program. S-Escape will save the screen under the filename *screen*, allowing you to take a "snapshot" of the screen at any time. In order to recover such a screen type:

```
MODE 1
*LOAD "screen"
```

The box at the centre of the screen shows the limit to the extent of the field lines. The size of the box can be changed by pressing 'B' before entering one of the four main options. Changing the size of the box does not change the field lines, but allows them to be drawn over a larger area.

THEORY

The force acting between two stationary point charges of charge q_1 and q_2 is given by:

$$F = (q_1 * q_2) / (4 * \text{PI} * E * r^2)$$

where r is the distance separating the charges and E is the physical constant 'epsilon nought'. The force is said to be proportional to $1/r^2$. The force is directed along the line joining the two forces. Hence, if the distance and direction of each of the charges is known (from a specified point), the force due to each charge can be calculated. By doing the correct vector sum, the direction of the force can be found, and the direction of the force is parallel to the field line. This calculation (which is the basis of the entire program) is performed by *FNdirection(Cx,Cy,p%)*, where C_x and C_y are the co-ordinates of the point at which the direction of the field is calculated. V_x and V_y are the output from the function, and give the direction of the field (V_x, V_y). The function returns the length the vector (V_x, V_y).

By calculating the direction of the field at a point, drawing this vector and then repeating for the point at the end of the last vector, an approximation to the field lines is drawn.

TECHNICAL NOTE

To increase the maximum number of charges, change the value assigned to the variable *maxch%* in line 110.


```

10 REM Program Charges
20 REM Version B 1.0
30 REM Author David Lowndes Williams
40 REM BEEBUG June 1992
50 REM Program Subject to Copyright
60 :
100 ON ERROR GOTO 230
110 MODEL: maxch%=9: DIM charge(3, maxch%
+1): PROCinit
120 PROCinitscreen
130 :
140 REPEAT
150 a%=FNchoose(A%): A%=a%
160 IF a%=0 PROCcreate
170 IF a%=1 PROCedit
180 IF a%=2 PROClist
190 IF a%=3 PROCinitscreen: PROCDfield
200 UNTIL FALSE
210 END
220 :
230 IF ERR<>17 REPORT: PRINT" at line "
; ERL: END
240 IF ERR=17 AND INKEY-1 END
250 IF ERR=17 AND INKEY-82 OSCLI"SAVE
screen 3000 8000": GOTO 120
260 IF ERR=17 VDU3, 4, 7: GOTO120
270 END
280 :
1000 DEF PROCinit
1010 VDU23, 128, 0, 56, 124, 254, 254, 124
, 56
1020 blank$=STRING$(38, CHR$(32))
1030 charge(0, 0)=1: charge(0, 1)=640
1040 charge(1, 1)=512: A%=0: E%=0: out%=0
1050 X1=340: X2=940: Y1=712: Y2=312: c=1
1060 ENDPROC
1070 :
1080 DEF PROCinitscreen
1090 GCOL0, 3: COLOUR128: CLS: PROCTopLine
1100 COLOUR128: MOVE16, 976: DRAW1263, 976
1110 DRAW1263, 48: DRAW16, 48: DRAW16, 976
1120 PROCDBound: max%=charge(0, 0)
1130 PROCdispPoints
1140 ENDPROC
1150 :
1160 DEF PROCTopLine
1170 COLOUR128: VDU31, 0, 0, 32: COLOUR129: C

```

```

COLOUR3: PRINT; " CREATE "; : COLOUR128: PRINT
SPC3; : COLOUR129: PRINT" EDIT "; : COLOUR128
: PRINTSPC3; : COLOUR129: PRINT" LIST "; : COL
OURL28: PRINTSPC3; : COLOUR129: PRINT" DISPL
AY ": COLOUR128
1180 ENDPROC
1190 :
1200 DEF FNchoose(p%)
1210 *FX4, 1
1220 *FX138, 0, 32
1230 REPEAT: g%=GET
1240 IF g%=137 THEN p%=p%+1
1250 IF g%=136 THEN p%=p%-1
1260 IF g%=66 PROCmoveBoundary
1270 IF g%=9 PROCchangeCharge
1280 IF p%>3 p%=3
1290 IF p%<0 p%=0
1300 PROCDispChoice(p%): UNTIL g%=13
1310 COLOUR128: COLOUR3: *FX4, 0
1320 =p%
1330 :
1340 DEF PROCDispChoice(p%)
1350 PROCTopLine: COLOUR131: COLOUR0
1360 IF p%=0PRINTTAB(1, 0)" CREATE "
1370 IF p%=1PRINTTAB(12, 0)" EDIT "
1380 IF p%=2PRINTTAB(21, 0)" LIST "
1390 IF p%=3PRINTTAB(30, 0)" DISPLAY "
1400 COLOUR128: COLOUR3: ENDPROC
1410 :
1420 DEF PROCcreate
1430 PROCinitscreen
1440 IF charge(0, 0)>maxch% VDU7: ENDPROC
1450 IF E%=TRUE PROCDispChoice(1) ELSE
PROCDispChoice(0)
1460 PRINTTAB(0, 31)"Enter point.";
1470 a%=FNGetPoint(charge(0, charge(0, 0)
), charge(1, charge(0, 0)))
1480 IF a%=32PRINTTAB(0, 31)blank$: ENDP
ROC
1490 GCOL0, 1: PLOT69, x, y: PRINTTAB(0, 31)"
Plus or minus (+/-) ?";
1500 REPEAT: g%=GET: IF g%=ASC"+" OR g%=A
SC "-" OR g%=32 exit%=TRUE ELSE exit%=FAL
SE
1510 UNTIL exit%=TRUE
1520 PRINTTAB(0, 31)blank$;
1530 IF g%=32 THEN GCOL0, 0: PLOT69, x, y: E

```

Fiddling with Field Lines

```
NDPROC
1540 IF E%=TRUE charge(0,out%)=x:charge
(1,out%)=y
1550 IF E%=TRUE :IF g%=ASC"+"GCOL0,2:ch
arge(2,out%)=c ELSE charge(2,out%)=-c:GC
OL0,3
1560 IF E%=TRUE PLOT69,x,y:ENDPROC
1570 n=charge(0,0):charge(0,n)=x:charge
(1,n)=y
1580 charge(0,n+1)=x:charge(1,n+1)=y
1590 IF g%=ASC"+"GCOL0,2:charge(2,n)=c
ELSE charge(2,n)=-c:GCOL0,3
1600 charge(0,0)=n+1:PLOT69,x,y:ENDPROC
1610 :
1620 DEF FNGetPoint(Sx,Sy)
1630 *FX4,1
1640 GCOL3,2:PROCCursor(Sx,Sy)
1650 REPEAT
1660 g%=GET:PROCCursor(Sx,Sy):PROCifs
1670 PROCCursor(Sx,Sy)
1680 UNTIL g%<>136 AND g%<>137 AND g%<>
138 AND g%<>139
1690 PROCCursor(Sx,Sy)
1700 *FX4,0
1710 GCOL0,3:x=Sx:y=Sy
1720 =g%
1730 :
1740 DEF PROCCursor(x,y)
1750 MOVE x-16,y:DRAW x+16,y
1760 MOVE x,y-16:DRAW x,y+16
1770 ENDPROC
1780 :
1790 :
1800 DEF FNDirection(x,y,q)
1810 LOCAL Fx,Fy,Fz,loop,F,rx,ry,rz,r
1820 max%=charge(0,0):Vx=0:Vy=0:Vz=0
1830 FOR loop=1 TO max%
1840 rx=x-charge(0,loop)
1850 ry=y-charge(1,loop)
1860 r=SQR(rx*rx+ry*ry):IF r=0 =0
1870 F=q*charge(2,loop)/(r*r*r)
1880 Fx=F*rx:Fy=F*ry:Vx=Vx+Fx:Vy=Vy+Fy
1890 NEXT loop:=SQR(Vx*Vx+Vy*Vy)
1900 :
1910 DEF PROCdispVector(a,b,c,d)
1920 MOVEa,b:DRAWa+c,b+d
1930 ENDPROC
```

```
1940 :
1950 DEF PROCdBound
1960 GCOL3,1:MOVE X1,Y1:DRAWX2,Y1
1970 DRAWX2,Y2:DRAWX1,Y2:DRAWX1,Y1
1980 ENDPROC
1990 :
2000 DEF PROCmoveBoundary
2010 COLOUR128:COLOUR3
2020 PRINTTAB(0,31)*New boarder T-toggl
e,<Return>-stop.";
2030 *FX4,1
2040 *FX138,0,32
2050 GCOL3,2:Sx=X1:Sy=Y1:REPEAT:g%=GET
2060 MOVE Sx,Y2:DRAWSx,Sy:DRAWX2,Sy
2070 DRAWX2,Y2:DRAWSx,Y2:PROCifs
2080 MOVE Sx,Y2:DRAWSx,Sy:DRAWX2,Sy
2090 DRAWX2,Y2:DRAWSx,Y2
2100 UNTIL g%=ASC"T" OR g%=13
2110 X1=Sx:Y1=Sy:IF g%=13 GOTO 2200
2120 Sx=X2:Sy=Y2
2130 REPEAT:g%=GET
2140 MOVE X1,Sy:DRAWX1,Y1:DRAWSx,Y1
2150 DRAWSx,Sy:DRAWX1,Sy:PROCifs
2160 MOVE X1,Sy:DRAWX1,Y1:DRAWSx,Y1
2170 DRAWSx,Sy:DRAWX1,Sy
2180 UNTIL g%=ASC"T" OR g%=13
2190 X2=Sx:Y2=Sy:IF g%=ASC"T" GOTO 2050
2200 REM Prepare to exit procedure.
2210 IF X1>X2 THEN s=X1:X1=X2:X2=s
2220 PROCinitscreen:g%=0
2230 PRINTTAB(0,31)blank$;:ENDPROC
2240 :
2250 DEF PROCdispBlob
2260 VDU5:FOR u%=1 TO charge(0,0)-1
2270 IF charge(2,u%)<0 GCOL0,3 ELSE GCO
L0,2
2280 MOVE charge(0,u%)-16,charge(1,u%)+
18:VDU128:NEXT:VDU4
2290 ENDPROC
2300 :
2310 DEF PROCdispPoints
2320 FOR u%=1 TO charge(0,0)-1
2330 IF charge(2,u%)<0 GCOL0,3 ELSE GCO
L0,2
2340 IF charge(2,u%)=0 GCOL0,0
2350 PLOT69,charge(0,u%),charge(1,u%)
2360 NEXT
```



```

2370 ENDPROC
2380 :
2390 DEF PROCdfield
2400 G%=charge(0,charge(0,0))
2410 H%=charge(1,charge(0,0))
2420 IF charge(0,0)=1 VDU7:ENDPROC
2430 PROCDispChoice(3):PROCdispBlob
2440 REPEAT:PRINTTAB(0,31)"Enter point
on field line.":a%=FNGetPoint(G%,H%)
2450 IF a%=32 GOTO 2550
2460 PRINTTAB(0,31)blank$;:G%=x:H%=y
2470 FOR p%=-1 TO 1 STEP 2
2480 Cx=x:Cy=y:MOVE Cx,Cy:out%=FALSE
2490 MOVE Cx,Cy:out%=FALSE:REPEAT
2500 a=FNdirection(Cx,Cy,p%)
2510 IF a<0 Vx=5*Vx/a:Vy=5*Vy/a
2520 Cx=Cx+Vx:Cy=Cy+Vy:DRAW Cx,Cy
2530 UNTIL FNDist(Cx,Cy)<400 OR Cx<X1 O
R Cx>X2 OR Cy>Y1 OR Cy<Y2 OR a=0
2540 NEXT p%
2550 PRINTTAB(0,31)blank$;:UNTIL a%=32
2560 PROCdispBlob:ENDPROC
2570 :
2580 DEF FNDist(Sx,Sy)
2590 D=10000000:FOR u%=1 TO charge(0,0)
2600 d=(charge(0,u%)-Sx)^2+(charge(1,u%
)-Sy)^2
2610 IF d<D THEN D=d
2620 NEXT:=D
2630 :
2640 DEF PROCifs
2650 IF g%=136 THEN Sx=Sx-4:IF INKEY-1
Sx=Sx-12
2660 IF g%=137 THEN Sx=Sx+4:IF INKEY-1
Sx=Sx+12
2670 IF g%=138 THEN Sy=Sy-4:IF INKEY-1
Sy=Sy-12
2680 IF g%=139 THEN Sy=Sy+4:IF INKEY-1
Sy=Sy+12
2690 ENDPROC
2700 ;;
2710 DEF PROClist
2720 CLS:PRINTSPC7;:COLOUR129
2730 PRINT" List of charges present: "
2740 COLOUR128
2750 IF charge(0,0)=1 PRINT'SPC10"No c

```

```

harges present!":GOTO2810
2760 PRINT'"To the printer (Y/N)?"';
2770 g%=GET:IF g%=89 OR g%=121 VDU89,2
ELSE VDU78
2780 PRINT'"SPC8"X"SPC9"Y"SPC7"Charge"
2790 FOR u%=1 TO charge(0,0)-1
2800 PRINT charge(0,u%),charge(1,u%),ch
arge(2,u%):NEXT:VDU3
2810 PRINT'"Space to continue."
2820 REPEAT UNTIL GET=32:PROCinitscreen
2830 ENDPROC
2840 :
2850 DEF PROCchangeCharge
2860 a%=3:b%=20:c%=30:d%=6
2870 VDU28,a%,b%,c%,d%
2880 COLOUR129:CLS:COLOUR128
2890 VDU28,a%+1,b%-1,c%-1,d%+1,12
2900 PRINT"Magnitude of charge :";c
2910 PRINT'"Enter new value:":INPUT d
2920 IF d<>0 c=ABS(d)
2930 VDU26:PROCinitscreen:ENDPROC
2940 :
2950 DEF PROCedit
2960 PROCinitscreen:PROCDispChoice(1)
2970 IF charge(0,0)=1 VDU7:ENDPROC
2980 PRINTTAB(0,31)"Position cursor ove
r desired charge.";
2990 a%=FNGetPoint(charge(0,charge(0,0)
),charge(1,charge(0,0)))
3000 IF a%=32PRINTTAB(0,31)blank$;:ENDP
ROC
3010 u%=1:REPEAT
3020 a%=charge(0,u%):b%=charge(1,u%)
3030 r%=(a%-x)^2+(b%-y)^2
3040 IF r%<300 out%=u% ELSE out%=0
3050 u%=u%+1
3060 UNTIL out%<>0 OR u%=charge(0,0)
3070 PRINTTAB(0,31)blank$;
3080 IF out%=0 VDU7:ENDPROC
3090 charge(0,charge(0,0))=a%
3100 charge(1,charge(0,0))=b%:E%=TRUE
3110 PROCcreate:E%=FALSE
3120 charge(0,charge(0,0))=x
3130 charge(1,charge(0,0))=y
3140 PROCinitscreen:ENDPROC

```

Mr Toad's Machine Code Corner 3

More croakings from our resident amphibian.

It is a well-known fact that toads are very cultured pond dwellers, so MCC this month deals with the production of Great Classical Music using just a few bytes of simple machine code. The Toad Rom 90 makes Mr. T's Beeb play *Land Of Hope And Glory* on power-up, and Mrs. T's plays the Archers' signature tune. Only an EPROM can do the trick on power-up, or possibly a *!boot* file on a Master of course, but you can have fun with tunes in other ways.

The first thing to decide is how to produce the sound. On the face of it, the only way seems to be via OSWORD 7, but a glance at pages D3-10 to D3-14 of the Reference Manual shows that this is far too complex for a mere toad - there has to be an easier way and this is it: use VDU 7 - LDA #7:JSR &FFEE. OSBYTES &D3 to &D6 will change the channel, volume, pitch and duration, in that order, or we can just poke locations &263 - &266 with the values, which is all that the OSBYTES do.

Unlike other machines of the same generation, the Beeb has a proper sound chip which can hold five sets of sound parameters in each of three channels (plus the 'noise' channel, which is no use for playing tunes), so our routine will be able to send off the details of up to fifteen musical notes to the sound chip - which takes a few milliseconds - then the 6502 will be free to get on with other work while the sound chip does its job. If we need more than 15 notes we must either hold everything up while we wait for spaces in the sound queues to become free, or the following code must be timed with great accuracy to the point

when it must hand over to the sound routine again.

In fact, we shall see that at least two of these notes must be unheard, but 13 notes can do more than you might think. The data in the listing will play the first two lines of the above-mentioned tunes - just set the label in line 260 for the tune you want; *Elgar* or *Archers*. Incidentally, all limitations mentioned here apply equally to OSWORD 7. Mr. T knows that the books say 6 notes per channel, not 5, but you try it!

HERE'S ONE I PREPARED EARLIER

What we do is this: set up a neat line of four data values for each note, in the order mentioned above. In fact we only need to poke any given location when the parameter is to be changed, but the code is vastly simplified if we always read off and poke in all four parameters in a simple loop, then after each set of parameters we call OSWRCH with A=7.

CHANNEL

Channel is 1, 2 or 3 - easy. After channel 1 is full, we switch to channel 2. Delete lines 380 and 430 of *Elgar* and try it. Chaos! The code sets all 3 channels up practically instantaneously, and they all play at once. The answer is to add a silent 'filler' - line 380, the first 'note' on channel 2, is silent (volume 248) and lasts as long as all the notes on channel 1 combined, so channel 2 doesn't play an audible note until channel 1 has finished. Line 430 does the same on channel 3 when 2 is full. That one, of course, has to last as long as all the notes so far from the start.

To avoid a slight clicking noise, keep the pitch of the 'filler' note the same as the one before or after. There's a drawback here - we now have only four audible notes on channels 2 and 3, and no you can't use channel 0 for these 'fillers', it's never quite silent whatever value you use.

VOLUME

Volume should be a multiple of eight, 128 being the loudest and 240 the softest; 248 is silent. Really we should do it in hex, but the pitch values are a bit easier in decimal, so just for once Mr. Toad has descended to that vulgar level. If you use any value not divisible by eight (i.e. any of bits 0, 1 or 2 set) you get silence, and values below 128 refer to an envelope. In the listing, all notes but 'fillers' have the same volume, but you can have fun tinkering with this parameter.

PITCH

Now let's deal with pitch - middle C is value 52, then go up or down 4 per semitone from there. Mr. Toad writes in washable felt-pen on the tadpoles' piano keys, remembering to add 8, not 4, on the next white key if a black key comes between. He then plays the tune laboriously by ear, jotting down numbers now and then while the tadpoles chortle with glee. Another problem: if two consecutive notes have the same pitch, as with the first two in 'Land Of...', they play as one long one. Line 340 interposes a note of zero length and volume, which works as a separator. We really need another at 'MOTH - ER of the free', but we'd run out of notes. The omission doesn't seem so bad in the middle of the tune, but try deleting line 340 to see what you think.

DURATION

For duration, use the figures in the listing as a starting point, then do it by trial and error. If a crotchet is 7, then a minim should be 14 and a semibreve 28, but you can fiddle around with these values.

Finally we need an end marker byte. We could count the loops, but Mr. T is too lazy. &FF/255 is harmless when poked into the channel variable, so we'll use that.

This month's competition: write the parameters for a scale of C major, beginning with pitches 52,60,.... The first incorrect solution opened wins all Granny Toad's old Eddy Grundy L.P.'s. Next month, proof that Stonehenge is really a massive 6502 assembler program left by small green creatures from the planet Tharg.

```
10 REM Program Tune
20 REM Version B 1.0
30 REM Author Mr. Toad
40 REM BEEBUG June 1992
50 REM Program subject to copyright
60 :
70 FOR n%=0 TO 2 STEP 2
100 P%=&DD00:REMtransient utility area
110 [ OPT n%
120 LDA #&0F
130 LDX #0
140 JSR &FFF4 \ flush buffers
150 LDY #&FF
160 LDX #&FF
170 .loop
180 INY
190 INX
200 CPX #4 \ have we poked in 4 yet?
210 BNE next
220 LDA #7 \ if so, call 'BEEP'
230 JSR &FFEE
```

Mr Toad's Machine Code Corner 3

```
240 LDX #0 \ and set up for &263 again
250 .next
260 LDA Elgar,Y \ or 'Archers'
270 STA &263,X \ poke in parameter
280 CMP #&FF \ end-marker
290 BNE loop
300 RTS
310 :
320 .Elgar
330 EQU 1:EQU 176:EQU 104:EQU 14
340 EQU 1:EQU 248:EQU 104:EQU 0 \
separator
350 EQU 1:EQU 176:EQU 104:EQU 14
360 EQU 1:EQU 176:EQU 100:EQU 7
370 EQU 1:EQU 176:EQU 104:EQU 7
380 EQU 2:EQU 248:EQU 104:EQU 42 \
filler
390 EQU 2:EQU 176:EQU 112:EQU 14
400 EQU 2:EQU 176:EQU 92:EQU 23
410 EQU 2:EQU 176:EQU 84:EQU 23
420 EQU 2:EQU 176:EQU 76:EQU 26
430 EQU 3:EQU 248:EQU 76:EQU 128 \
filler
440 EQU 3:EQU 176:EQU 72:EQU 7
450 EQU 3:EQU 176:EQU 76:EQU 7
```

```
460 EQU 3:EQU 176:EQU 84:EQU 14
470 EQU 3:EQU 176:EQU 64:EQU 23
480 EQU 255
490 :
500 .Archers
510 EQU 1:EQU 160:EQU 92:EQU 8
520 EQU 1:EQU 160:EQU 80:EQU 4
530 EQU 1:EQU 160:EQU 112:EQU 7
540 EQU 1:EQU 160:EQU 100:EQU 4
550 EQU 1:EQU 160:EQU 92:EQU 7
560 EQU 2:EQU 248:EQU 80:EQU 30 \
filler
570 EQU 2:EQU 160:EQU 80:EQU 4
580 EQU 2:EQU 160:EQU 64:EQU 12
590 EQU 2:EQU 160:EQU 92:EQU 7
600 EQU 2:EQU 160:EQU 80:EQU 4
610 EQU 3:EQU 248:EQU 80:EQU 57 \
filler
620 EQU 3:EQU 160:EQU 112:EQU 7
630 EQU 3:EQU 160:EQU 108:EQU 4
640 EQU 3:EQU 160:EQU 100:EQU 12
650 EQU 3:EQU 160:EQU 92:EQU 12
660 EQU 255
670 J:NEXT
680 CALL &DD00
```

B

Days and Nights in School (continued from page 14)

Christmas Day? How long is the daylight period in Australia on Christmas Day, compared with Great Britain? Is it cold there on Christmas Day?

A slightly more advanced question would be to set the problem of finding the date of the Equinox, when we have an equal 12 hours of day and night. This would involve the need to try several dates, gradually homing in on the correct one.

PRINTING OUT MAPS

Children are more likely to remember things if they can relate to them, so they will be happy to type in their own birth date and time; this will show the state of

play when they were born, and it would be rewarding for them to try printing this out. On the BBC computer a suitable graphics dump program will be needed, while on the Archimedes *ScreenSave can be used, transferring the file to Draw or Paint to view or print.

This short topic could be the first in a series on 'The Earth, Sun, and Satellites' which would form a fascinating module in any IT course as well as tackling plenty of maths and science attainment targets. The second topic could begin the study of Earth satellites proper, asking pupils to work out where they would place satellites to perform particular tasks.

B

Replicating Polygonal Patterns

Mike Williams explores a striking class of mathematically based patterns.

Many users find a fascination in the many patterns which can be generated by computer, and in this month's Workshop we look at a particular class which offers plenty of scope for experimentation. What is often interesting about such pattern generation is the way in which a succession of points appears to define further curves or lines.

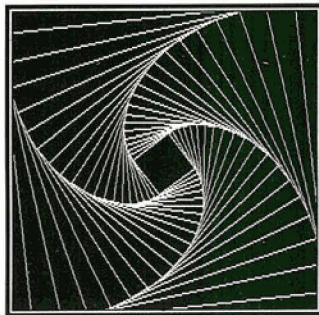


Figure 1

To see the kind of object which we are talking about, type in and then run the program given in listing 1. If you have copied this correctly you should find that it produces the pattern shown in figure 1. This looks quite interesting, and you should see what I meant earlier by the example of the four apparent curves spiralling in to the centre from the four corners.

In fact, the pattern consists of a succession of squares, each smaller than the previous one and rotated through a constant angle. The apparent curves are the loci (the paths traced out) of the corners of the reducing squares. In general terms, the four corners of a square can be represented as the points:

$$(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)$$

Assuming that the addition of indices is modulo 4 arithmetic (e.g. x_{i+1} with $i=4$ becomes $x_5=x_1$), the point on the line joining (x_i, y_i) to (x_{i+1}, y_{i+1}) is given by:

$$((1-\mu)x_i + \mu x_{i+1}, (1-\mu)y_i + \mu y_{i+1})$$

where:

$$0 \leq \mu \leq 1$$

In effect $\mu:(1-\mu)$ is the ratio in which the side is divided. If μ is fixed then the co-ordinates of the four corners of a new inner square can be calculated as above from the co-ordinates of the previous square, and the sides of the new square make an angle $\alpha = \tan^{-1}[\mu/(1-\mu)]$ with the corresponding sides of an outer square. Thus by keeping μ fixed, the angle between consecutive squares remains constant.

Now let us look at the program itself in a little more detail. We are only working in black and white, so mode 0 will give the best resolution, though mode 1 will also produce quite good results. The origin is also moved to the centre of the graphics screen.

The initial co-ordinates of the four corners of the square are given in data statements at the end of the program, and are read into the two arrays $X()$ and $Y()$. Line 140 sets 0.1 as a value for μ (written $S\mu$), and then 0.9 as the value of $(1-\mu)$ (written $R\mu$). The program draws a total of 21 squares using the FOR-NEXT loop starting at line 150 and ending at line 270.

BEEBUG Workshop - Replicating Polygonal Patterns

At any iteration of the loop, the arrays X() and Y() hold the co-ordinates of the corners of the new square, which is drawn by line 180, and two further arrays XD() and YD() are used to hold the newly calculated co-ordinates for the next square as these are progressively generated within a further FOR-NEXT loop from lines 170 to 220. Once a square has been drawn, and new co-ordinates calculated, these are copied into the arrays X() and Y() to become the current set of co-ordinates (lines 230 to 260). You might like to experiment with different values for μ , and the number of squares drawn (the loop terminator at line 150) to see the effects which can result.

If you look more carefully at the DATA statements at the end of the program (lines 500, 510), you will see that the corners are in the order:

(-400,400), (-400,-400), (400,-400), (400,400)

that is anticlockwise starting at top left. If you change the two DATA statements to read:

```
500 DATA -400, 400, 400,-400
```

```
510 DATA 400, 400,-400,-400
```

you will see, on re-running the program that the squares now rotate in a clockwise direction.

There is one further refinement which we can introduce. If you look at the orientation of the innermost square of the resulting pattern you will see that this at no recognisable angle. As already noted, the angle of rotation between one square and the next is given by $\tan^{-1}[\mu/(1-\mu)]$. After $n+1$ squares have been drawn, the innermost square will be at an angle $n \tan^{-1}[\mu/(1-\mu)]$ relative to the outermost square. It would be more pleasing if this angle was a multiple of $\pi/4$ (say). This can be achieved by adjusting the value of μ . For example, to achieve a rotation of $3\pi/4$ with $n=20$ (for a total of 21 squares):

$$n \tan^{-1}[\mu/(1-\mu)] = 3\pi/4$$

and hence:

$$\mu = (\tan(3\pi/4n)) / (\tan(3\pi/4n) + 1)$$

With $n=20$, this gives a value of 0.10583. Substitute this as the value for μ in the program, and you should now find that the innermost square is nicely aligned relative to the original.

To pursue this class of pattern further, I have created a revised version of listing 1, and this program, called *Squares2*, is presented as listing 2. This creates a grid of squares, but by alternating left and right rotations a more striking result can be obtained (see figure 2).

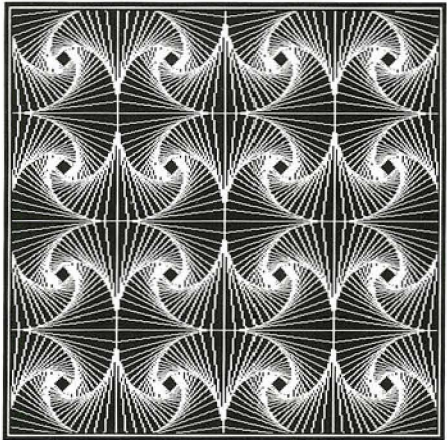


Figure 2

To allow for more user choice, the program has been modularised using two procedures, *PROCpatterns* and *PROCsquare*. The parameters of the first of these allow the user to specify the x,y screen co-ordinates for the bottom left-hand corner for the complete display (in effect an offset from 0,0). The parameter S indicates the size for each individual square (the resulting length of side being 2^*S). R should be set either to 1 (indicating that the rotation of the first square is clockwise) or to 0 (indicating that its rotation is anticlockwise). The data for both orders are contained nominally in DATA statements from line 2000. These nominal co-ordinates are multiplied by the value of S to get the true co-ordinates. From the value of R an

BEEBUG Workshop - Replicating Polygonal Patterns

offset D is calculated for use in the RESTORE statement at line 1110 which selects the appropriate set of nominal coordinates. The last parameter in *PROCpattern* is the number of squares up and across to form the grid (in the example listing this is selected as 4).

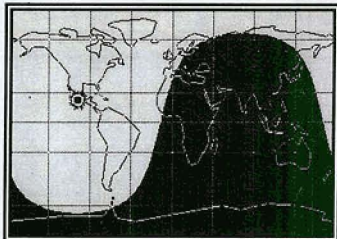
The second procedure, *PROCsquare*, is just a rewrite of our original program. The first two parameters, as above, are an offset for the whole display. The next two, A and B , are determined by the FOR-NEXT loops in *PROCpattern*, and are indicators of the position of an individual square in the grid. The value of S is passed through to this procedure, and the calculated offset D for use in the RESTORE statement. I have also used the recalculated value for μ .

Well, that's all there is to it. I hope you have fun experimenting with this pattern, and if you are really feeling ambitious, why not try writing similar programs but based on a triangle or hexagon. As here, I suggest you work on a single polygon before attempting to experiment with multiple displays.

```
10 REM Program Squares1
20 REM Version B 1.0
30 REM Author Mike Williams
40 REM BEEBUG June 1992
50 REM Program Subject to Copyright
60 :
100 MODE0:VDU29,640,512;
110 DIM X(4),Y(4),XD(4),YD(4)
120 FOR I=1 TO 4:READ X(I):NEXT
130 FOR I=1 TO 4:READ Y(I):NEXT
140 Smu=0.1:Rmu=1-Smu
150 FOR I=1 TO 21
160 MOVE X(4),Y(4)
170 FOR J=1 TO 4
180 DRAW X(J),Y(J)
190 NJ=J MOD4+1
200 XD(J)=Rmu*X(J)+Smu*X(NJ)
210 YD(J)=Rmu*Y(J)+Smu*Y(NJ)
220 NEXT J
230 FOR J=1 TO 4
240 X(J)=XD(J)
250 Y(J)=YD(J)
260 NEXT J
```

```
270 NEXT I
280 END
290 :
500 DATA-400,-400,400,400
510 DATA 400,-400,-400,400
```

```
10 REM Program Squares2
20 REM Version B 1.0
30 REM Author Mike Williams
40 REM BEEBUG June 1992
50 REM Program Subject to Copyright
60 :
100 MODE0
110 DIM X(4),Y(4),XD(4),YD(4)
120 Smu=0.10583:Rmu=1-Smu
130 PROCpattern(120,32,120,1,4)
140 END
150 :
1000 DEF PROCpattern(X,Y,S,R,N)
1010 D=20*R
1020 FOR A=1 TO 2*N STEP 2
1030 D=20-D
1040 FOR B=1 TO 2*N STEP 2
1050 PROCsquare(X,Y,A,B,S,D)
1060 D=20-D
1070 NEXT B,A
1080 ENDPROC
1090 :
1100 DEF PROCsquare(X,Y,A,B,S,D)
1110 RESTORE D+2000
1120 FOR I=1 TO 4:READ X(I):NEXT
1130 FOR I=1 TO 4:READ Y(I):NEXT
1140 VDU29,X+S*A,Y+S*B;
1150 FOR I=1 TO 21
1160 MOVE S*X(4),S*Y(4)
1170 FOR J=1 TO 4
1180 DRAW S*X(J),S*Y(J)
1190 NJ=J MOD4+1
1200 XD(J)=Rmu*X(J)+Smu*X(NJ)
1210 YD(J)=Rmu*Y(J)+Smu*Y(NJ)
1220 NEXT J
1230 FOR J=1 TO 4
1240 X(J)=XD(J)
1250 Y(J)=YD(J)
1260 NEXT J,I
1270 ENDPROC
1280 :
2000 DATA -1,-1,1,1
2010 DATA 1,-1,-1,1
2020 DATA -1,1,1,-1
2030 DATA 1,1,-1,-1
```



- PERSONALISED ADDRESS BOOK** - on-screen address and phone book
- THE EARTH FROM SPACE** - draw a picture of the Earth as seen from any point in space
- PAGE DESIGNER** - a page-making package for Epson compatible printers
- WORLD BY NIGHT AND DAY** - a display of the world showing night and day for any time and date of the year

Applications I Disc

- BUSINESS GRAPHICS** - for producing graphs, charts and diagrams
- VIDEO CATALOGUER** - catalogue and print labels for your video cassettes
- PHONE BOOK** - an on-screen telephone book which can be easily edited and updated
- PERSONALISED LETTER-HEADINGS** - design a stylish logo for your letter heads
- APPOINTMENTS DIARY** - a computerised appointments diary
- MAPPING THE BRITISH ISLES** - draw a map of the British Isles at any size
- SELECTIVE BREEDING** - a superb graphical display of selective breeding of insects

MagScan - a comprehensive magazine database



An updated version of Magscan, which contains the complete indexes to all BEEBUG magazines from Volume 1 Issue 1 to Volume 10 Issue 10.

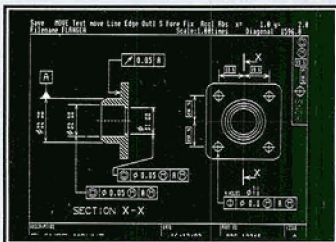
Some of the features Magscan offers include:

- ◆ full access to all BEEBUG magazines
- ◆ rapid keyword search
- ◆ flexible search by volume number, article type and up to two keywords
- ◆ keyword entry with selectable AND/OR logic
- ◆ extensive on-screen help
- ◆ hard copy option
- ◆ easily updatable to include future magazines
- ◆ yearly updates available from BEEBUG

Magscan allows you to locate instantly all references to any chosen subject mentioned anywhere in the 95 issues of BEEBUG magazine to date.

Just type in one or two descriptive words (using AND/OR logic), and you can find any article or program you need, together with a brief description and reference to the volume, issue and page numbers. You can also perform a search by article type and/or volume number.

The Magscan database can be easily updated to include future magazines. Annual updates are available from BEEBUG for existing Magscan users.



ASTAAD

Enhanced ASTAAD CAD program for the Master, offering the following features:

- * full mouse and joystick control
- * built-in printer dump
- * speed improvement
- * STEAMS image manipulator
- * Keystrips for ASTAAD and STEAMS
- * Comprehensive user guide
- * Sample picture files

	Stock Code	Price		Stock Code	Price
ASTAAD (80 track DFS)	1407a	£ 5.95	ASTAAD (3.5" ADFS)	1408a	£ 5.95
Applications II (80 track DFS)	1411a	£ 4.00	Applications II (3.5" ADFS)	1412a	£ 4.00
Applications I Disc (40/80T DFS)	1404a	£ 4.00	Applications I Disc (3.5" ADFS)	1409a	£ 4.00
General Utilities Disc (40/80T DFS)	1405a	£ 4.00	General Utilities Disc (3.5" ADFS)	1413a	£ 4.00
Arcade Games (40/80 track DFS)	PAG1a	£ 5.95	Arcade Games (3.5" ADFS)	PAG2a	£ 5.95
Board Games (40/80 track DFS)	PBG1a	£ 5.95	Board Games (3.5" ADFS)	PBG2a	£ 5.95

All prices include VAT where appropriate. For p&sp see Membership page.

Board Games

SOLITAIRE - an elegant implementation of this ancient and fascinating one-player game, and a complete solution for those who are unable to find it for themselves.

ROLL OF HONOUR - Score as many points as possible by throwing the five dice in this on-screen version of 'Yahtze'.

PATIENCE - a very addictive version of one of the oldest and most popular games of Patience.

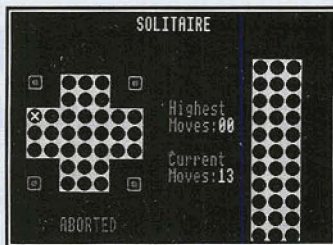
ELEVENSES - another popular version of Patience - lay down cards on the table in three by three grid and start turning them over until they add up to eleven.

CRIBbage - an authentic implementation of this very traditional card game for two, where the object is to score points for various combinations and sequences of cards.

TWIDDLE - a close relative of Sam Lloyd's sliding block puzzle and Rubik's cube, where you have to move numbers round a grid to match a pattern.

CHINESE CHEQUERS - a traditional board game for two players, where the object is to move your counters, following a pattern, and occupy the opponent's field.

ACES HIGH - another addictive game of Patience, where the object is to remove the cards from the table and finish with the aces at the head of each column.



Applications II Disc

CROSSWORD EDITOR - for designing, editing and solving crosswords

MONTHLY DESK DIARY - a month-to-view calendar which can also be printed

3D LANDSCAPES - generates three dimensional landscapes

REAL TIME CLOCK - a real time digital alarm clock displayed on the screen

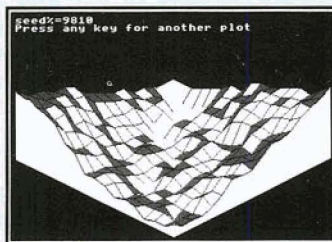
RUNNING FOUR TEMPERATURES - calibrates and plots up to four temperatures

JULIA SETS - fascinating extensions of the Mandelbrot set

FOREIGN LANGUAGE TESTER - foreign character definer and language tester

SHARE INVESTOR - assists decision making when buying and selling shares.

LABEL PROCESSOR - for designing and printing labels on Epson compatible printers



Arcade Games

GEORGE AND THE DRAGON - Rescue 'Hideous Hilda' from the flames of the dragon, but beware the flying arrows and the moving holes on the floor.

EBONY CASTLE - You, the leader of a secret band, have been captured and thrown in the dungeons of the infamous Ebony Castle. Can you escape back to the countryside, fighting off the deadly spiders on the way and collecting the keys necessary to unlock the coloured doors?

KNIGHT QUEST - You are a Knight on a quest to find the lost crown, hidden deep in the ruins of a weird castle inhabited by dangerous monsters and protected by a greedy guardian.

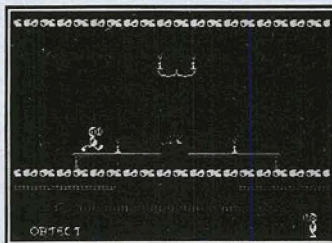
PITFALL PETE - Collect all the diamonds on the screen, but try not to trap yourself when you dislodge the many boulders on your way.

BUILDER BOB - Bob is trapped on the bottom of a building that's being demolished. Can you help him build his way out?

MINEFIELD - Find your way through this grid and try to defuse the mines before they explode, but beware the monsters which increasingly hinder your progress.

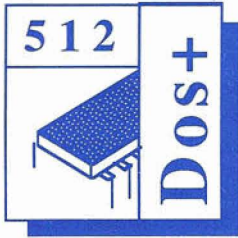
MANIC MECHANIC - Try to collect all the spanners and reach the broken-down generator, before the factory freezes up.

QUAD - You will have hours of entertainment trying to get all these different shapes to fit.



	Stock Code	Price		Stock Code	Price
File Handling for All Book	BK02b	£ 9.95	File Handling for All Disc (3.5" ADFS)	BK07a	£ 4.75
File Handling for All Disc (40/80T DFS)	BK05a	£ 4.75	Joint Offer book and disc (3.5" ADFS)	BK06b	£ 11.95
Joint Offer book and disc (40/80T DFS)	BK04b	£ 11.95	Magscan (40 DFS)	0011a	£ 4.75
Magscan (40 DFS)	0005a	£ 9.95	Magscan Upgrade (40 DFS)	0010a	£ 4.75
Magscan (80T DFS)	0006a	£ 9.95	Magscan Upgrade (80T DFS)	1458a	£ 4.75
Magscan (3.5" ADFS)	0007a	£ 9.95	Magscan Upgrade (3.5" ADFS)		

All prices include VAT where appropriate. For p&p see Membership page.



512 Forum

by Robin Burton

This month I thought we'd start to investigate one of the most complex, but potentially one of the most useful of the 512's utilities. So far as I recall it's never been a Forum topic before.

In fact this program is common to all DOS systems, is easy to use and is capable of rescuing you from potential disaster in certain circumstances. In spite of that, it's probably one of if not the least understood of DOS's utilities.

DISC PHILOSOPHIES

Before we get on to the program in question, a reminder about how DOS discs are structured is in order, along with a mention of the sort of problems that can afflict anyone. If you have the 512 Technical Guide from Dabs Press you can find more about the subject of disc organisation in chapter nine, but for the rest of you here's the short version.

Just like most microcomputer filing systems, the contents of DOS discs are stored in a hierarchical directory structure. As you know, if you issue a *INFO in either DFS or ADFS, one of the pieces of information given for each object is the physical location on the disc where the object is stored. Since there are no other pointers to a file or directory in these systems, the whole of the data must naturally follow the indicated start-point on the disc.

There's a problem with this simplistic approach to disc storage though. The entire file or directory must be able to fit onto the disc in one 'chunk', otherwise

you get the "Disc full" message in DFS or ADFS, or alternatively "Compaction required" in ADFS if there's space but it's not in big enough pieces.

Likewise, if you extend an existing DFS file there's always the risk of a "Can't extend" error if another file physically follows the one being updated. ADFS sometimes avoids this by moving a file if it can, but that's slow and still requires a large enough free area elsewhere on the disc. If there isn't one you're back to the previous problem. Added to these limitations, the number of files you can have in a directory is also fixed.

DOS DISCS

DOS doesn't suffer from any of these difficulties, because its disc organisation is more sophisticated. It doesn't store the physical address of an object with its name in the directory, but instead each directory entry, for files and sub-directories, has only a pointer into a table. This table, called the file allocation table, or FAT for short, is where the physical location of the data is stored.

To explain the FAT we need to look at one more piece of the puzzle. Physically DOS discs are formatted into tracks and sectors of course, but for the purposes of data storage a number of sectors (always 1, 2, 4 or 8) are grouped together into a notional unit called a cluster, and a cluster is the unit of disc space that is allocated or added to a file or a directory. That's why clusters are also called allocation units.

Since every cluster on a disc has a FAT entry of its own, the number of the FAT entry directly yields the cluster number

by simple addition, while a further simple calculation on the cluster number gives the physical start sector of the cluster on the disc.

Since the actual number of the FAT entry tells the filing system where the data is physically located, the contents of the FAT entry are free to be used for another purpose. Specifically, the content of a FAT entry is either the number of the next FAT entry used by the object, or is an 'end of chain' marker in the last cluster. Clusters can also be marked as faulty, but that needn't concern us here.

Of course, sector sizes vary from disc type to disc type, as can cluster sizes, but the basic method of storing data remains constant. All you need is the location of the FAT, the number of sectors per cluster and the size of a sector, and you can work out where everything is. On DOS discs this information is stored in the first data sector of the disc.

DOS can therefore allocate files or sub-directories (not the root, which is always present and of a fixed specified size) one cluster at a time, so the minimum size for any object or an extension to an object is one cluster. If, for example, a file is to be extended, DOS simply looks for the next free FAT entry, places its number in the old 'last' one and makes the new one the end of file cluster. On DOS discs therefore, when you get a disc full message it really does mean that the data will not fit.

It's easy to see that over time, as new files are added, old ones grow and others contract or are deleted, a file or a sub-directory might well consist of numerous clusters dotted all over the disc. This is exactly what happens and when it does you might notice, particularly on floppies, that disc performance deteriorates. However, this is part the price to be paid

for avoiding the more frequent, more annoying and occasionally disastrous problems of the BBC's simpler filing systems.

The problem of fragmented files, as this situation is usually referred to, is easily fixed by copying the files, one at a time, to an empty disc. As each file is copied to the new disc it is allocated consecutive clusters, so the files become defragmented. Copy the new disc back to the working disc and the problem is removed.

POTENTIAL PROBLEMS

Of course most of the time discs and drives are pretty reliable, but trouble can strike, so is the system inherently unreliable?

No it isn't, but failures can be caused by a hardware problem or more simply by power failure. Unfortunately in some cases, even a minor power fluctuation might be enough to cause problems for the 512, particularly if yours is housed in an external co-processor adaptor like mine is. Perhaps I'd better explain that remark.

All models of BBC micro employ a switched-mode power supply. These are quite sophisticated devices and are also quite expensive, but using one means that the machine will withstand quite serious power drops or surges without difficulty. In fact my local dealer once tested a model B and found it ran perfectly at less than 180 volts!

If your 512 is fitted into a Master it naturally shares the Master's switched-mode power supply, hence it is reasonably immune to the vagaries of the electricity supply.

However, for cost reasons the power supplies used in co-processor adaptors are

the much cheaper and simpler linear type. These are no more than a transformer, so any input power fluctuation is passed on, albeit at reduced voltage, in its original proportions. If the mains voltage drops by 10% momentarily, the output power drops 10% too. This is more than sufficient to crash a 512.

Where I live the mains supply becomes highly unreliable at the least suggestion of bad weather. On many occasions over the years I've found the BBC host appears fine, the screen looks OK and the BBC's interrupts are still running (verified by pressing either Caps Lock or Shift Lock and watching the lights), but the 512 has crashed. I know this is a power problem, because frequently I've noticed the lights dim for a moment when the problem appears. The BBC is quite happy, but it's too much for the 512.

Nine times out of ten a re-boot is all that's needed, plus a few minutes to catch up from where the last automatic save had protected my data. At the same time I also usually mutter a few words to the lords of National Power wishing they could hear me, but perhaps it's just as well that they can't.

Of course the problem is that, inevitably, from time to time one of these power glitches occurs just when a disc is being updated. The results depend on precisely when the fluctuation occurs, since although physical disc updates take place in the BBC micro, the instructions for them are issued from the 512.

If problems happen after the file data has been written but before the FAT is updated, all the file data can be on the disc, but the FAT knows nothing about the changes. Alternatively, if the glitch occurs during the issuing of instructions for a physical disc write, almost anything can happen. The file data or the FAT can

be corrupted. Worse, though thankfully not too often, the data for either the file or the FAT can be written to the wrong place on the disc.

Whatever happens, if you have a disc problem you encounter the major disadvantage of DOS's more sophisticated method of storing disc data. If all else fails it's quite easy to recover data from DFS or ADFS discs by reading sectors. After all, even without the directory information, if you can find the start of a file you automatically know where the rest of it is.

By contrast, on DOS discs it's extremely difficult and very slow and tedious to decode the FAT 'by hand', even if you assume that its information is accurate. Having found the first part of a file you must reverse calculate the FAT entry number, then try to find out where the next piece of the file is.

Of course, if the FAT is corrupt you could consider searching an entire disc, sector by sector, but remember that a file's logical clusters can have been allocated in any physical order. You might just attempt this job with a floppy if you were absolutely desperate and the data was life or death, but on a hard disc the idea is utterly out of the question. As a guide and to justify that statement, my own hard disc's DOS partition consists of over 100,000 sectors.

Even using floppies, in practical terms it's nearly impossible to recover a file of any reasonable length from a DOS disc successfully by manual means, so what do you do?

ENTER CHKDSK

CHKDSK is a utility program supplied on disc one of the 512's DOS PLUS issue disc set. It's supplied in both versions, 1.2 and 2.1, and is also issued with all current PC versions of DOS.

As its name suggests, the basic function is to check a disc, but in fact it can also repair some of the types of damage to files and directories mentioned above.

The syntax of the command is:

```
CHKDSK [DRIVE] [/OPTION]
```

where DRIVE is any valid drive, supplied in the usual format (A:, B: etc) but it may be omitted when the current drive is assumed. OPTION, which we'll come to later, may also be omitted, in which case CHKDSK will simply check the disc concerned and report its findings.

The checking process involves reading all the entries in the root directory and following up the FAT entries, making sure they are all valid and that each of the linked list of clusters are correctly assigned and terminated. This sort of operation is easy for a program, but is a nightmare if attempted by hand. After the root directory, any sub-directories are read and similarly checked until all the directories and the FAT have been verified as looking sensible.

Assuming no errors, CHKDSK will just report the organisation of the disc and its free and used space. Alternatively, you can request more information by using the '/V' (verbose) option switch, which makes CHKDSK tell you what it's doing as it does it. Try, for example:

```
CHKDSK A: /V
```

with a disc in drive A: that contains several sub-directories with a number of files in each. The program can take quite a few seconds to run, but at least with the verbose option you can see what it's doing.

By the way, should CHKDSK find any problems it will report them and ask if you want them fixed, but unless you've specified the fix option (more of which next month) it won't actually rewrite anything, so it doesn't matter what you reply.

We're out of space again, so the other five option switches and what they do will have to wait until next month. B

Weather Station (continued from page 8)

```
2040 newmonth$=MIDS(M$,month%,3)
2050 nextmonth$=newmonth$+" "+STR$(yr%)
2060 *K.9 CH.nextmonth$|M
2070 *FX138,0,137
2080 ENDPROC
2090 :
2100 DEF PROClastmnth
2110 IF month$="JAN"THEN newmonth$="DEC
":yr%=yr%-1:GOTO 2150
2120 month%=INSTR(M$,month$)
2130 month%=month%-3
2140 newmonth$=MIDS(M$,month%,3)
2150 lastmonth$=newmonth$+" "+STR$(yr%)
2160 *K.9 CH.lastmonth$|M
2170 *FX138,0,137
2180 ENDPROC
2190 :
2200 DEF PROCdump
2210 REM Your own dump routine
```

```
2220 ENDPROC
2230 :
2300 DEF PROCerr
2310 VDU6
2320 REPORT:PRINT" at line ";ERL:END
2330 :
2340 REM Replace this data with your ow
n.
2350 :
2400 DATA JAN,88
2410 DATA 31
2420 DATA 1,29.83,2,29.38,3,29.4,4,29.4
8,5,29.53,6,29.39,7,30.07,8,30.28,9,30.2
1,10,30.28,11,30.28,12,30.18,13,29.78,14
,29.92,15,30.18
2430 DATA 16,30.33,17,30.38,18,30.27,19
,29.88,20,29.79,21,29.94,22,28.86,23,30.
08,24,29.54,25,30.33,26,29.53,27,29.94,2
8,29.84,29,29.09,30,29.34,31,29.93
```

Smart Renumber

*David Spencer presents a smart renumber utility from June 1988
to tidy up your Basic programs.*

Program listings always look much nicer, and are easier to follow, if the various sections are split up, with different number ranges for each part. This is a style that we at BEEBUG have always adopted in our program listing. In general, programs listed in BEEBUG follow a particular standard, namely:

- Lines 10 onwards are REM statements
- Lines 100 onwards are the main program
- Lines 1000 onwards are the function and procedure definitions, and data.

The problem with this convention is renumbering the program when any changes are made. The RENUMBER command in Basic will only renumber the entire program with a fixed increment between lines. Several utility ROMs offer a partial renumber, which allows only a section of the program to be renumbered, but this still means handling each part separately. However, the program presented here solves all that by renumbering a Basic program in sections with one simple star command.

To use the renumber command, the program listed here should be typed in, saved and run. This will save a machine code routine under the name *RENUM*. Make sure that you don't call the Basic source code *RENUM* as well, because it will then get overwritten by the machine code. If the name *RENUM* clashes with any star commands already in use, change it to an alternative of your choice.

In its simplest form, the renumber command is used by typing **RENUM* with no parameters. This will renumber the program currently in memory to

BEEBUG style, as described earlier. If the command is followed by one numeric parameter, for example **RENUM 2000*, then the program will be renumbered as before, but instead of procedures, functions and data starting at line 1000, they will start at the given line number.

The most complex form of the command is when two parameters are used, such as **RENUM 2000,1000*. In this form, the renumbering will be done as before, with procedures etc. starting at the line specified by the first parameter, but this time, each procedure, function or block of data is renumbered separately. The second parameter specifies the gap between blocks. In the example above, the first procedure would start at line 2000, the second at line 3000, the third at 4000 and so on.

The renumber command contains a 'safety' feature to prevent lines getting out of order. If at the start of a block (procedure, function etc.), the current line number is greater than the new line number, then the renumber just carries on without using the new number. For example, with **RENUM 1000,1000* if the first procedure finished after line 2000, then the second procedure would be numbered from where the first one finished, rather than going back to line 2000, in which case more than one line would have the same line number.

**RENUM* automatically deals with line number references in *GOTO*, *GOSUB*, *RESTORE* etc. in exactly the same way as the normal Basic *RENUMBER*. In other words, any simple line reference can be

changed, such as GOTO 1234, but calculated references, for example GOSUB 4*A%, cannot be handled. The 'Failed at ...' warning is generated if a reference is made to a non-existent line.

Because the assembled RENUM program is over &300 bytes long, it cannot load into the serial buffers at &900, which is the most common place for transient programs such as this. Instead, RENUM loads in at &404, which is part of Basic's workspace. This means that the resident integer variables (A%-Z%) are lost when *RENUM is used. The reason for starting at &404 and not &400 is so that the variable @% isn't lost, as this would upset the printing of numbers. Another consideration is the length of the program. The Basic workspace is only 1K (&400) bytes long. Therefore the program must not end past &7FF. As it stands, the *RENUM command ends at &709, so there is plenty of room to make any further additions.

If you want the first line to have a number other than 10, or for the main program to start at a line other than 100, then this can be done with a few simple changes to the program. The first line number is set in line 590 (LDA #10), and the first line after the header is determined by lines 700 and 720 (CMP #100 and LDA #100), and both instructions must be changed. The increment between lines is set in line 960 (ADC #10). Any of these values can be changed to a number in the range 0 to 255.

HOW IT WORKS

It is very easy to write a simple renumber routine for BBC Basic, because each line of the program stored in memory starts with the line number in binary.

Therefore, to renumber a program all that you need do is go through each line in turn and change the line number stored in memory. Each line also contains a length byte so that you can jump straight to the start of the next line without scanning through all the intervening bytes of code. The end of the program is identified by a line number that is greater than 32767. To renumber in increments of 10, starting at line 10, you merely need to store a value of 10, use that value as the new line number, and then add 10 to it before doing the next line.

The *RENUM command differs from a simple renumber as outlined above in three respects, namely:

1. It is a star command, which needs special routines to decode the parameters etc.
2. Lines are renumbered according to a certain standard, rather than with a constant increment.
3. References to line numbers in the program are adjusted as necessary.

Going through the listing line by line:

Lines 240-410 decode the parameters given after the *RENUM command.

Lines 420-570 store the line number of each line in memory starting immediately after the program. These numbers are needed to handle line references. If there is not enough space between TOP and HIMEM in which to store the line numbers, a 'RENUMBER space' error is generated.

Lines 580-890 perform the actual renumbering. Each line is examined to see what the next line number should be.

Smart Renumber

Lines 910-980 move the pointer to the program onto the next line, and add 10 to the current line number.

Lines 1000-1010 set up a pointer to the start of the Basic program in memory.

Lines 1030-1050 add the appropriate value to the line number of the last procedure to get the number for the next procedure. This is used when *RENUM is given with two parameters.

Lines 1070-1100 decode line references within a program. Rather than storing line numbers in GOTOs etc. as binary, BBC Basic stores them as a byte containing &8D, followed by the line number in a three byte form. This routine decodes the value to a sixteen bit binary number.

Lines 1120-1230 scan through the program after it has been renumbered, looking for line references. Each time one is found, the next routine is called to alter it.

Lines 1250-1680 are the heart of the line reference handler. The principle used is to decode the line reference into a binary number (see above), and then search through the list of old line numbers made at the start until that number is found. By counting the position of the number in the table, it is possible to work out the new line number by looking through the program. This new number is then converted back into the special form and put in the program. If the number can't be found in the table then a 'Failed at ..' message is given, followed by the current line number in decimal.

Lines 1700-1720 form a subroutine to skip spaces in the command line.

Lines 1740-1940 read a decimal number from the command line.

```
10 REM Program Smart Renumber
20 REM Version B1.0
30 REM Author David Spencer
40 REM BEEBUG June 1992
50 REM Program subject to copyright
60 :
100 DIM code &400
110 bassp=6:top=&12:page=&18
120 base=&70:inc=&72:dflag=&74
130 ptr=&75:line=&77:datf=&79
140 ptr2=&7A:lno=&7C:ptr3=&7E
150 temp=&80:curl=&81:dflag2=&83
160 ytemp=&84
170 osargs = &FFDA
180 osnew1 = &FFE7
190 oswrch = &FFEE
200 :
210 FOR pass=0 TO 3 STEP 3
220 P%=&404:O%=code
230 [OPT pass+4
240 LDA #1:LDY #0:STY dflag
250 LDX #ptr:JSR osargs
260 LDA #1000 MOD 256:STA base
270 LDA #1000 DIV 256:STA base+1
280 LDA #&FF:STA dflag2:JSR sskp
290 BCC renum:JSR nget:BCC bok
300 .syntax
310 BRK:EQUB &DC
320 EQU "Syntax: RENUM [<start>,<
increment>]]"
330 EQUB 0
340 :
350 .bok STA base:STX base+1
360 JSR sskp:CMP #ASC",":BNE bok2
370 INY:JSR sskp
380 .bok2 CMP #ASC" ":BCC renum
390 DEC dflag:JSR nget:BCS syntax
400 STA inc:STX inc+1:JSR sskp
410 BCS syntax
420 .renum:JSR setptr:LDA top
430 STA ptr2:LDA top+1:STA ptr2+1
440 .noloop
450 LDY #0:LDA (ptr),Y:STA (ptr2),Y
460 PHA:INY:LDA (ptr),Y:STA (ptr2),Y
470 PLA:BMI main:INY:LDA (ptr),Y
```



```

480 CLC:ADC ptr:STA ptr:BCC noup
490 INC ptr+1
500 .noup LDA ptr2:CLC:ADC #2
510 STA ptr2:BCC noup2:INC ptr2+1
520 .noup2 LDA ptr2+1:CMP bassp+1
530 BCC noloop:BEQ chklo
540 .err BRK:EQUB 0
550 EQUUS "RENUMBER space":EQUB 0
560 .chklo LDA ptr2:CMP bassp
570 BCC noloop:BCS err
580 .main LDA #&FF:STA datf
590 JSR setptr:LDA #10:STA line
600 LDA #0:STA line+1
610 .loop LDY #3
620 .loop2 LDA (ptr),Y:INY
630 CMP #ASC" ":BEQ loop2
640 CMP #&F4:BEQ remok
650 CMP #ASC"." :BNE endofrem
660 .remok LDY #0:LDA line+1
670 STA (ptr),Y:INY:LDA line
680 STA (ptr),Y:INY:LDA (ptr),Y
690 JSR update:BCC loop:JMP exit
700 .endofrem LDA line:CMP #100
710 BCS loop3:LDA line+1:BNE loop3
720 LDA #100:STA line
730 .loop3 LDY #3
740 .loop4 LDA (ptr),Y:INY
750 CMP #ASC" ":BEQ loop4
760 CMP #&DC:BCC nodef:CMP #&DE
770 BCS nodef:CMP #&DD:ROR datf
780 BIT dflag2:BPL nodef:LDA datf
790 AND #&C0:BEQ nodef:LDA dflag
800 STA dflag2
810 .upnok LDA line:CMP base
820 LDA line+1:SEC base+1:BCC upok
830 JSR lineadd:BCC upnok
840 .upok LDA base:STA line
850 LDA base+1:STA line+1:JSR lineadd
860 nodef LDY #0:LDA line+1
870 STA (ptr),Y:INY:LDA line
880 STA (ptr),Y:INY:LDA (ptr),Y
890 JSR update:BCC loop3:BCS exit
900 :
910 .update CLC:ADC ptr:STA ptr
920 BCC update2:INC ptr+1

```

```

930 .update2 LDY #0:LDA (ptr),Y
940 BPL update3:SEC:RTS
950 .update3 LDA line:CLC
960 ADC #10:STA line:BCC update4
970 INC line+1
980 .update4 CLC:RTS
990 :
1000 .setptr LDA page:STA ptr+1
1010 LDA #1:STA ptr:RTS
1020 :
1030 .lineadd CLC:LDA base
1040 ADC inc:STA base:LDA base+1
1050 ADC inc+1:STA base+1:RTS
1060 :
1070 .dec LDA (ptr),Y:ASL A:ASL A
1080 TAX:AND #&C0:INY:EOR (ptr),Y
1090 STA lno:INY:TXA:ASL A:ASL A
1100 EOR (ptr),Y:STA lno+1:RTS
1110 :
1120 .exit JSR setptr:.eloop
1130 LDY #0:LDA (ptr),Y:BPL eloop1
1140 RTS
1150 .eloop1 STA curl+1:INY
1160 LDA (ptr),Y:STA curl:INY
1170 .eloop2 INY:LDA (ptr),Y
1180 CMP #13:BEQ exnxt:CMP #&8D
1190 BNE eloop2:JSR change
1200 JMP eloop2
1210 .exnxt SEC:TVA:ADC ptr
1220 STA ptr:BCC eloop:INC ptr+1
1230 BCS eloop
1240 :
1250 .change INY:STY ytemp
1260 JSR dec:LDA top:STA ptr3
1270 LDA top+1:STA ptr3+1:LDA page
1280 STA ptr2+1:LDA #1:STA ptr2
1290 .hunt LDY #0:LDA (ptr2),Y
1300 BMI nof:LDA lno+1:CMP (ptr3),Y
1310 BCC nof:BNE hnext:LDA lno
1320 INY:CMP (ptr3),Y:BCC nof
1330 BNE hnext:LDA (ptr2),Y
1340 LDY ytemp:INY:PHA:AND #&3F
1350 ORA #&40:STA (ptr),Y
1360 STY ytemp+1:LDY #0
1370 LDA (ptr2),Y:LDY ytemp+1

```

Smart Renumber

```
1380 PHA:AND #&3F:ORA #&40
1390 INY:STA (ptr),Y:DEY:DEY
1400 PLA:AND #&C0:LSR A:LSR A
1410 LSR A:LSR A:STA temp:PLA
1420 AND #&C0:LSR A:LSR A
1430 ORA temp:EOR #&54:STA (ptr),Y
1440 INY:INY:RTS
1450 .hnext LDA ptr3:CLC:ADC #2
1460 STA ptr3:BCC hnext2
1470 INC ptr3+1
1480 .hnext2 LDY #2:LDA (ptr2),Y
1490 CLC:ADC ptr2:STA ptr2
1500 BCC hunt:INC ptr2+1
1510 BCS hunt
1520 .nof LDX #9
1530 .nof2 LDA fmess,X:JSR oswrch
1540 DEX:BPL nof2:LDA curl
1550 STA temp:LDA curl+1:STA temp+1
1560 CLC:PHP:LDY #4
1570 .dp LDX #&30
1580 .dp2 SEC:LDA temp:SBC tenlo,Y
1590 PHA:LDA temp+1:SBC tenhi,Y
1600 BCC dp3:STA temp+1:PLA
1610 STA temp:INX:BCS dp2
1620 .dp3 PLA:TXA:CMP #&30
1630 BEQ dp4:PLP:SEC:BCS dp5
1640 .dp4 PLP:BCS dp5:PHP
1650 BCC dp6
1660 .dp5 PHP:JSR oswrch
1670 .dp6 DEY:BPL dp:JSR osnew1
1680 PLP:LDY ytemp:INY:INY:RTS
1690 :
1700 .sskp DEY
1710 .sskp2 INY:LDA (ptr),Y
1720 CMP #ASC" ":BEQ sskp2:RTS
1730 :
1740 .nget LDA #0:STA ptr2+1
1750 LDA (ptr),Y:JSR tenchk
```

```
1760 BCC nget2:RTS
1770 .nget2 STA ptr2
1780 .nget3 INY:LDA (ptr),Y
1790 JSR tenchk:BCC nget4:CLC
1800 LDA ptr2:LDX ptr2+1:RTS
1810 .nget4 PHA:LDA ptr2+1:PHA
1820 LDA ptr2:ASL A:ROL ptr2+1
1830 ASL A:ROL ptr2+1:ADC ptr2
1840 STA ptr2:PLA:ADC ptr2+1
1850 ASL ptr2:ROL A:STA ptr2+1
1860 PLA:ADC ptr2:STA ptr2
1870 BCC nget5:INC ptr2+1
1880 .nget5 JMP nget2
1890 :
1900 .tenchk CMP #ASC"9"+1
1910 BCS tenchk1:CMP #ASC"0"
1920 BCC tenchk2:CLC:AND #&F
1930 .tenchk1 RTS
1940 .tenchk2 SEC:RTS
1950 :
1960 .fmess EQU$ " ta deliaF"
1970 :
1980 .tenhi
1990 EQU$ 1 DIV 256
2000 EQU$ 10 DIV 256
2010 EQU$ 100 DIV 256
2020 EQU$ 1000 DIV 256
2030 EQU$ 10000 DIV 256
2040 .tenlo
2050 EQU$ 1 MOD 256
2060 EQU$ 10 MOD 256
2070 EQU$ 100 MOD 256
2080 EQU$ 1000 MOD 256
2090 EQU$ 10000 MOD 256
2100 :
2110 ]NEXT
2120 OSCLI("SAVE RENUM "+STR$~code+" "+
STR$~0%+" 404 404")
```

Points Arising....Points Arising....Points Arising....Points Arising....

PROGRAM DEVELOPMENT MADE EASY

(Vol.10 No.3)

Occasionally the Basic-to-Text routines XED and VED corrupt the program. To

correct the problem add the following line to XEDbas and VEDbas:

```
285 LDA top:CMP #&FD:BCC ov:DEY:.ov
```




Direct Memory Access (2)

by Alan Wrigley

DOLLAR OPERATOR

In last month's *1st Course* we introduced the subject of direct memory access, and considered the method by which this is done, using the so-called indirection operators. Two of these were described in detail - the *query* (?) and *pling* (!) operators which access a single byte and four bytes of memory respectively. We need now to look at the third indirection operator, *dollar* (\$). As you might expect, this is used to write strings directly into memory, and to read them back again. The syntax is exactly the same as for query and pling:

```
&A00="Hello"  
text$=$loc%
```

The first example places a string consisting of the word "Hello" into memory starting at location &A00, while the second example reads a string of characters starting at the address held in *loc%*.

Whereas the other two operators by their very nature access a specific number of bytes in each operation, a string can be any length from 1 to 255 characters. The dollar operator therefore adds a carriage return (ASCII 13) to the end of the string when placing it in memory. When reading a string using the operator, the string is made up of all characters up to the first carriage return found. However, if no carriage return is found after the first 255 characters, a null string is returned. You can see the process at work if you type in the following short program:

```
10 DIM block% 256  
20 $block%=STRING$(255,"A")  
30 PRINT $block%
```

You should see a string of 255 "A"s

displayed on screen - this is the string which you have placed at *block%* in line 20. The string starts from position 0 upwards, i.e. *block%+0*, *block%+1* etc., leaving room for the carriage return at *block%+255*. Now add the following line and run the program again:

```
25 block%?255=65
```

This time nothing will be displayed. This is because you have overwritten the carriage return at *block%+255* with a further A (ASCII 65), and as a result the dollar operator in line 30 will not now find a carriage return within the first 255 characters. If you type:

```
PRINT LEN($block%)
```

to ascertain the length of the string at *block%*, you will get a result of zero.

Now remove line 25 and alter line 20 to:

```
20 $block%="Hello"+CHR$13+"World"
```

The word "Hello" will be printed, but not the rest of the string you inserted in line 20, because the operator will read only as far as the carriage return you have placed between "Hello" and "World". So you cannot store a carriage return as part of a string using the \$ operator.

Unlike query and pling, there is no extension to the syntax as we described in last month's article. Thus although you can access the byte at *block%+4* by using *block%?4*, if you want to access the string starting at *block%+4* you must use the full syntax: $\$(block\%+4)$ - note that the brackets are essential.

WHICH AREA OF MEMORY?

We have now discussed how to access memory directly, but we still have two important questions to answer: why would we need to do so, and how do we

know which part of memory to access? The answers to both these questions are inextricably linked, since the location of the memory that we access usually depends on the reason for using the technique in the first place. However, we can divide the use of memory broadly into two categories: those situations where we address a specific location in the computer's memory map, and those situations where we ask Basic to reserve a block of memory for us. We will consider each of these methods in detail, but first a few brief examples of the kind of tasks for which you might want to access memory will help to clarify the following sections.

USES FOR DIRECT MEMORY ACCESS

The following are all common uses of direct memory access: loading and running machine code utilities, either from within a program or as a separate stand-alone operation; creating blocks of data for processing by your program, which can be saved and loaded with just one **Save* or **Load* command; creating single-byte arrays (Basic's arrays always have four-byte elements, which is wasteful of memory if they will only ever hold values of 255 or less); accessing input/output ports; handling chunks of text (e.g. as in word processors); writing directly to the screen (as many games do for speed). This list is by no means exhaustive but gives you some idea of the range of possibilities. We will start to look at some of these in detail next month.

SPECIFIC MEMORY LOCATIONS

To access particular locations in memory explicitly, we can either quote the address itself (which is usually and more conveniently done in hex):

```
!&A00=1234
```

or we can use a variable which holds the address:

```
FOR loc%=&7000 TO &7020  
PRINT ?loc%  
NEXT
```

Obviously we need to know which addresses to specify. This depends on whether our aim is to gain access to a specific area of the computer's operation, such as the screen memory or the operating system, or whether it is to use for ourselves a spare chunk of memory which is not currently being used for any other purpose.

The computer's memory is divided into sections, all of which are allocated a specific purpose. But how much of this memory is actually being used at any one time depends on the configuration of the computer, the nature of the program which is running, the screen mode in use and so on. It is often possible to predict which areas of memory will remain unused throughout the operation of a program, and would therefore be available for your own use if you so wished. Similarly, areas which are in use will normally be in a predictable place if you wish to access them; for example, if you know the screen mode being used, you can predict where the screen memory will be.

A full description of the memory map is beyond the scope of this article (see *1st Course*, Vol.10 No.9 for more information), but certain parts of it will be considered in more detail later in the series when we look at some specific uses of direct memory access. For the moment, however, we can look at one or two areas which are often useful if you are looking for a safe little place to store a block of data or a machine code program, for example.

Pages 9 and 10 of the memory map (i.e. locations &900-&9FF and &A00-&AFF) are popular with programmers seeking storage space. They are normally set aside for such purposes as cassette, RS423 and speech buffers, and additional sound workspace (for envelopes 5-16). In other words, on a disc-based micro without a speech system which does not use the RS423 port or the additional sound envelopes, these areas are quite safe to use.

Pages 11 and 12 (&B00-&BFF and &C00-&CFF) have different uses depending on which computer you have. On a model B they are used for the soft key expansion buffer and for expanded character definitions respectively. So if you have any function keys defined, or you have defined the characters 224-255 by using VDU 23, then using these areas for any other purpose will corrupt the definitions. On a Master, however, these pages are used as Econet workspace, so you can use them if your computer is not connected to a network.

In our *1st Course* article on pseudo-variables (Vol.10 No.9) we said that on a disc-based model B, many programs could be moved down in memory to create more space by lowering the value of PAGE. We could equally well use some of this space below PAGE for our own purposes, provided of course that the program does not need the extra memory itself in order to run. In this case, we would leave PAGE where it is (typically &1900 on a model B) and access the memory below it directly. The same restrictions apply as were described in the earlier article - the lowest location you can use depends on how much space will be needed for filing system buffers.

If the areas mentioned above are unusable, or if you need even more

space, you can always raise PAGE or lower HIMEM, and use the memory thus freed for your own storage. PAGE must be altered *before* you load the program, but HIMEM can be changed from within the program (though not within a function or procedure). If you are using mode 7 or a shadow mode, there is often plenty of spare memory available to do this unless your program is very long or needs a large amount of space for its own variable storage. You do need to make sure, though, when using this method that there is no danger of the program running out of memory when it is run.

MEMORY RESERVED BY BASIC

In the case of memory which is reserved by Basic, we do not need to know where it is situated. We reserve a block of memory by using the DIM statement, as in the following example:

```
DIM myblock% 999
```

In response to this statement, Basic reserves a block of 1000 bytes (it is always one more than the value specified) somewhere in the area between the top of the program and HIMEM - i.e. the area set aside for program variable storage. It places the address of the start of this block into *myblock%* (this variable can have any name you like). As far as the program is concerned, the actual address of *myblock%* is utterly irrelevant, since all access to it can be carried out by using the indirection operators on the variable itself, as in the following examples:

```
PRINT $(myblock%+20)
!myblock%=&4D3A
val%=myblock%?offset%
```

If you are unfamiliar with the DIM statement, you should note that there are two distinct ways of using it. In the example given above, the syntax is:

```
DIM <variable_name> <length>
```

First Course - Direct Memory Access

(note the mandatory space between the two parameters) and this is the form you should always use to reserve a block of memory as described. The other use of DIM is to dimension an array, and this takes the form of the following examples:

```
DIM name$(11)
DIM number%(3,25)
```

where the maximum size of each dimension of the array follows the variable name in brackets (in this case we have declared a single-dimensional string array called *name\$* with 12 elements, and a two-dimensional numeric array called *number%* with maximum sizes of 4 and 26 respectively).

You must not get confused between these two quite separate uses of DIM, and for the purposes of this article only the first is relevant. You should also note that, whereas when dimensioning an array all the elements of the array are initialised

(by setting them to zero if a numeric array and to null strings if not), when you reserve a block of memory Basic makes no attempt to initialise it in any way. In other words, you should not make any assumptions about the initial contents of any of the memory locations in the block.

We have now looked in detail at indirection operators and at the choice of memory itself - the "how" and the "what" of direct memory access. Next month we shall start to look at the reasons for using the technique - in other words the "why". We will also consider how you decide for each individual purpose whether to use specific memory locations or blocks reserved by Basic, since although we have described both methods we have not given any guidance on when each is best used.

B

Input ROM (continued from page 12)

```
1470      INY:LDA(param),Y
1480      JSR check:STA low:INY
1490      LDA(param),Y:SEC:SEC low
1500      TAX:INX:STX high:RTS
1510 .tab  PHX:PHY:LDA#1:SEC
1520      SBC tb:STA tb:TAX:CLC
1530      LDA#4:LDY#0:JSR byte
1540      PLY:PLX:JMP loop
1550 .alter JSR osrdch:BCS retla:CLC
1560      CLC:ADC#128:JMP ascii
1570 .retla JMP loop
1580 .bad   TSX:LDA &108,X:STA temp
1590      LDY#0
1600 .badlp PHY:LDA param:STA &F6
1610      LDA param+1:STA &F7
1620      LDY temp:JSR rdsc:PLY
1630      STA buffer,Y:CLC
1640      LDA param:ADC#1:STA param
1650      LDA param+1:ADC#0
1660      STA param+1:INY:CPY#5
1670      BNE badlp:LDA#buffer
```

```
1680      STA param:STZ param+1:RTS
1690 .check CMP#0:BNE lowok:LDA#1
1700 .lowok RTS
1710 ]
1720 tb=&D92
1730 max=&D93
1740 low=&D94
1750 high=&D95
1760 temp=&D96
1770 wvec=&D97
1780 rsp=&D99
1790 oldcur=&D9A
1800 rom=&D9B
1810 NEXT
1820 T%=0:FOR A%=ass TO (0%-1)
1830 T%=T%+?A%:NEXT
1840 PRINT"to install type : "
1850 PRINT""SRWRITE ";~ass;" ";~0%;" 8
000 4"
1860 PRINT"and press CTRL-BREAK."
1870 END
```

B

The Hybrid Music System for the BBC Micro

Reviewed by Robert Lindsell

Product Supplier	Music Publisher Hybrid Technology Ltd. 273 The Science Park, Cambridge CB4 4WE. Tel. (0223) 420360
Price	£70.50 inc. VAT

As a part-time music teacher, I find there is often a need to write out pieces for pupils; in addition, some arranging of ensemble parts is called for. Writing out bars and bars of manuscript is a time-consuming job and very laborious when accurate layout is needed, as in the case of scores, when alignment of the parts is vital. What better, then, with the Beeb sitting there, than to have a program to take some of the slog out of all this and produce clear and accurately laid out copies?

```
Hybrid MUSIC PUBLISHER
current score: none
size: 180 of 10243.
  clear score
  load score
  save score
  edit score
  set-up page
  start bar: 0
  preview score
  print score
Use ↑ ↓ then RETURN or ← →.
(C) 1991 A L Rowles/Hybrid
```

Music Publisher main menu

The existence of programs for other (more powerful) machines was already known, but the arrival on the scene of such a program for the BBC series presented an opportunity not to be missed. The requirements were modest indeed: a BBC Micro and either a 9-pin or 24-pin Epson-compatible graphics printer. My own setup of a Master 128

with 6502 co-processor, twin 80-track disc drives and an Epson MX-100 looked encouraging. Still, as a result of previous unfavourable experiences with certain programs, I would need convincing that this program would really do what was claimed before buying a copy. The opportunity to get some hands-on experience was kindly provided by BEEBUG and what follows is the result.

FIRST IMPRESSIONS

The package consists of an attractive book-style container, housing the ROM, an Issue Disc and a User Guide; there are also some labels for attaching to copies of the system disc generated from the Issue Disc.

After opening the User Guide and taking a quick look at the Introduction, I studied the Installation section. The immediate impression was of a well written and presented Guide, which runs to 70 pages (admittedly not all used). Following the instructions, the ROM was duly inserted into one of my external cartridges. The Issue Disc was in 40/80 track DFS format. Being used to ADFS, I wondered if it could be backed up onto ADFS, but this did not work, so DFS was selected and the system booted by the Shift/Break method. Nothing happened, so I switched off my Tube and started again. This time, things began to whirl. The program requires a DFS formatted disc to become a 'System disc' (for which the labels were supplied). Following the instructions on the screen, and a short time later, the message "Generation complete" appeared. I now had a start-up disc, synonymous with the System disc mentioned above.

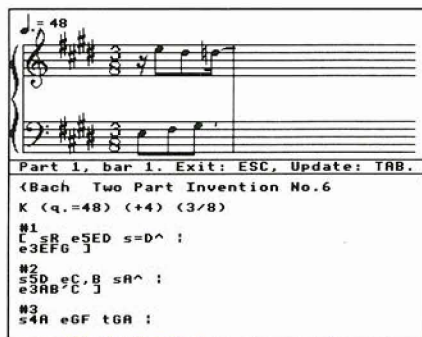
The start-up disc contains a number of musical examples as well as five

The Hybrid Music System for the BBC Micro

standard *templates* for such layouts as a single staff with treble clef, keyboard (treble and bass clefs with curly brackets) and so on, up to string quartet and four-part *choir*.

GETTING STARTED

In Section 3, "Getting Started", one is led through booting the disc and loading an example of a Bach Two-Part Invention to examine both the means of inputting the code and the end result on-screen. On booting, a menu appears, clearly laid out, giving a choice of clearing, loading, saving, or editing a score; in addition, there is a facility to *set-up page*, an item *start bar: 0*, *preview score* and *print score*. Selection is by means of the cursor keys and Return.



The screenshot shows a window titled "Music Publisher - composing". At the top, there is musical notation for two staves (treble and bass clefs) in G major, 3/8 time, with a tempo marking of quarter note = 48. Below the notation, the text reads: "Part 1, bar 1. Exit: ESC, Update: TAB." followed by "<Bach Two Part Invention No.6". Below that is the key signature: "K (q.=48) (+4) (3/8)". The main area contains three numbered staves of code: "#1 [e3R e5ED s=D^ : e3EFG]", "#2 s5D eC.B sA^ : e3AB E]", and "#3 s4A e6F t6A :".

Music Publisher - composing

A brief examination of the material on screen whetted my desire to enter some music of my own; the procedure seemed very logical, so I loaded the single line template and began input. The program uses two categories of code for entering the music in a word processing form, called *out-bar* and *in-bar*. Section 7 explains that *out-bar* refers to "overall aspects of the score such as the title, number of parts etc.", whilst the *in-bar* codes are for the actual entry of notes, rests and so forth. One great advantage of the system is that it works just like a word processor, so any alterations, additions or deletions can be made at any

time. Indeed, it is even suggested that editing could be done on a word processor, thereby making use of more powerful functions like copying (repeating) identical bars. The file is stored in ASCII format:

I duly entered and corrected a short tune - the program advises the user when mistakes are made, such as the incorrect total of note values in a bar. Using the *preview score* facility enables instant inspection bar by bar, to check the input. Another welcome feature is that the stems on notes automatically follow the usual convention and groups of quavers and shorter notes are correctly beamed. The user has an option of overriding these facilities when necessary.

DOWN TO TECHNICALITIES

Perhaps it is time to examine the means of entry a little more closely. Section 4 deals with this in a very clear way. By selecting *clear score*, the memory is cleared, then *edit score* produces the *out-bar* codes at the bottom half of the screen:

```
tTitle  
cby Composer
```

```
(t1) ; stave: treble clef & 1 part  
(c=90) ; metronome marking, e.g. (c.=80)  
(0) ; key signature, e.g. (+1), (-1)  
(4/4) ; time signature, e.g. (3/4), (6/8)
```

```
[  
]
```

The 't', followed immediately by the chosen title, places the title centrally in double width script. On the next line, the details following 'c' will be printed right-justified below the title. If not required, this can either be left blank or the line can be omitted. Then choose the number of staves and clefs (note that the semicolons are comment markers, i.e. everything which follows is ignored by the program). Up to six parts can be written, in treble, alto, tenor or bass clefs. The

The Hybrid Music System for the BBC Micro

Pharoah's song

No.5 from 'Go Moseal!'

Edward Davidson

The musical score is presented in two systems. The first system starts with a tempo marking of quarter note = 120. It features a vocal line in treble clef and a piano accompaniment in bass clef. The key signature has one sharp (F#). The lyrics are: "Take your peo- ple from this land, Take your bags and bag- gage too, Go Mos- es go!". The second system continues the vocal and piano parts. The lyrics are: "Through the drear- y des- ert sand, Take your cir- cus and your zoo, Go Mos- es go!, North or south or". The score includes bar lines, rests, and various musical notations such as beams and slurs.

'(c=90)' represents the metronome marking of 90 crotchets per minute, the example in brackets using the dotted crotchet as the unit. The key signature is defined by '+' and '-' for sharps and flats respectively, followed by the number required, up to a maximum of 7. The sharps or flats are then placed automatically, according to clef. Changes of key signature within a piece can be accommodated and are preceded by a double bar. First-time and second-time bars can also be produced.

IN-BAR EXPERIENCES

The first (empty) bar is then indicated, ready to begin writing. For the purpose of entering the notes, a range from 1 to 6 has been chosen for the octave in which the note is to be placed. For example, middle C would appear as 4C and that an octave higher would be 5C, all the intervening notes being prefixed by 4. Accidentals are denoted by +, -, and = respectively for sharp, flat and natural, these preceding the note which they qualify. The note values allowed are from semibreve to hemidemisemi-quaver and

are coded: w, m, c, q, s, d and h. Note the 'w' (whole note) which avoids confusion with semiquaver. If a dotted quaver F sharp above middle C were chosen, the code would be: q.4+F. It is unnecessary to repeat the figure for the pitch definition within a bar unless notes outside the chosen octave are to follow. In this case, there is a choice of techniques to perform this task, either by prefixing with a different figure or by using the relative codes "' and ',' for higher and lower octaves. The rests corresponding to the above note values are written simply by preceding 'R' by the length code. The program caters for percussion notes by using 'x' and triplets are produced by a '/' between the affected notes. Chords and arpeggios are easily produced by adding further characters.

Some further features are slurs and ties, staccato and tenuto signs, pauses and trill signs. Changes of register and pedal marks are also provided. Lyrics can also be added to song parts. When using lyrics, it may be necessary to remove portions of the beams which normally

The Hybrid Music System for the BBC Micro

group quavers and shorter notes. This is achieved by inserting a 'g' (for gap) at the appropriate point.

CODE ENTRY (CONTINUED)

Returning to my first efforts and completing the short tune and editing it bar-by-bar, I wondered how the hard copy would look. I referred to the *set-up page* instructions in Section 6 of the Guide to find that settings could be made, which would be saved and automatically re-loaded in future, for my Epson 9-pin printer and paper width. There is a note to the effect that Music Publisher does not fully use the additional resolution of the 24-pin printer but the smaller pin size is used to produce finer and more accurate stave lines. After setting and saving these options I selected *print score*. This option prints out the current score, using the *start bar* and *set-up page* parameters, and offers a choice of single-strike (quick) or double-strike (slower but denser) mode; printing can be terminated at any time by pressing Escape. It was at this point that I hit a snag: title and other textual heading features appeared, followed by a carriage return and one '@'. Hmm.....

On contacting Hybrid Technology, who asked for my printer manual to check the coding necessary for the bit-image graphics printing mode, I learned that my MX-100 could not be made to function (a problem inside the printer's ROM, they said). They kindly lent me an Epson FX-80 but in spite of trying every trick I knew, like *UNPLUGging other ROMs which might be incompatible, and using both serial and parallel data transfer, this didn't work either. Out of desperation, I took up an offer to visit a friend with a Kaga Taxan KP-810 (attached to a M128) and, finally - success!

LIKES AND DISLIKES

The concept of the program and the means of entering the musical notation are very commendable; and it must be

said that the User Guide is very well produced and easily understood. It takes very little time and practice to become quite adept at producing the required layout. With reasonable touch-typing facility, notation is at least as quick as with hand-written efforts and much better presented.

Three features I would have liked to have, but which were sadly missing, were:

1. Accents and "hairpins" for *cresc.* and *dim.* (I made do with actually entering written abbreviations as above in place of lyrics, followed by some dots showing the duration of the changes).
2. Ornament signs (mordents, etc.) and the ability to produce grace-notes.
3. Transposition. I suppose one can't have everything on the modest Beeb, with its limited RAM, but when arranging for families of instruments, as I do, it would be highly desirable.

I also wonder what could be done to make use of such enhancements as ADFS and, possibly, a second processor. Bearing in mind the problems involved in transferring graphics data across the Tube, there may well be reasons for excluding it.

As is the case with all utilities, it comes down to the personal requirements of the user. For many, Music Publisher would be very helpful as a means to producing good, clear and well laid-out music, and to be able to run off a copy at will is an advantage. In terms of value for money, it also seems reasonably priced. For my personal needs, I would prefer to have a program with those items mentioned above, certainly those in (1) and (2), which surely ought to be given serious consideration by the program writers. In conclusion, after my problems with obtaining hard copy (after all, that's the main object), I should insist on the program being demonstrated to work on my setup before parting with my

B

Games Review: Explorer

Peter Rochford explores a new adventure game for the BBC micro.

Product	Explorer
Supplier	Dragonsoft P.O. Box 22, Whitchurch, Shropshire SY13 2ZZ. Tel. (0948) 840522
Price	£25.99 inc. VAT and p&p

There has been little in the way of new adventure games for the Beeb in quite some time, so the arrival of this animated graphic adventure called *Explorer* from Dragonsoft should be met with some enthusiasm from the dungeons and dragons aficionados.

Supplied on a single 16K EPROM and two 5.25" discs, the game can be used on all BBC machines though it will benefit from the availability of some sideways RAM. A single disc drive is all else that is needed, although twin drives are definitely preferable, and both 40 and 80 track discs are supported. A fairly lengthy manual is supplied, and this contains some clear and easy to follow instructions along with a rather bizarre and humorous introduction to the game scenario.

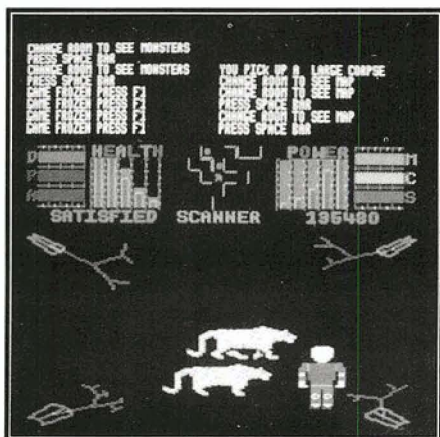
Briefly, the game revolves round the fact that you have been expelled from the Explorers Club for boasting about impossible exploits and must now prove yourself by setting off on a hazardous journey to the underworld of Hallar to conquer the fiendish Vandar.

Loading up the game, you are presented with a three section screen. The top features text messages and menus, whilst the middle contains status information on strength and position. The bottom section of the screen is the gameplay area.

Movement is via the usual Snapper style of keys, and you guide your man around the various rooms which are redrawn each time you move off screen. Other controls are provided via the use of the function keys or by a menu that appears in the text area. In

use, all this works very well, and progress is achieved with little fuss. A pause and save game facility is provided, as are controls for sound and a fast play feature.

The general look of the game is quite professional and the graphics, although not brilliant, are quite effective. Sound is kept to a minimum and used well to enhance the game where appropriate.



The game has some thirty-one levels, and each level consists of 31x31 rooms. There is a multitude of objects to collect and a great number of monsters and adversaries to overcome. Animation is good, and there is plenty of it to add interest to the game.

At this stage, even with the cheat version of the game available to the reviewer, I haven't managed to get too far with Explorer, but sufficient for this reviewer to say that this is certainly a challenging and involving game with a massive play area. I can find little to criticise in general, and the programming appears to be excellent.

If you are looking for a new graphic adventure to get your teeth into, then in my view Explorer is well worthwhile buying and should provide many hours of fun and frustration. Recommended. **B**

Public Domain Software

Alan Blundell continues his look at the PD scene with a review of educational shareware and free software.

Last month I ended by mentioning that John Lyons had decided to re-release almost his entire range of software as shareware, and summarised the meaning of 'shareware' as software which may be distributed by anyone to anyone, but which you must pay for if you find it useful. In my opinion, shareware is an excellent distribution method, enabling the user to 'try before you buy'. It does rely on honesty on the part of the user, but as I occasionally try to persuade some publishers, the shareware distribution method has many advantages: it costs nothing, removes the need for expensive advertising and (in the case of the Beeb) will reach people who may not see the product otherwise because the lack of commercial profitability at this stage in the life of the micro means that expensive advertising is not viable.

I hope and expect that much more ex-commercial software will become available as shareware or PD in time, but for now John's range is an excellent start and I congratulate him on his initiative. The range consists of 60 packages, many of which were originally marketed to technical colleges and the like. They originally sold for £15.00 per title but, as shareware, the registration fee (honest payment if you find the software useful) is only £7.50.

Many of the packages cover subjects such as hairdressing, food microbiology and hygiene, and areas of nutrition such as vitamins, fats and oils. Some of these will be of wider interest such as the

challengingly named 'Are You a Gourmet?' (I'm not, it seems), 'Eating Out' (understanding menus) and 'World in a Glass', which is ideal for would-be wine buffs. On a more down to earth level there are programs about both hot and cold water plumbing systems and, in line with the educational theme, a teachers' mark book package and programs to produce bar charts and graphs. Most of the packages are in the form of quizzes and multiple choice question and answers rather than tutorials, so there is plenty of opportunity for interaction.

Finally, the range of programs produced by Peter Davy (mentioned briefly in a recent column), although perhaps not of wide interest, is definitely worthy of investigation by anyone involved in Adult Basic Education. Peter has obviously devoted much time and effort to this area, and has a catalogue of 125 programs which deal with the subject of literacy and numeracy, making use of his experience as a volunteer literacy tutor. Whilst I'm not sure that these programs are strictly PD, they are free, and are well produced. Each program deals with a very specific area of numeracy or literacy and is designed to help teach or give practice in a specific skill. Many of the programs overlap in purpose but differ in their demands on the student, thus recognising the strengths and weaknesses of the individual - for example, a student may be a good reader generally, but have difficulties with spelling.

The skills dealt with range from the basic four rules of arithmetic to problems of time, fractions and percentages. Similarly the literacy programs cover word building, specific teaching points such as 'oo' and 'ee' sounds, to vocabulary comprehension for students learning English as a second language. Some programs have been written to make use of the Superior Software 'SPEECH!' system, which is a sensible use of other software by Peter.

Generally, his programs are very well thought out, have a fairly standard appearance so as not to distract students by making them think about the program instead of the exercise in hand, give encouraging comments at the end of an exercise, and display a sense of humour needed to make study interesting. Whilst all of the areas of literacy and numeracy covered also need to be taught to children, these programs were obviously written for use by adults, who don't need graphics such as 'Fun School' jumping teddies or lots of sound to keep their attention on the screen. For teaching of specific points, some of these programs may nevertheless be useful with older children.

I would like to congratulate Peter on his dedication and efforts. I know that his programs are in use all over the country in schools, hospitals, prisons, colleges and so on, and am sure that they are proving genuinely useful. If you would like to find out more about Peter's software, I'm sure he won't mind my mentioning his address again. Write to him at: 68 Headlands Road, Ossett, Wakefield, West Yorkshire, WF5 8HX. Please send an SAE if you do write, and be patient if you don't receive a very quick reply - he may have a lot of

correspondence to deal with in the near future!

I am keen to see more educational software appearing in the public domain, and am investigating several possibilities. To some extent, I am hampered in this by my lack of familiarity with the educational system. I have seen a number of programs which were developed for primary school use some years ago and distributed by the MEP. ('Microelectronics in Education Project'), for example. Whilst most primary schools will have copies of these, many parents will not have access to them, and I have been asked for more details in the past. As the MEP ceased to exist quite some time ago, I presume that the copyright in the software will have passed to some other body. Perhaps someone more in touch with the education world may be able to help me to investigate the possibility of making such software more widely available, to the benefit of us all?

There must be many useful educational programs which are PD which I don't know about, and probably many more which the authors would be willing to place in the public domain. I'm thinking of programs written by parents for their children, programs written by teachers for use in their own schools and other such worthy efforts. There are a number of parents who would be grateful for the opportunity to give good quality programs wider use, so I hope to see more in the future. If I do, you will hear about it in this column, but for next month, I will describe what's available in the field of applications software, by which I mean programs such as word processors, spreadsheets, databases and the like.

BEEBUG Function/Procedure Library (12)

by Cliff Blake

This month we present a set of useful graphical procedures and functions from Cliff Blake, using some clever mathematical functions to draw a

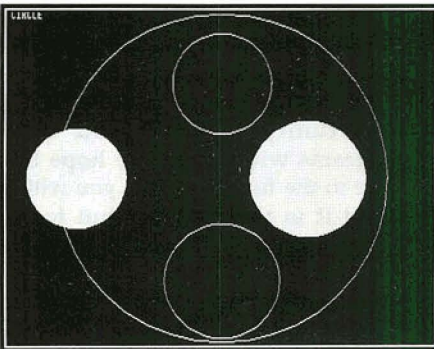
variety of regular shapes. The listing provided also includes a demonstration program so that you can see these ideas at work.

THE FUNCTION/PROCEDURE LIBRARY (PART 12)

Routine 1: Convert cartesian to polar co-ordinates.
Type: FUNCTION
Syntax: FNangle(x,y)
Purpose: Returns unambiguous angle for x,y co-ordinates.
Parameters: x Horizontal value y Vertical value
Notes: A trap should ensure that the function is not called if both x and y are zero. An ELSE branch should be provided to prevent the previous angle being used again incorrectly. The returned angle is in radians.
Related: None

Example:
140 IF x1<>0 OR y1<>0 THEN
rads=FNangle(x1,y1) ELSE GOTO 200
200 NEXT

Routine 2: Draw circle.

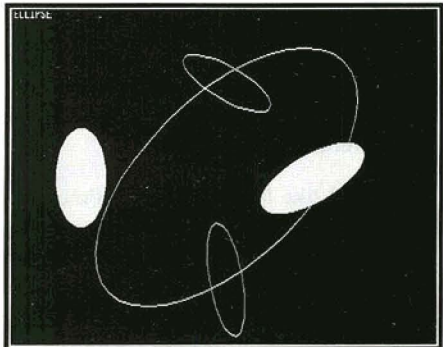


Circles

Type: PROCEDURE
Syntax: PROCcircle(Xc,Yc,r,f%)
Purpose: Plots a circle or disc.
Parameters: Xc,Yc Centre
r Radius
f% Fill=1 Clear=0
Notes: Uses the fast incremental technique to avoid repeated SIN or COS calculation.
Related: None

Example:
10 MODE 0:GCOL 0,1
20 PROCcircle(640,512,230,1)

Routine 3: Draw ellipse.



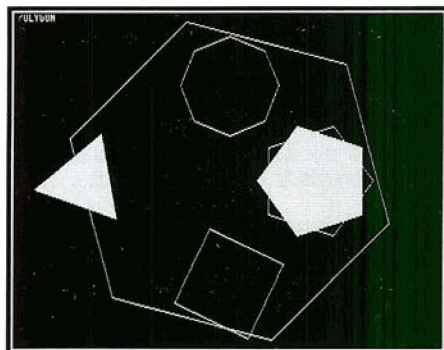
Ellipses

Type: PROCEDURE
Syntax: PROCellipse(Xc,Yc,ma,mi,t,f%)
Purpose: Plots an ellipse or platter.
Parameters: Xc,Yc Centre
ma Major semiaxis
mi Minor semiaxis

t Tilt angle degs.
f% Fill=1 Clear=0
Notes: Uses the fast incremental technique to avoid repeated SIN or COS calculation. The tilt angle is of the major axis.
Related: None

Example:
 10 MODE 0:GCOL 0,1
 20 PROCellipse (640,512,500,250,45,1)

Routine 4: Draw polygon.

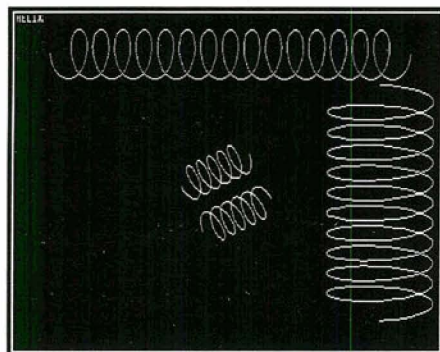


Polygons

Type: PROCEDURE
Syntax: PROCpolygon(s%,Xc,Yc,r,t,f%)
Purpose: Plots a clear or filled polygon.
Parameters: s% Number of sides
 Xc,Yc Centre
 r Radius to corner
 t Tilt angle degs.
 f% Fill=1 Clear=0
Notes: Uses the fast incremental technique to avoid repeated SIN or COS calculation. The tilt angle is of the start corner relative to horizontal.
Related: None

Example:
 10 MODE 0:GCOL 0,1
 20 PROCpolygon(6,640,512,250,45,0)

Routine 5: Draw helix.

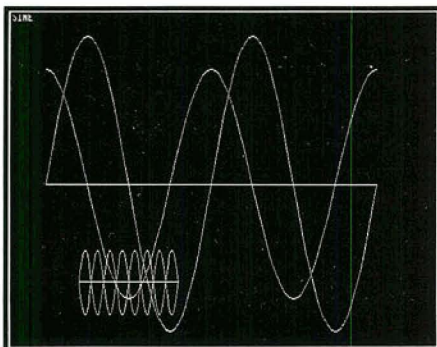


Helixes

Type: PROCEDURE
Syntax: PROCHelix(Xs,Ys,Xf,Yf,L%,w)
Purpose: Plots helix.
Parameters: Xs,Ys Start
 Xf,Yf Finish
 L% Number of loops
 w Width of coil
Notes: Uses the fast incremental technique to avoid repeated SIN or COS calculation. Useful for electrical inductors or mechanical springs.
Related: None

Example:
 10 MODE 0:GCOL 0,1
 20 PROCHelix(500,500,700,600,5,50)
 30 PROCHelix(770,470,570,370,5,50)

Routine 6: Draw sine wave.



Sine Waves

BEEBUG Function/Procedure Library

Type: PROCEDURE
Syntax: PROCsine(Xs,Xf,Yz,Yp,cy,ph)
Purpose: Plots a sinusoidal wave
Parameters: Xs Start
Xf Finish
Yz Zero level
Yp Peak amplitude
cy Number of cycles
ph Phase at start
Notes: Uses the fast incremental technique to avoid repeated SIN or COS calculation.
Related: None

Example:
10 MODE 0:GCOL 0,1
20 PROCsine(100,1100,500,450,4.25,45)

```
10 REM Program P-demo
20 REM Version B 1.0
30 REM Author Cliff Blake
40 REM BEEBUG June 1992
50 REM Program subject to copyright
60 :
100 *FX11
110 MODE 3:VDU 19,0,4,0,0,0
120 PRINT"ANGLE"
130 FOR x1=3.5 TO -3.5 STEP -3.5
140 FOR y1=3.5 TO -3.5 STEP -3.5
150 IF x1<>0 OR y1<>0 rads=FNangle(x1,
y1) ELSE 200
160 PRINT
170 PRINT TAB(10)"x = ";x1;TAB(22)"y =
";y1;
180 @%=&2020A
190 PRINT TAB(35);rads;" radians or
";DEG(rads);" degrees"
200 @%=&90A
210 NEXT y1:NEXT x1
220 PRINT""TAB(20)"Press any key to m
ove on each time."
230 g%=GET
240 :
250 MODE 0:GCOL 0,1
260 PRINT"CIRCLE"
270 PROCcircle (640,512,500,0)
280 PROCcircle (200,512,150,1)
290 PROCcircle (640,200,175,0)
300 PROCcircle (900,512,175,1)
310 PROCcircle (640,800,150,0)
320 g%=GET
330 :
340 MODE 0:GCOL 0,1
350 PRINT"ELLIPSE"
```

```
360 PROCellipse (640,512,500,250,45,0)
370 PROCellipse (200,512,150,75,90,1)
380 PROCellipse (640,200,175,50,-80,0)
390 PROCellipse (900,512,175,70,30,1)
400 PROCellipse (640,800,150,50,-30,0)
410 g%=GET
420 :
430 MODE 0:GCOL 0,1
440 PRINT"POLYGON"
450 PROCpolygon (6,640,512,500,45,0)
460 PROCpolygon (3,200,512,150,70,1)
470 PROCpolygon (4,640,200,175,20,0)
480 PROCpolygon (5,900,512,175,36,1)
490 PROCpolygon (5,900,512,175,0,0)
500 PROCpolygon (8,640,800,150,0,0)
510 g%=GET
520 :
530 MODE 0:GCOL 0,1
540 PRINT"HELIX"
550 PROCelix(500,500,700,600,5,50)
560 PROCelix(770,470,570,370,5,50)
570 PROCelix(100,900,1180,900,15,75)
580 PROCelix(1100,100,1100,800,10,160
)
590 g%=GET
600 :
610 MODE 0:GCOL 0,1
620 PRINT"SINE"
630 PROCsine(100,1100,500,450,2,0)
640 PROCsine(100,1100,500,350,2,90)
650 PROCsine(200,500,200,100,4,0)
660 PROCsine(200,500,200,100,4,180)
670 g%=GET
680 :
690 MODE 7
700 PRINT TAB(16,12)"THE END"
710 *FX12
720 VDU7:END
730 :
1000 REM *****
1010 REM Procedures/Functions
1020 REM start here.
1030 REM *****
1040 :
1050 REM Calculates unambiguous
1060 REM angle from X & Y
1070 :
1080 DEF FNangle(x,y)
1090 LOCAL rd
1100 IF x>0 AND y=0 =0
1110 IF x<0 AND y=0 =PI
1120 IF x=0 AND y>0 =PI/2
```

continued on page 58



VIRTUAL DISC DRIVES

Some time ago I saw a reference to certain BBC micros where more than four virtual drives were available, i.e. not just the normal two double-sided drives. I would be grateful on your advice on the possibilities of achieving this and the likely cost.

I should explain that I have a model B with Torch Z80 disc pack (a twin double sided 5.25" drive). When operating in Torch MCP mode (a CP/M lookalike), drives 0 & 2 become drive A, and drives 1 & 3 become drive B. I would like to add an additional 3.5" double-sided drive (to transfer data to or from this size of disc used by PCs). Any advice you can give would be very gratefully received.

E.L.de Bourcier

We are unable to help on this - if any readers can provide relevant information we will pass it on to Mr. de Bourcier.

BBC BASIC ON A PC

Is it possible to get BBC Basic on a disc and input this into a PC like other programs so that I can use some of the programs from BEEBUG on this machine?

B.F.Kane

A version of BBC Basic to run on PCs was produced by M-TEC Computer Services, The Market Place, Reepham, Norfolk NR10 4JJ. This was BBCBasic(86) which was reviewed in BEEBUG Vol.8 No.2 (in 512 Forum) when it cost £96.00 inc. VAT and p&p.

PRINTING CYRILLIC CHARACTERS

I would like to lay my hands on a way of printing out Cyrillic (Russian) characters. Is there anyone who can supply a suitable program, of a word processing nature?

My machine is a Master Compact, and the printer a Star Multi-font LC24-15, both of which are used more or less exclusively for word processing.

Michael Carter

We will pass on to Mr Carter any replies.

FINDING BBC MICRO ITEMS

I have noted with interest the signs that some owners of the trusty BBC B are a little worried about the back-up they can expect to get in the future, and I am sure your recent comments in BEEBUG will be most welcome (see Editorial, Vol.10 No.10).

I recently sought the Acornsoft ISO-Pascal for the Beeb. I had no success with my usual sources, but I did receive an offer in answer to a private advertisement, and I am now the happy owner of the required software. The gentleman in question also sent me a list of various computer items he had for sale. It seems to me that there is Beeb material available around the country, but how do we find it? Personal ads in BEEBUG are a great help, but too many advertisers are trying to sell complete outfits. Is there any chance of BEEBUG running a secondhand 'sideline' for BBC material?

C.R.Jones

The members' ads in BEEBUG can be used for both sales and wants, and comment from other readers (buyers and sellers) indicates that these are an effective way of selling unwanted items, and obtaining those things which are otherwise hard to find. Whilst many advertisers no doubt prefer to sell a complete system as such, I am sure that most would welcome enquiries for individual items.



```

1130 IF x=0 AND y<0 =3*PI/2
1140 IF ABS(x)/ABS(y)<1E-18 THEN x=SGN(
x)*ABS(y)*1E-18
1150 rd=ATN(y/x)
1160 IF x<0 =PI+rd
1170 IF y<0 =2*PI+rd
1180 =rd
1190 :
1200 REM Plots fast circle or disc
1210 :
1220 DEF PROCcircle (Xc,Yc,r,f%)
1230 LOCAL A,CA,SA,a,Ca,Sa,Cp,Sp,x,y
1240 a=2*PI/40
1250 Ca=COS(a):Sa=SIN(a)
1260 CA=1:SA=0
1270 MOVE r*CA+Xc,r*SA+Yc
1280 FOR A=1 TO 40
1290 Cp=CA:Sp=SA
1300 CA=Cp*Ca-Sp*Sa:SA=Sp*Ca+Cp*Sa
1310 x=r*CA+Xc:y=r*SA+Yc
1320 IF f%=0 DRAW x,y ELSE MOVE Xc,Yc:P
LOT 85,x,y
1330 NEXT A
1340 ENDPROC
1350 :
1360 REM Plots fast ellipse or platter
1370 :
1380 DEF PROCellipse(Xc,Yc,ma,mi,t,f%)
1390 LOCAL A,CA,SA,a,Ca,Sa,Cp,Sp,Ct,St,
x,y
1400 Ct=COS(RAD(t)):St=SIN(RAD(t))
1410 a=2*PI/40
1420 Ca=COS(a):Sa=SIN(a)
1430 CA=1:SA=0
1440 MOVE ma*Ct+Xc,ma*St+Yc
1450 FOR A=1 TO 40
1460 Cb=CA:Sb=SA
1470 CA=Cb*Ca-Sb*Sa:SA=Sb*Ca+Cb*Sa
1480 x=ma*Ct*CA-mi*St*SA+Xc:y=ma*St*CA+
mi*Ct*SA+Yc
1490 IF f%=0 DRAW x,y ELSE MOVE Xc,Yc:P
LOT 85,x,y
1500 NEXT A
1510 ENDPROC
1520 :
1530 REM Plots fast polygon
1540 :
1550 DEF PROCpolygon (s%,Xc,Yc,r,t,f%)
1560 LOCAL A,CA,SA,a,Ca,Sa,Cp,Sp,x,y
1570 a=2*PI/s%
1580 Ca=COS(a):Sa=SIN(a)
1590 CA=COS(RAD(t)):SA=SIN(RAD(t))
1600 MOVE r*CA+Xc,r*SA+Yc

```

```

1610 FOR A=1 TO s%
1620 Cp=CA:Sp=SA
1630 CA=Cp*Ca-Sp*Sa:SA=Sp*Ca+Cp*Sa
1640 x=r*CA+Xc:y=r*SA+Yc
1650 IF f%=0 DRAW x,y ELSE MOVE Xc,Yc:P
LOT 85,x,y
1660 NEXT A
1670 ENDPROC
1680 :
1690 REM Plots fast helix
1700 :
1710 DEF PROCChelix(Xs,Ys,Xf,Yf,L%,w)
1720 LOCAL Xd,Yd,Rd,Xc,Yc,Xi,Yi,Tn,end,
den,mi,A,CA,SA,a,Ca,Sa,Cp,Sp,Ct,St,x,y
1730 Xd=Xf-Xs:Yd=Yf-Ys:Rd=SQR(Xd*Xd+Yd*
Yd)
1740 Ct=Xd/Rd:St=Yd/Rd
1750 Tn=L%+.5:end=40*Tn
1760 den=40*(Tn+1)
1770 Xi=Xd/den:Yi=Yd/den
1780 mi=.6*Rd/(Tn+1)
1790 Xc=Xs+mi*Ct:Yc=Ys+mi*St
1800 a=2*PI/40
1810 Ca=COS(a):Sa=SIN(a)
1820 CA=-1:SA=0
1830 MOVE -mi*Ct+Xc,-mi*St+Yc
1840 FOR A=1 TO end
1850 Cp=CA:Sp=SA
1860 CA=Cp*Ca-Sp*Sa:SA=Sp*Ca+Cp*Sa
1870 x=mi*Ct*CA-w*St*SA+Xc:y=mi*St*CA+w
*Ct*SA+Yc
1880 DRAW x,y
1890 Xc=Xc+Xi:Yc=Yc+Yi
1900 NEXT A
1910 ENDPROC
1920 :
1930 REM Plots fast sinusoid
1940 :
1950 DEF PROCsine(Xs,Xf,Yz,Yp,cy,ph)
1960 LOCAL A,CA,SA,a,Ca,Sa,Cp,Sp,x,y
1970 MOVE Xs,Yz:DRAW Xf,Yz
1980 step=(Xf-Xs)/cy/40
1990 a=2*PI/40
2000 Ca=COS(a):Sa=SIN(a)
2010 CA=COS(RAD(ph)):SA=SIN(RAD(ph))
2020 MOVE Xs,Yp*SA+Yz
2030 FOR x=Xs+step TO Xf STEP step
2040 Cp=CA:Sp=SA
2050 CA=Cp*Ca-Sp*Sa:SA=Cp*Sa+Sp*Ca
2060 y=Yp*SA+Yz
2070 DRAW x,y
2080 NEXT x
2090 ENDPROC

```


Please keep sending in your hints on anything relevant to the BBC and Master computers. Don't forget, we pay for all hints we publish.

HARD SPACE AND RIVERS IN VIEW

Mr Toad

View has no *hard space*, i.e. a space at which two words cannot be split over the end of a line during formatting with right justification on. You can get around this by substituting an otherwise unused character such as '@', formatting the text, and then changing it to a space by going to the command screen and typing:

```
C/@/ /
```

just before printing.

Sometimes, right justified text on a word processor leaves ugly chains of spaces running down the text. Printers (the human variety, that is) call them *rivers*. In View, you usually find that you can get rid of these by moving the cursor to a line somewhere inside the paragraph and pressing f0 (Format Paragraph). Different cursor positions give different layouts when this key is pressed.

BEEBUG MAGAZINE DISCS AND THE MASTER

David Polak

The magazine disc is not available in 5.25" ADFS format, and Master users without 3.5" drives may not always wish to use the DFS. An easy way to transfer the programs is to make a backup of the BEEBUG magazine disc, format side 2 of the backup disc using the ADFS Formatter AForm on the Welcome disc, and then use CopyFiles (again off the Welcome disc) to copy all the files from DFS (side 0) to ADFS (side 2). Then you must build a !Boot file on side 2 (using *BUILD !Boot), containing the lines:

```
*BASIC
```

```
CHAIN "MENU"
```

Now you must change the !Boot file on the DFS side to read:

```
*FADFS
```

```
*MOUNT 2
```

```
*EXEC !Boot
```

Now Shift-D-Break will run the !Boot file on side 0 in DFS, which will then automatically switch to ADFS and will run the programs off side 2.

FOUR INTO ONE

Ajay Ramaswamy

In BEEBUG Vol.10 No.2 there was a hint from T.D.Parsons on programming 2 ROM images into one EPROM in a BBC Model B. I have improved on this somewhat. To get 4 ROM images into one EPROM, program a 27512 with the 4 images at 16K boundaries. Then bend up pins 1 and 27 on the EPROM and solder a wire link between pin 1 of the EPROM and pin 12 of IC76 (74LS163), and similarly connect pin 27 of the EPROM to pin 11 of IC76.

This works by switching the extra bits (2 and 3) of the ROMSEL latch (IC76 at &FE30) to the extra address lines of the EPROM. Since the 74LS163 can drive up to 10 TTL loads you can safely connect up to 4 modified EPROMs into your system to get a total of 16. If you put the modified EPROM into the socket for IC52 (logical ROM socket 12), the four ROM images will appear in positions 0,4,8 and 12.

NOTE: Although we can verify that this information is correct, any readers who undertake this action do so at their own risk. Ajay has performed this modification on about 30 computers, and it has been working reliably for more than two years.

ⓑ

Personal Ads

BEEBUG members may advertise unwanted computer hardware and software through personal ads (including 'wants') in BEEBUG. These are completely free of charge but please keep your ad as short as possible. Although we will try to include all ads received, we reserve the right to edit or reject any if necessary. Any ads which cannot be accommodated in one issue will be held over to the next, so please advise us if you do not wish us to do this. We will accept adverts for software, but prospective purchasers should ensure that they always receive original copies including documentation to avoid any abuse of this facility.

We also accept members' Business Ads at the rate of 40p per word (inclusive of VAT) and these will be featured separately. Please send all ads (personal and business) to MEMBERS' ADS, BEEBUG, 117 Hatfield Road, St. Albans, Herts AL1 4JS.

Archimedes 310, colour monitor, 4 slot backplane, 20Mb ST506 internal drive, external disc buffer, Oak SCSI interface, RX80 printer plus 50 discs of software £800, 20Mb ST506 drive £60, 2 slot backplane £15 (unused). Games; Nevryon, Dropship and Powerband £12 each, Missile control and jet fighter £6 each. Tel. (0775) 710640 eves or (0775) 761161 extn 4361 day.

BBC 512 co-processor & Essential's memory expansion, Original system discs & Essential's C.Lmouse driver, RAMdisc, F5Tboot & miscellaneous discs, also Tull mouse driver, Shibumi Problem Solver & Essential's CPFS ROM, complete with User Guides £200. Tel. 081-5437800.

BBC B Issue 7 OS 1.20, perfect condition (incl. all leads and manuals) £125 ono. GIS Advanced Teletext Adaptor incl. ATS ROM, also spotless £80. Tel. (0727) 41126 evenings.

BBC Hardware: Watford video digi, manuals & ROM inc. £65; Watford 32K sideways ROM card £35; Acorn Music 500 complete with operating software & tunes £30; Cub monitor with leads £165; Morley teletext adaptor inc. software & aerial £45; Watford User port splitter £13, AMX Art ROM software £12; Watford 40/80T 5.25" DS £60; Watford DumpOut 2 printer ROM £12; BBC Computer Issue 7 (broken - ideal for spare parts) inc. DFS 0.90 and Acorn speech synthesiser £80; Plenty of tape & disc software, over 200 games - £2 per tape, £3 per disc, discount for large orders; 5.25" blank discs 30p each; BEEBUG mags Vol.1-7 £4 per volume or £21 the lot. Tel. (0767) 680802.

WANTED: BBC B copy of FierBird's "Duck!" on either tape or (preferably) DFS disc. Tel. (0423) 500994.

WANTED: Electron Plus One advanced interface in good working order. Tel. (0423) 505372.

WANTED: APEX ROM/RAM board, made by Altair Electronics. Tel. 071-7276752.

Master 512, DFS and ADFS, PC emulator with GEM software and mouse, colour monitor, dual 40/80 disc drive. Many ROMs incl. Master ROM, InterWord, ViewSheet, The Publisher, complete OverView suite. Care quad ROM cartridge. More than 10 books, incl. Master Welcome guide, Reference Manuals 1 & 2, Dabs Master guide. £550. Tel. (0933) 678053

BBC B with Watford 32K RAM expansion board. Including Akhter 40/80 metal cased disc drives with mains PSU; Philips high res. mono monitor; Brother HR15 daisy wheel printer; InterWord ROM. Complete with all original manuals plus leads etc.; other BBC manuals plus general 6502 programming books at no extra cost. Will include Akai stereo tape deck for program storage if wanted. Buyer collects, £220. Tel. (0734) 320508.

WANTED: Computer Concepts InterSheet ROM complete with documentation for BBC B computer. Tel. (0753) 539609 evenings.

BBC 512 Co-processor in Acorn Universal box. Complete with all essential software, user/technical guides, £125. Tel. (0750) 20956.

Brother M1009 printer complete with NLQ font, manual, spare ribbons, BBC lead, £60 ono postage paid. Tel. (0829) 270176 evenings.

WANTED: Acornsoft LISP ROM for BBC. Spellmaster ROM by Computer Concepts. Tel. (0829) 270176 evenings.

BBC B Issue 7 with Watford MkII DDFS & 12 ROM board + sideways RAM £150. Acorn Master series RGB colour monitor £75. 40/80T disc drive £35. Canon PW-1080A printer £75. Data recorder £15. Pair of joysticks £8. Watford Turbo co-processor £35. ROMs (boxed as new): Watford MkI DDFS kit £25. Acorn ADFS, View 2.1, ViewSheet, ViewStore, DNFS, BEEBUG Toolkit Plus, Watford Dumpout 3, PrintMaster all £10; Logo £20; InterWord, InterSheet £15; Watford Wapping Editor plus mouse £50. Sensible offers for: Mini Office II, 'Image' Graphics, Clipart Library, View Printer Driver Generator, Calligraphy, Interactive 3d, Micro User 10 of the Best Graphics Utilities, RAM & Disc Utilities, Basic Utilities, Fonstyle, Diagnostics Disc, Sideways RAM utilities, Best of BEEBUG Applications I & II, ASTAAD CAD package, Repton 2 & 3, palace of Magic, Breakthrough, The Gold Collection, Summer Olympics, Cavern Quest, micro User 10 of the Best Games, Elite, Aviator, Revs, Revs 4 Track, Soccer Manager, Time Trucker, Word Mover, Jungle Maths, Wordspell, BEEBUG magazine Vol.1 No.1 to present, Advanced User Guide, 30 Hour Basic, Dabhand Guide to ViewSheet and ViewStore, DFS manual, ViewSheet manual, Understanding InterWord book, The

Epson FX/Kaga Printer Commands revealed, First Steps with the BBC micro. Tel. (0245) 321211.

M128 + internal modem, music 2000, 3000, 4000, 5000, various books, software and BEEBUG mags. Immaculate condition and BEEBUG mags. Vol.1 to 10 (1 to 9 bound) £11 per vol. All excellent condition, discs boxed with manuals, prices inc. postage. Tel. (0494) 562650 evenings.

For BBC B: Mini Office II 80T disc £6, MasterFile II DFS 80T disc £13, Advanced Disc User Guide (Pharo) £6, Assembly Language Programming for BBC micro (Birnbaum) £5, BEEBUG mags Vol.1 to 10 (1 to 9 bound) £11 per vol. All excellent condition, discs boxed with manuals, prices inc. postage. Tel. (0494) 562650 evenings.

A3 Plotmate Plotter with direct drive interface, for BBC micro or Master. Includes sideways ROM (with built-in Acorn GXR routines), manual, software, cable for User port. Bargain at £150 + carriage. Preferably collect (heavy). Tel. 081-681 3019 after 7pm.

BBC Issue 7 fitted with ADFS and Integra B board in a Viglan case with a separate keyboard £117. Tel. 081-318 5155.

Master 128 with OverView and all manuals including Reference Manuals parts 1 & 2, £175. 512 board with ES1Mb expansion, GEM software, mouse and manual, Dabs press user and technical guides £150. WE bridge 5.25/3.5" disc drives £75. ADT and UltraCalc ROMs with manuals £10 each. microview Cub monitor £50. Shinwa CP80 printer and spare ribbons £25. Numerous other books, magazines and discs. Tel. (0684) 563408.

Stolen: Standard Arc monitor and Arc mouse, and Star XB24-200 colour printer. If you are offered either of these please could you contact York Police on (0904) 631321 or myself at home (0904) 431336.

Acorn 80186 co-processor board. Offers? Tel. (0392) 406103 during office hours.

WANTED: 'Basic ROM User Guide for the BBC microcomputer and Acorn Electron' by Mark Plumley, published by Adder Publishing. Tel. 081-464 2447.

Epson LX800 printer, good condition, original packaging and manuals, spare ribbon £100. Tel. (0935) 77581 evenings.

Modem Pace Nightingale with CommStar II ROM £50. Hyperdriver ROM and disc, Pineapple Diagram (disc), Island Logic Music System (discs) all £15 each. BEEBUG Studio 8, Advanced Disc Toolkit, Wordwise Plus, Aviator all £10 each. Revs and Revs 4 Track £15; 4Mation Stretch, Speech!, ModemMaster comms software, all £5 each. All with relevant manuals and plus postage. Tel. (0527) 32613.

Master 512 with mouse; Opus high res monitor; Cumana switchable dual disc drives with PSU; twin stand; fitted ROMs (incl. manuals): InterWord, InterSheet, InterChart, InterBase, Wordwise Plus; four dual cartridges. Manuals: Welcome guide, View, ViewSheet, Reference guides 1 & 2, disc drive guide, 512 manual, Introduction to Wordwise Plus, Commsoft, Voyager. Discs (with manuals): The Account Book, ASTAAD CAD, Speech!, Gemini Office Mate, Gemini Office Master. Book: Assembly Language Programming for the BBC micro (Birnaum). BEEBUG mags April 1984 to current. Offers invited for whole, buyer collects. Tel. (0582) 410773.

Mexican built BBC B with View, Speech, NFS, 40/80T disc drive, Cub colour monitor, bound manuals, joystick as new £300. CST Procyon IEEE interface £15. 12" NEC mono monitor £30. A310 base unit and keyboard £375. Wordwise ROM £12, APTL ROM board £20, mouse as new £30. WANTED: StarWord ROM and Electron joystick interface software. Tel. (0483) 480632.

BEEBUG Master ROM, in original box £20. Morley teletext Adaptor for BBC or Master £55. Tel. (0253) 67987 after 7pm.

BBC Master 128 Turbo, Technomatic 40/80 twin drives in plinth, Mitsubishi medium res. colour monitor, Prism 2000 modem, cables, Master ROM, manuals, books, software includes Sharemaster and Fortell stockmarket programs, Pro-Punter, Brimardon, mini Office etc. Light home use. Cost over £1500, accept £500. Also Juki 5510 printer £80. Tel. 081-653 3895.

Music Master with microphone interface 5.25" disc, handbook. Mupados recorder tutor with Ensemble, Duet and Classroom Network packs (5 5.25" discs, handbooks and cassettes). Micro Maestro with 5.25" disc and 6 cassettes. All for £32 (worth £179). Tel. (0256) 27018.

WANTED: Universal 2nd co-processor (Ref:0233) for Master/512. Also wanted, book: KRYLOW, V.I. (tr. A.H.STROUD) "Approximate Calculations of Integrals" Macmillan, New York. Tel. (0376) 331760.

BBC B, tape recorder and lead, Citizen 120D 9-pin printer & lead, TV lead, many games on cassette and Mini Office II / All at £95. Tel. (0264) 790952.

WANTED: BBC B in any condition for spares. Tel. (0821) 642652.

Don't forget that
Labelmaster Plus
is still available for the Master 128 and Compact

The best labelling software for the BBC micro,
according to A&B Computing and the
Times Educational Supplement.

Packed with features, including full mailing list facility,
comprehensive control of printer styles and typefaces.
Extremely user-friendly and virtually foolproof.

Price: **£16.95 inc. VAT and p&p**
(add £3.00 for multi-colour version if you have a colour printer).

Rheingold Enterprises
17 Ingfield Terrace, Slaithwaite, Huddersfield HD7 5BJ
Telephone (0484) 846126 for full details

Study Electronics on the BBC Micro

An interactive approach to learning.
Three program titles now available. 'Introduction to Electronics Principles', 'Electronics Mathematics' and 'Digital Techniques'.
Programs include theory, examples, self test questions, formulae, charts and circuit diagrams.
User input and calculated outputs.

£29.95 each + £2 p&p.

Cheque or Postal Order to;
EPT Educational Software,
Pump House, Lockram Lane, Witham, Essex
CM8 2BJ

Please state BBC B/Master series and disc size.

**Wish something new was happening
for your BBC Micro, Master or
Electron?
Something is!**

Snacker - One of the latest batch of additions
to the catalogue, and probably the most
professional PD game yet!
Send £1.50 for catalogue and sampler disc to;

**BBC PD, 18 Carlton Close, Blackrod,
Bolton, BL6 5DL**

Make cheques payable to;
'A Blundell' or send an A5 s.a.e. for more details
(Please state disc size and format)

I have two Pets with a disc drive, printer, manuals and software in working order, and also two Apple IIe's with disc drives etc. Can anyone make use of them? Tel. 081-395 2346.

WANTED: Basic II. Also instructions for sideways RAM on Watford sideways ROM board. Tel. (0394) 385799.

Comminet Software, unused. CEC DataChat modem. Acorn Electron. Offers for lot or separate. Tel. (04243) 6213.

M128, 40/80T drive, games, mags, joystick, blank discs £250 plus carriage. Also Electron hardware and software details: SAE N.Tustin, 36 Cornmore, Pershore, Worcs WR10 1HX.

Graphics Digitisers by Minor Miracles for BBC B, complete with instructions and either tape or 40 track disc software (specify which). Recently tested and OK, £15 inc. p&p. Tape to disc utilities, transfers even locked programs, WE Executor and Clares Replica on 40 track discs, £3 each inc. p&p. Tel. (0821) 642652.

BBC B Issue 7 + sideways ROM /RAM board + 40/80 DD + books/discs £200. Amber monitor £30. Centronics printer £30. Acorn teletext adaptor £40. Home-brew EPROM burner £10. PET 32K + DD offers? Tel. (0245) 258368.

Master 128 MOS 3.5. ROMs: ADI, Commstar 2, DumpMaster, Help, Master ROM, ViewStore, Zoom & Smart cartridge. Discs: Kansas Finance, Viewchart and various games. Complete set of manuals. £300. Tel. 081-892 9278.

Master 128K, interWord, DumpOut 3, DFS, ADFS, View, ViewSheet, Edit, Reference Manuals parts 1 & 2 + Welcome manual £200. Tel. (0604) 21570.

WANTED: Mouse for Master Compact, CC SpellMaster ROM and manual. Tel. (0536) 723988.

Archimedes 310 computer, Acorn colour monitor, IPFL 2Mb memory board (upgradable to 4Mb), fitted with RISC OS 2 and MEMC1a, PC Emulator, Euclid, some games and PD. First class condition, original boxes. £700. Datachat 1223 modem with ROM and lead for BBC B/M128 £30. M128 OS ROM £15. Tel. 051-606 0289.

Watford CS4005 40/80 disc drive with PSU £50. ViewStore package in original box £15; ViewSpell package in original box £15. Tel. (0792) 865691.

Master software: System Delta ROM database and System Delta Reporter £30. Elite 3.5" (for Compact) £5.

Dabhand Guide to Master OS and 3.5" disc £5. Voltmace Delta 14b joystick for BBC £5. Lots of 5.25" disc and tape software going very cheap. Tel. (0358) 43389.

SPECIAL OFFERS TO BEEBUG MEMBERS

BEEBUG members can purchase BEEBUG and RISC Developments' products at special discounted prices. For additional offers see the centrefold in the magazine.

0077b	C - Stand Alone Generator	14.56	0084b	Command	29.88
0081b	Masterfile ADFS M128 80 T	16.86	0073b	Command(Hayes compatible)	29.88
0024b	Masterfile DFS 40 T	16.86	0053b	Dumpmaster II	23.76
0025b	Masterfile DFS 80 T	16.86	0004b	Exmon II	24.52
0074b	Beebug C 40 Track	45.21	0087b	Master ROM	29.88
0075b	Beebug C 80 Track	45.21	1421b	BEEBUG Binder	4.20

SUBSCRIPTION RATES

Send applications for membership renewals, membership queries and orders for back issues to the address below. All membership fees, including overseas, should be in pounds sterling drawn (for cheques) on a UK bank. Members may also

BEEBUG	BEEBUG & RISC USER	
£18.40	1 year UK, BFPO, Ch.I	£28.90
£27.50	Rest of Europe & Eire	£42.90
£33.50	Middle East	£53.10
£36.50	Americas & Africa	£58.40
£39.50	Elsewhere	£62.50

BACK ISSUE PRICES (per issue) 1 July 1991

All overseas items are sent airmail. We will accept official UK orders for subscriptions and back issues, but please note that there will be a £1 handling charge for orders under £10 which require an invoice. There is no VAT on magazines.

Volume	Magazine	5"Disc	3.5"Disc
6	£1.00	£3.00	£3.00
7	£1.10	£3.50	£3.50
8	£1.30	£4.00	£4.00
9	£1.60	£4.75	£4.75
10	£1.90	£4.75	£4.75

POST AND PACKING

Please add the cost of p&p when ordering. When ordering several items use the highest price code, plus half the price of each subsequent code.

Stock Code	UK, BFPO Ch.I	Europe, Eire	Americas, Africa, Mid East	Elsewhere
a	£ 1.00	£ 1.60	£ 2.40	£ 2.60
b	£ 2.00	£ 3.00	£ 5.00	£ 5.50

BEEBUG 117 Hatfield Road, St.Albans, Herts AL1 4JS
Tel. St.Albans (0727) 40303, FAX: (0727) 860263
 Manned Mon-Fri 9am-5pm (for orders only 9am-6pm and 9am-5pm Saturdays)
 (24hr Answerphone for Connect/Access/Visa orders and subscriptions)

CONTRIBUTING TO BEEBUG PROGRAMS AND ARTICLES

We are always seeking good quality articles and programs for publication in BEEBUG. All contributions used are paid for at up to £50 per page, but please give us warning of anything substantial that you intend to write. A leaflet 'Notes to Contributors' is available on receipt of an A5 (or larger) SAE.

Please submit your contributions on disc or cassette in machine readable form, using "View", "Wordwise" or other means, but please ensure an adequate written description is also included. If you use cassette, please include a backup copy at 300 baud.

In all communication, please quote your membership number.

RISC Developments Ltd (c) 1992

Printed by Arlon Printers (0923) 268328

ISSN - 0263 - 7561

BEEBUG MAGAZINE is produced by
 RISC Developments Ltd.

Editor: Mike Williams
 Assistant Editor: Kristina Lucas
 Technical Editor: Mark Moxon
 Editorial Assistance: Marshal Anderson
 Production Assistant: Sheila Stoneman
 Advertising: Sarah Shrive
 Subscriptions: Sue Baxter
 Managing Editor: Sheridan Williams

All rights reserved. No part of this publication may be reproduced without prior written permission of the Publisher. The Publisher cannot accept any responsibility whatsoever for errors in articles, programs, or advertisements published. The opinions expressed on the pages of this journal are those of the authors and do not necessarily represent those of the Publisher, RISC Developments Limited.

To celebrate the tenth anniversary of the founding of BEEBUG magazine, we have put together a selection of the best programs which we have published over the past ten years. This disc is packed with applications, utilities and games most of which have not been available previously other than when first published in BEEBUG. All the programs come with full on-screen help files, which can be printed out as well for permanent reference.

BEEBUG 10th Anniversary Disc

Celebrate BEEBUG's 10th anniversary with this disc at the special low price of £4.95 and you will have something to celebrate too.



DRAUGHTS

DRAUGHTS - An implementation of the classic board game in which you pit your wits against a computerised opponent.

KEYSTRIP DESIGNER - A very well written program to enable the creation, editing and printing of function key strips.

GARP - GARP (Geographical Atlas using Radial Projection) allows views of the globe to be displayed from any point above the Earth's surface.

MULTI-COLUMN PRINTING - This utility formats any text file into columns, and prints the result using an Epson FX-80 or compatible printer.

PERPETUAL CALENDAR - This program can display or print the calendar month by month for any year between 1753 and 5000 A.D. in the United Kingdom, or even earlier in other countries.

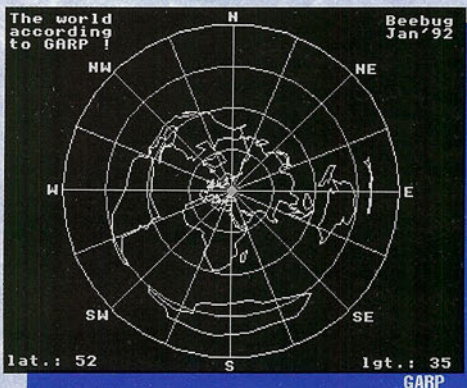
QUAD - Quad is a Tetris-like game, in which you must manipulate falling blocks to slot into each other. Dangerously addictive!

STEALTH - In this game you play against an opponent (or the computer), who sets a number of targets for you to find, and you must use your skill to discover the locations of the targets in as few goes as possible.

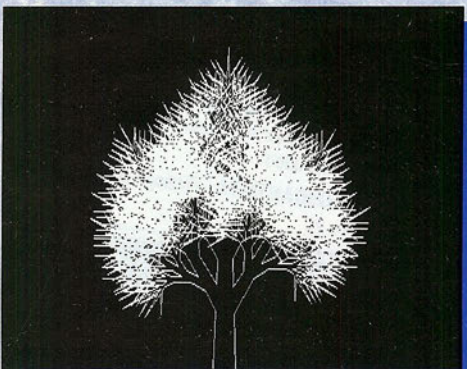
RECURSIVE TREES - This fascinating program uses recursion to create an infinite variety of tree-like designs - you choose a set of numbers, and the computer does the rest.

THE WORLD BY DAY AND NIGHT - This program will draw a map of the world showing graphically where the sun is in the sky or where it's night at every spot on earth.

CROSSWORD COMPILER - This program allows crosswords to be designed and the clues compiled.



GARP



RECURSIVE TREES

PBB3a 3.5" ADFS £4.95 inc. VAT + £1 p&p
PBB5a 5.25" DFS 40/80T £4.95 inc. VAT + £1 p&p