

Vol.10 No.3 July 1991

# BEEBUG

FOR THE  
BBC MICRO &  
MASTER SERIES



## Chinese Tangrams

- DISC RECOVERY UTILITY ● THREE-DIMENSIONAL GRAPHS
- PROGRAM DEVELOPMENT MADE EASY ● GAMES REVIEW

## FEATURES

Chinese Tangrams	7
BEEBUG Workshop: Three-Dimensional Graphs	11
Program Development Made Easy	15
Recreational Mathematics: The Mathematics of Encryption	22
Passing Information via Function Keys	27
DFS Disc Recovery Utility	28
BEEBUG Education	34
First Course: VDU and FX Calls (4)	37
Wordwise User's Notebook (2)	41
Printing Scientific Characters with Word Processors (3)	43
512 Forum	45

## REVIEWS

Games Review: Play It Again Sam 15	19
---------------------------------------	----

## REGULAR ITEMS

Editor's Jottings	4
News	5
Personal Ads	56
RISC User	58
Postbag	59
Hints and Tips	61
Subscriptions & Back Issues	62
Magazine Disc	63

## HINTS & TIPS

Coloured Disc Titles	
Saving Sideways RAM	
Using Watford Sideways RAM Boards	

## 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

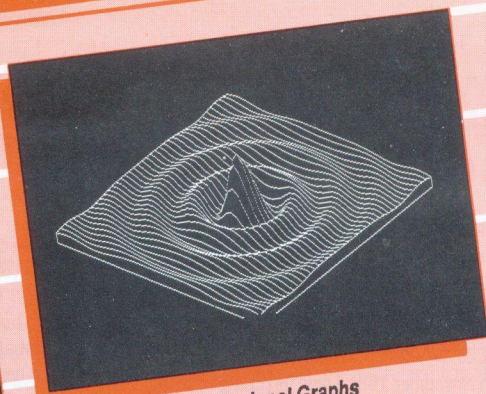
Picked:3 Fix with RTN

**Sianese Cat**

KEY CONTROL  
 Move = Z X ? \*  
 Turn = < and >  
 Fast = + SHIFT  
 Colour in = <0>  
 Empty out = SPC  
 Goto Menu = ESC

Construct Tangram here

### Chinese Tangrams



### Three-Dimensional Graphs

· USING ·  
 · HOLDING ·  
 · LIVES · 6 ·

YOU THEN 000000 00:00:33

LAST HURRAH

### Games Review

Year AD	WORLD WAR TWO	Event
1946		A Hundred Years of History
1945		World leaders in Lond
1944		Victory in Europe for
1943		D-Day Landings a Triu
1942		'Dambusters' raid on
1941		Monty on top in Afric
1940		Blitz blasts London
1939		Escape from Dunkirk
1938		War declared against
1937		Czechslovakia taken
1936		Hitler wants "Living
		King Edward VIII Abdi

Close Open Count Edit  
 Use ← or → then RETURN.

### BEEBUG Education

Simulation of the method of encryption of Rivest, Shamir and Adleman  
 Encipher or Decipher or Quit, E or D or Q? E

Two prime numbers are to be entered.  
 Their product must be less than INT(SQR(2^31-1)) = 46340,  
 and greater than 20^3 = 27000.

What is the value of the first prime factor of the modulus (e.g. 149)?  
 149

What is the value of the second prime factor of the modulus (e.g. 233)?  
 233

The modulus is 149 x 233 = 34717 and its Euler Number is 148 x 232 = 34336

The exponent must be relatively prime to the Euler number of the modulus, and  
 so it must not share any factors with 148 or 232.

What is the exponent (e.g., a prime, 26987)?  
 26987

Write a message in CAPITAL LETTERS (space, comma and full stop also allowed)  
 with not more than 135 characters. A.S. Use oppo key with:  
 NOW IS THE TIME FOR ALL GOOD MEN, AND WOMEN, TO COME TO THE AID OF THE PARTY.  
 NOW IS THE TIME

836098937183291853617522

### Recreational Mathematics

Tel. 0482 523854  
 Membership No. E0029253  
 Your ref. M/D4/09

15 Caphurst Road,  
 Colwyn Bay,  
 Clwyd,  
 LL22 6JP.  
 20th May 1991

Mr. M. Williams,  
 BEEBUG Limited,  
 117 Hatfield Road,  
 St. Albans,  
 Hertfordshire,  
 AL1 4JS.

Dear Mr Williams,  
WORDWISE RELATED ITEMS OF INTEREST

Thank you for your letter of 8th May. In my letter of  
 21st April 1991 I omitted to include the fact that once my  
 BOOT file had been run.

### Wordwise User's Notebook

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.

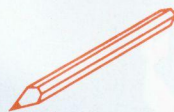


Program needs at least one bank of sideways RAM.



Program is for Master 128 and Compact only.

# Editor's Jottings



## REFLECTIONS

Back in November 1981 I was fortunate in being among a small group of privileged people who were invited to spend a weekend familiarising themselves with Acorn's then new BBC micro. This was as part of the BBC computer awareness campaign (backed up by local support around the country), and the start of the government's MEP (Microelectronics in Education Programme) scheme which was to last three years.

In a hotel, just outside Cambridge, one room was set aside and kitted out with what must have been some 30 BBC micros. None had power supplies built in - hence the wires radiating from the back of every machine - and of course discs were unheard of in those dim and distant days. What a marvel those machines appeared to be, as John Coll, author of the original User Guide, expounded on the many features of this new micro.

Given the climate of public opinion and understanding at that time, it is hardly surprising that I still remember that event as the dawning of a new era as far as computing was concerned.

And yet it all seems so far away now. Computers are commonplace today, in the home and in every sphere of human activity. Indeed it is difficult to think of any sphere of human activity which does not now depend on some form of computer. Even the smallest enterprise is now computerised (our local video shop and doctor's surgery to name but two).

One cannot attribute all the changes that have taken place since the Beeb first appeared to that machine alone, but it does occupy, in my view, a significant place in the history of computing in this country.

It has also lasted remarkably well, and there are many tens of thousands of BBC micros still doing

yeoman service up and down the country. Look at the news programmes on television, and it is surprising how often a BBC micro (often still a model B) appears in camera.

However, there are now many faster, more sophisticated systems, not least Acorn's own Archimedes range, to tempt the computer enthusiast. On the other hand, many users are content with the system they have, and see no reason to change when their existing machine satisfies all their needs.

BEEBUG has always been dedicated to supporting users of the BBC micro and later Master series, and that is still the case today. Whereas users once had a choice of several magazines, the glossier offerings are steadily turning their attention to the Archimedes, and in one case already, caters exclusively for that machine.

BEEBUG is the only magazine catering solely for BBC micro users, and will continue as long as there is sufficient demand. The loyalty of *all* our readers is vital if we are to continue this support. And the more BEEBUG members can contribute (articles, letters, programs, reviews, or just your comments), the better and more lively I am sure the magazine will be.

## BINDERS OFFER

As a result of a recent purchase we are able to offer BEEBUG binders at a considerably reduced price - see leaflet enclosed with this issue.

## NEXT ISSUE

This is just a reminder that the next issue of BEEBUG will be for the two months of August and September, and mailout is expected mid-August.

M.W.

### SHOWS FOR ACORN USERS

Still some way off, but the *BBC Acorn User Show* looks like being the premier event of the year as far as BBC micro owners are concerned. The show is scheduled for 11th to 13th October 1991 at the Wembley Conference Centre (Thames Suite). This venue offers more space than last year, and has massive on-site parking. The show is supported by both BBC Acorn User and Acorn Computers Ltd., and BEEBUG is among the many companies who have already booked for the show. In fact, we shall have three back-to-back stands, one for BEEBUG and RISC User magazines (and associated products), one for our growing range of hardware and software, and one for our extensive retail business.

The exhibition is organised by Safesell Ltd., Market House, Cross Road, Tadworth, Surrey KT20 5SR, tel. (0737) 814084.

### LOW COST UNIX FROM ACORN

BBC micro users thinking of upgrading to an Archimedes can now buy a low cost Unix system from Acorn to run on a 400-series Archimedes system. RISC iX version 1.2 includes the latest versions of C, Fortran and Pascal compilers, the X Window System (Version 11 Release 4) and other supporting software. New installations start at £999. Acorn claims that this release allows it to offer the lowest cost Unix platform in the marketplace. For more information contact Acorn Computers Ltd. at Fulbourn Road, Cherry Hinton, Cambridge CB1 4JN, tel (0223) 245200.

### BEEBS DOWN UNDER

BEEBUG readers may be interested to learn about *OZBEEB*, the Acorn/BBC user group of Sydney, Australia. The group, active since 1984, meets twice monthly. Last year *OZBEEB* introduced a fully functional Bulletin Board system called *OzWorld BBS*. Since its launch in

March 1990 its facilities have been expanded through FidoNet, SIGnet, K12 and by other means, allowing members to communicate with other users around the world. *OZBEEB*'s only regret is that its BBS runs on a PC compatible (through lack of suitable software for an Acorn system). For those who may be interested, *OzWorld BBS* supports V22(1200), V23 (1200/75), V22bis (2400), and PEP (19,200) baud rates, with a word format of 8 data bits, no parity and 1 stop bit (8N1). The telephone number is (+612) 891 1886, and the board operates 24 hours per day.

For more traditional methods of communication, *OZBEEB* is at P.O.Box 1030, Parramatta, NSW 2150, Australia, tel. (+2) 635 4868 (day), (+74) 54 3413 (evenings) - their time!

### BACK TO THE CHALK FACE

Educational software house, Chalksoft, has announced the availability of its new software catalogue. Chalksoft is expanding its range of Archimedes compatible software, but still boasts a varied range of educational software for the BBC micro and Master series, covering a wide variety of curriculum areas. For a copy of the new catalogue write to Chalksoft Ltd., P.O.Box 49, Spalding, Lincs. PE11 1NZ, or telephone (0775) 769518.

### PUBLIC KEY CRYPTOGRAPHY

*The Public Key* is a magazine specialising in the subject of public key cryptography (see this month's article in the *Recreational Mathematics* series for more information on the RSA method of encryption). Volume 1 No.2 has recently been published and costs just £1.50 (£2.50 for other EEC countries, and £3.50 for overseas airmail). Issue 1 explaining the principles of public key cryptography is also available for £1 (£2 or £3). For more information, contact The Editor, 'Waterfall', Uvedale Road, Oxted, Surrey RH8 0EW, tel. (0883) 712440.

**BBC**  
**ACORN USER**  
**SHOW '91**

WEMBLEY CONFERENCE CENTRE - LONDON - 11th to 13th OCTOBER 1991

**THE SHOW**  
**FOR ALL ACORN USERS**

ARCHIMEDES - A3000 - BBC - MASTER - ELECTRON

**\* DON'T MISS \***

**\* OVER 60 LEADING EXHIBITORS \***

- \* LATEST SOFTWARE & HARDWARE**
- \* ACORN COMPUTERS FEATURE STAND**
- \* COMPUTER PROBLEM 'CLINICS'**
- \* FABULOUS GAMES ARCADE**
- \* CELEBRITY VISITS**
- \* SCHOOLS PROJECTS**
- \* INFORMATIVE & ENTERTAINING FEATURES**
- \* BBC ACORN USER MAGAZINE EDITOR'S OFFICE**
- \* MUCH - MUCH - MORE \***

**ADVANCE DISCOUNT TICKET APPLICATION BBC ACORN USER SHOW '91**

TICKETS AT DOOR - ADULTS £6.00 - UNDER 16's £4.00 - FAMILY TICKETS £16.00 - UNDER 5's FREE

ADULTS £5.00 NAME .....

UNDER 16's £3.00 ADDRESS .....

FAMILY TICKETS £13.00 .....

(2 Adults & 2 Children) .....

..... POSTCODE .....

PLEASE SEND ME ..... ADULT ..... UNDER 16's ..... FAMILY TICKETS

I ENCLOSE A CHEQUE/POSTAL ORDER FOR £..... (MADE OUT TO SAFESELL EXHIBITIONS LTD)

SEND THIS FORM TO ; SAFESELL EXHIBITIONS LTD, MARKET HOUSE, CROSS ROAD,  
TADWORTH, SURREY, KT20 5SR. SHOW INFORMATION WILL BE SENT WITH TICKETS.

# Chinese Tangrams

*We present a program by Jim Proctor based on the ancient Chinese Tangrams.*

Tangrams are designs which are made by assembling a set of seven shapes - five triangles of various sizes, one square and one rhombus. Using just these seven shapes, an enormous variety of designs can be achieved. These can represent people, animals, domestic objects and so on.

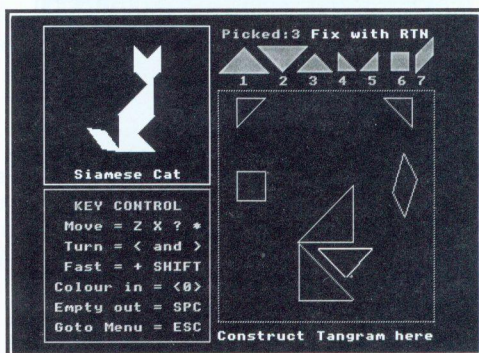
The program listed here allows you to create these designs on the screen from the set of seven shapes, which can be moved around using the keyboard until they are positioned where you want them. Twelve designs are built into the program for you to copy, and there is also the facility to create your own designs.

Type in listing 1 carefully and save it as *Tan1*. This program simply sets up the introductory screens which give you some information about the program. Next, type in listing 2 and save it as *Tan2*. This is the main program, which is automatically chained by *Tan1*.

Using the program is very simple, with on-screen help to guide you. The main screen offers a choice of the 12 built-in designs for you to emulate; you may either select one of these or choose to create your own design.

Choosing a design displays a further screen, showing the chosen design on the left, and the 7 shapes in a box on the right. Each of the shapes can be selected in turn by pressing keys 1-7. Once a shape has been selected, its colour changes from yellow to white. It can then

be moved around the screen using the Z and X keys for horizontal movement, or the \* and ? keys for vertical. The shape can also be rotated by using the < and > keys. When you are satisfied with its position, just press Return to fix it in place (it can always be selected and moved again later). Continue until you have positioned all the pieces. At any time, you can press 0 (zero) to fill the shapes with colour, or Space to empty them again, while Escape returns you to the main screen.

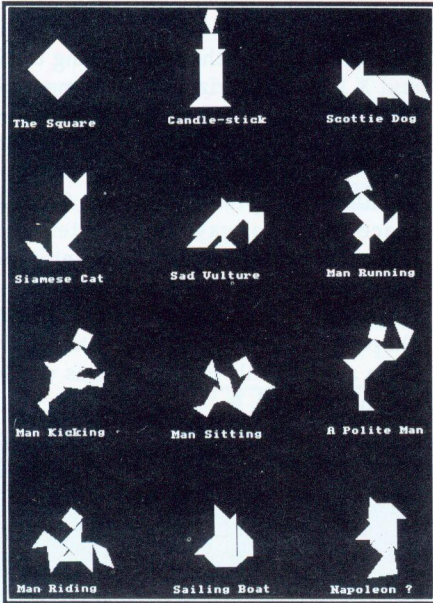


*Using the program to emulate one of the built-in designs*

Creating a design is very similar, except that in this case, there is no master design shown, and you simply move the shapes to where you want them. When you have finished and coloured the shape in, the design data will be displayed in the left-hand box, as a set of 7 rows containing three figures each, for x-pos, y-pos and angle. If you want to incorporate your shape into the program as one of the defaults, you should replace the data in one of the data lines from 3040-3150 with the figures given, reading from left to right, top to bottom,

# Chinese Tangrams

and also alter the name of the design in line 3000.



*Some examples of tangram designs*

## Listing 1

```

10 REM          >Tan1
20 REM Program Tangram Intro
30 REM Author   Jim Proctor
40 REM Version B 1.0
50 REM BEEBUG   July 1991
60 REM Program subject to copyright
70 :
100 MODE6:M%=1:P%=2
110 REPEAT:PROCIntro(M%)
120 M%=M%+1:UNTILM%>P%
130 CHAIN"Tan2"
140 END
150 :
1000 DEFPROCIntro(C%)
1010 C%=3*(1+C%MOD2):VDU19,1,C%;0;
1020 READT$:PRINTTAB((40-LENT$)/2,2)T$
1030 N%=1:end=FALSE
1040 PRINT':REPEAT:READI$
1050 IFI$<>"@PRINTTAB((40-LENI$)/2)I$:
ELSEend=TRUE
1060 SOUND1,-7,150+4*N%,1

```

```

1070 N%=N%+1:UNTILEnd=TRUE
1080 IFM%=P% S$="to start":ELSE S$="for
more"
1090 PRINT'TAB(9)"Press space "S$;
1100 G=GET:REPEATUNTILG=32:CLS
1110 ENDPROC
1120 :
1130 DATA "ALL ABOUT TANGRAMS"
1140 DATA "The making of designs (Tangr
ams) by","assembling seven shapes (5 tri
angles","1 square and 1 rhombus) is a pa
stime","that comes to us from Ancient Ch
ina. ","Using only the 7 shapes - known
as Tans"
1150 DATA "and traditionally made out o
f wood","many hundreds of designs are p
ossible. ","These represent persons, anim
als and a","variety of domestic objects.
","The purest design is a perfect squar
e!","@"
1160 DATA "ATTEMPTING THE ART OF THE TA
NGRAM"
1170 DATA "In this computer simulation,
you have","the choice of copying 12 exi
sting","designs listed in the Menu or tr
ying","to create your own original Tangr
am. ","If you elect to make your own des
ign"
1180 DATA "then on completion, you will
be given","the essential data for the T
angram you","have created. This gives yo
u the chance","to incorporate it in the
program","replacing an existing design."
,
1190 DATA "You will find that there are
clear","indications on the working scre
en of","the function of all keys to be u
sed. ","@"

```

## Listing 2

```

10 REM          >Tan2
20 REM Program Tangrams
30 REM Version B 1.0
40 REM Author   Jim Proctor
50 REM BEEBUG   July 1991
60 REM Program subject to copyright
70 :
100 ON ERROR IF ERR=17 RUN:ELSE PROCer
r:END
110 IF PAGE>&E00 PROCpage:END

```



```

120 *FX3,0
130 MODE1
140 PROCinit
150 PROCmenu
160 PROCstart
170 REPEAT
180 PROCselect
190 IF I%<0 I%=0
200 UNTIL FALSE
210 :
1000 DEF PROCget(lg%,ug%)
1010 *FX21,0
1020 REPEAT
1030 *FX202,32
1040 G%=GET
1050 UNTIL G%>lg% AND G%<ug%
1060 ENDPROC
1070 :
1080 DEF PROCmenu
1090 COLOUR 1
1100 PRINTTAB(8,1)"THE ART OF THE TANGRAM"
1110 VDU5
1120 PROCTans(-304,736)
1130 VDU4:COLOUR 1
1140 PRINTTAB(0,8)" Assemble the 7 Tans shown, using only"" edge and point contact (no overlap) to"" form any of the Tangrams listed below. "" Or create your own individual Tangram!"
1150 FOR M%=1 TO 12
1160 COLOUR 3
1170 PRINTTAB(22-17*(M%MOD2),15+2*((M%+1)/DIV2))CHR$(64+M%)";
1180 COLOUR 2
1190 PRINTn$(M%)
1200 NEXT:VDU7
1210 COLOUR 3
1220 PRINT"" Select <A-L> or <M> to make your own";
1230 PROCget(64,78)
1240 own=FALSE:M%=G%-64
1250 IF M%=13 own=TRUE
1260 CLS:ENDPROC
1270 :
1280 DEF PROCselect
1290 PROCget(47,56)
1300 REPEAT
1310 PROCend
1320 UNTIL G%<>48

```

```

1330 N%=G%-48
1340 VDU4,23,1,0;0;0;0;
1350 COLOUR 2
1360 PRINTTAB(18,1)SPC21TAB(18,1)"Picard:";N%;
1370 COLOUR 3
1380 PRINT" Fix with RTN":VDU5
1390 SOUND 1,-9,N%*12+41,4
1400 PROCplace:I%=0
1410 GCOL 0,0:PROCPick(1)
1420 REPEAT
1430 GCOL 3,3:PROCPick(1)
1440 REPEAT UNTIL INKEY-98 OR INKEY-67 OR INKEY-105 OR INKEY-73 OR INKEY-103 OR INKEY-104 OR INKEY-74
1450 GCOL 3,3:PROCPick(1)
1460 PROCkeymove
1470 UNTIL I%<0
1480 x%(N%)=x%:y%(N%)=y%
1490 a(N%)=a:sc(N%)=sc
1500 c%=sc(N%)*COS(a(N%))
1510 s%=sc*N%*SIN(a(N%))
1520 PROCset
1530 ENDPROC
1540 :
1550 DEF PROCkeymove
1560 IF INKEY-74 I%=N%:VDU4,23,1,0;0;0;0;:COLOUR 3:PRINTTAB(18,1)"Key 1-7 to select Tan":VDU5,7
1570 IF INKEY-1 d%=32:da=RAD22.5 ELSE d%=4:da=RAD1
1580 IF INKEY-67 AND x%<1196 x%=x%+d% ELSE IF INKEY-98 AND x%>632 x%=x%-d%
1590 IF INKEY-73 AND y%<704 y%=y%+d% ELSE IF INKEY-105 AND y%>96 y%=y%-d%
1600 IF INKEY-103 a=a+da ELSE IF INKEY-104 a=a-da
1610 c%=sc*COS(a):s%=sc*SIN(a)
1620 ENDPROC
1630 :
1640 DEF PROCPick(p%)
1650 IF N%<6 PROCtri(x%,y%,p%)
1660 IF N%=6 PROCsqu(p%)
1670 IF N%=7 PROCrhomb(p%)
1680 ENDPROC
1690 :
1700 DEF PROCfillmt(col%)
1710 FOR N%=1 TO 7
1720 PROCplace
1730 GCOL 0,col%

```

# Chinese Tangrams

```
1740 PROCpick(81)
1750 IF own=TRUE PROCown
1760 NEXT:ENDPROC
1770 :
1780 DEF PROCown
1790 VDU4
1800 IF col% COLOUR 1:PRINTTAB(1,8+N%);
: @%=5:PRINTx%,y%,INT( DEG (a) )MOD360:@%=8:
ELSE PRINTTAB(1,8+N%) SPC(15)
1810 VDU5:ENDPROC
1820 :
1830 DEF PROCtans(ax%,ay%)
1840 RESTORE 3010:READ n$
1850 FOR N%=1 TO 7
1860 READ x%(N%),y%(N%),a(N%)
1870 sc(N%)=sc(N%)/1.6
1880 x%=x%(N%)+ax%:y%=y%(N%)+ay%
1890 a=RAD(a(N%)):sc=sc(N%)
1900 c%=sc*COS(a):s%=sc*SIN(a)
1910 GCOL 0,2
1920 PROCpick(81)
1930 sc(N%)=sc(N%)*1.6
1940 NEXT:ENDPROC
1950 :
1960 DEF PROCstart
1970 VDU5:GCOL 0,1
1980 MOVE 0,500:DRAW 536,500:DRAW 536,1
000:DRAW 0,1000:DRAW 0,500
1990 GCOL 0,2
2000 MOVE 564,64:PLOT 21,1264,64:PLOT 2
1,1264,796:PLOT 21,564,796:PLOT 21,564,6
4
2010 MOVE 0,0:DRAW 536,0:DRAW 536,484:D
RAW 0,484:DRAW 0,0
2020 IF own=FALSE PROCpictan ELSE VDU4:
COLOUR 1:PRINTTAB(1,2)" YOUR TANGRAM"TAB
(1,4)"Once completed,"TAB(1,5)"the desig
n data"TAB(1,6)"is set out as:"TAB(1,7)"
Xpos,Ypos,Angle"
2030 VDU5:PROCTans(0,768)
2040 MOVE 632,840:PRINTn$
2050 RESTORE 3020:READ n$
2060 FOR I%=1 TO 7
2070 READ x%(I%),y%(I%),a(I%)
2080 x%(I%)=x%(I%)+672
2090 y%(I%)=y%(I%)+224
2100 a(I%)=RAD(a(I%))
2110 PROCset
2120 NEXT
2130 GCOL 0,3
```

```
2140 MOVE 560,32:PRINTn$
2150 VDU4,23,1,0;0;0;0;
2160 COLOUR 3
2170 PRINTTAB(18,1)"Key 1-7 to select T
an"
2180 COLOUR 2
2190 PRINTTAB(3,18)"KEY CONTROL"TAB(1,2
0)" Move = Z X ? *"TAB(1,22)" Turn = < a
nd >"TAB(1,24)" Fast = + SHIFT"TAB(1,26)
"Colour in = <0>"TAB(1,28)"Empty out = S
PC"TAB(1,30)"Goto Menu = ESC"
2200 VDU5,7
2210 ENDPROC
2220 :
2230 DEF PROCpictan
2240 RESTORE(3030+10*M%)
2250 FOR N%=1 TO 7
2260 READ x%(N%),y%(N%),a(N%)
2270 x%=x%(N%)/1.6+224
2280 y%=y%(N%)/1.6+640
2290 a=RAD(a(N%)):sc=sc(N%)/1.6
2300 c%=sc*COS(a):s%=sc*SIN(a)
2310 GCOL 0,1
2320 PROCpick(81)
2330 NEXT
2340 MOVE 96,544:PRINTn$(M%)
2350 ENDPROC
2360 :
2370 DEF PROCplace
2380 x%=x%(N%):y%=y%(N%)
2390 a=(N%):sc=sc(N%)
2400 c%=sc*COS(a):s%=sc*SIN(a)
2410 ENDPROC
2420 :
2430 DEF PROCset
2440 N%=I%
2450 PROCplace
2460 GCOL 0,2
2470 PROCpick(1)
2480 ENDPROC
2490 :
2500 DEF PROCtri(x%,y%,p%)
2510 MOVE x%,y%
2520 PLOT 0,-s%,c%
2530 PLOT p%,-c%+s%,-s%-c%
2540 PLOT p%,2*c%,2*s%
2550 PLOT p%,-s%-c%,c%-s%
2560 ENDPROC
2570 :
```

*Continued on page 52*

# Three-Dimensional Graphs

by Bernard Hill

Paper and flat screens are very good for illustrating normal  $(x,y)$  graphs, where  $y$  and  $x$  have a fixed relationship between them (often of the form  $y=f(x)$ ) which gives some sort of curve on the screen. We have more of a problem visualising a function of two variables:  $z=f(x,y)$ . Here we take the  $x$  and  $y$  axes to define between them a plane, and the value which  $z$  has at a particular point is given by the height above that plane (see figure 1).

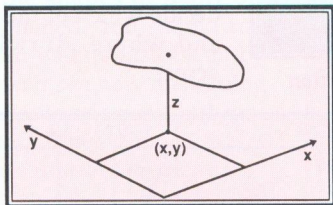


Figure 1

Representing the surface which this gives, however, is more of a problem on a computer screen, and this month's article and program is based on two quite distinct ways of achieving this.

Listing 1 contains, from line 1000 onwards, a function  $FNz(x,y)$  which can be changed by the

user. The program first calls the procedure `PROClimit` which asks the user for the range of  $x$  and  $y$  values, and reports on the range of  $z$  values likely to be found. It also requests the type of picture to be drawn, which we will deal with next.

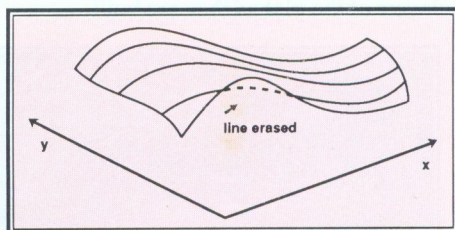


Figure 2

## A 3D PERSPECTIVE VIEW

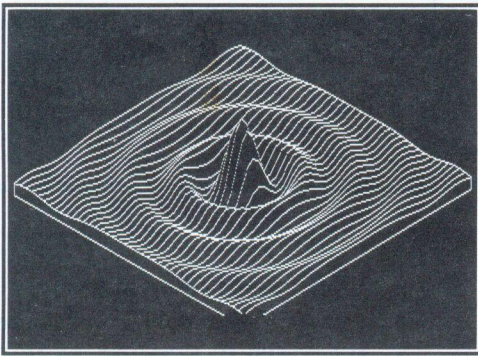
This type of presentation has been addressed in previous BEEBUG articles (Vol.2 No.5 and Vol.3 No.8) but none of these gives a general-purpose routine for drawing a user-defined function.

The problem of hidden-line elimination in a perspective view is fairly easy to solve if we draw the picture from the back, and erase as we draw forward. In figure 2 we have drawn a selection of curves in the  $x$  direction and are moving forward in the  $y$ . When one of the new curves rises over a previous one then it is necessary simply to erase all points on the screen lying underneath the new curve.

This is the function of the procedure `PROCjoin3D` in Listing 1. It draws lines joining the screen co-ordinates  $(oldx,oldy)$  to  $(px,py)$  which have been

## Workshop - Three Dimensional Graphs

previously calculated. It does this by placing dots on the screen along the required line, and erasing all points underneath each spot by drawing a line in background to the bottom of the screen. The rest of the procedure PROCplot3D does the necessary arithmetic to transform the coordinates the user enters to those on the screen. It is interesting not only to see the final picture but to see the layers of the plot as they are built.



3D perspective view of built-in function

### CONTOUR PLOTS

A second representation of a 3D surface is given by a contour map. This type of picture is also possible with listing 1, and is produced by PROCcont. If this option is selected the user can enter the spacing of the z-heights which will be used for contouring. These will appear in white or yellow (depending on whether they are above or below the mid-height) and four other contours will appear between these in red.

In order to draw the contours, the x-y plane is divided into squares, and the

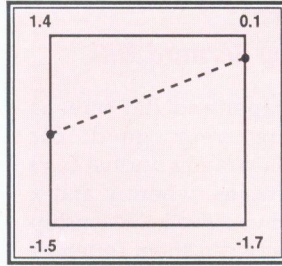


Figure 3. Contour across a square for  $z=0$

to assess the path of contour  $z=0$ . From the z-values (heights) at the corners we could guess that this probably intersects the left-hand side around the mid-point, and the right-hand side at a point near the top. In this way we draw a straight line across the square to approximate the contour, and by looking at the values of z at each corner of each of a number of small squares, we can thus decide which of the sides it intersects and where, drawing the contour across.

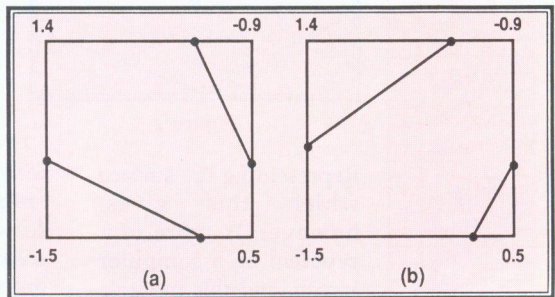


Figure 4. Possible contours

But a problem comes if the contour intersects more than two of the sides (usually all 4). In this case we could have contours as shown in figures 4a and 4b and we have no way of knowing which is more accurate. The solution is to re-contour the square with a finer grid but

## Workshop - Three Dimensional Graphs

in the end our contours are only a guess anyway. In the program the shortest of the two contours is assumed to be the correct one, but this is only correct in half the cases. It is possible to enhance this section of the program to produce an automatic subdivision of the square to cover ambiguous cases, but the program runs fairly slowly anyway, and ideally contouring should be done with a fine grid in the first place.

Should you select the *Contour Map* option of the program then you will be given a coarse grid of the map which is then recontoured at two successively finer scales (lines 1420-1430).

### CONCLUSION

This is a very simplistic introduction to the problems of 3D plotting, but enough is present in this program to enable a good grasp of the shapes of many 3D surfaces to be obtained. You should run the given program with limits of x and y from -20 to +20 (and see how misleading the first coarse contouring is!) but you could also try the following functions (at line 1000), or any variations on them:

```
DEF FNz(x,y)=x*x+y*y (a basin)
DEF FNz(x,y)=SINx*COSy (bumps and hollows)
DEF FNz(x,y)=x*x*EXP(-x*x-y*y) (twin humps)
or:
DEF FNz(x,y)=FNlim(1/((x-PI)^2+(y-PI)^2))
with:
DEF FNlim(a):IF a>10 THEN =10 ELSE =a
(a tree stump)
```

Editor's Note: This represents the last workshop, for the time being at least, contributed by Bernard Hill. I am sure readers would wish to join me in thanking Bernard for writing such a stimulating series of articles over many

issues of BEEBUG. We hope that the Workshop series will continue under a new author: any readers who would like to contribute to this series, on either an occasional or a regular basis, should contact the editor as soon as possible.

```
10 REM Program Graph3D
20 REM Version B1.1
30 REM Author Bernard Hill
40 REM BEEBUG July 1991
50 REM Program subject to copyright
60 :
100 ON ERROR MODE 7:PROCerror
110 MODE7
120 REM lines on 3D plot, view angle
130 n=40:th=30
140 sin=SINRADth:cos=COSRADth
150 PROClimits
160 IF Ctr THEN MODE1 ELSE MODE0
170 PROCcursor(FALSE)
180 IF Ctr THEN PROCcont ELSE PROCplot
3D
190 REM insert any screen dump here
200 PROCcursor(TRUE):END
210 :
1000 DEF FNz(x,y)
1010 LOCAL z:=SQR(x*x+y*y)
1020 IF z=0 THEN =1 ELSE =SINz/z
1030 :
1040 DEF PROCcursor(x)
1050 VDU23;11,x;0;0;0:ENDPROC
1060 :
1070 DEF PROCerror:IF ERR=17 END
1080 REPORT:PRINT" at line ";ERL:END
1090 :
1100 DEF PROClimits
1110 PRINTTAB(5,10)"Contour or 3D view
(C/3) :";
1120 REPEAT x$=GET$:C=INSTR("Cc3",x$)
1130 UNTIL C>0:Ctr=C<3
1140 VDU12,31,10,22,132,157,131
1150 IF Ctr THEN PRINT"CONTOUR MAP ";
ELSE PRINT"3D VIEW ";
1160 PRINTCHR$156
1170 PRINTTAB(5,20):INPUT "X min:"x0
```

## Workshop - Three Dimensional Graphs

```
1180 PRINTTAB(30,20);:INPUT "X max:"x1
1190 PRINTTAB(0,19);:INPUT"Y min:"y0
1200 PRINTTAB(0,1);:INPUT"Y max:"y1
1210 IF y0=y1 OR x0=x1 THEN PRINTTAB(0,
23)"Invalid limits":END
1220 PRINTTAB(3,5);CHR$136;"Evaluating
Z limits - please wait";
1230 max=-1E36:min=-max
1240 FOR x=x0 TO x1 STEP (x1-x0)/10
1250 FOR y=y0 TO y1 STEP (y1-y0)/10
1260 z=FNz(x,y)
1270 IF z>max THEN max=z
1280 IF z<min THEN min=z
1290 NEXT y:NEXT x
1300 zmid=(max+min)/2
1310 PRINTTAB(3,5)SPC34
1320 PRINTTAB(5,14);"Max z is : "max
1330 PRINTTAB(5,12);"Min z is : "min
1340 IF max=min THEN PRINTTAB(0,23)"Not
hing to draw":END
1350 PRINTTAB(5,16);
1360 IF Ctr THEN INPUT"Major Z contour
interval:"zinc5:zinc=zinc5/5 ELSE PRINT"
Press any key to continue";:IF GET
1370 ENDPROC
1380 :
1390 DEF PROCcont
1400 xscale=(x1-x0)/1280
1410 yscale=(y1-y0)/1024
1420 sq=128:REPEAT:PROCdisplay(sq)
1430 sq=sq DIV 2:UNTIL sq=16
1440 ENDPROC
1450 :
1460 DEF FNbelow(a,s)
1470 IF a<0 THEN =FNabove(-a,s)
1480 =s*INT(a/s)
1490 :
1500 DEF FNabove(a,s)
1510 IF a<0 THEN =FNbelow(-a,s)
1520 IF a/s=INT(a/s) THEN =a
1530 =s*(INT(a/s)+1)
1540 :
1550 DEF FNmax2(a,b)
1560 IF a<b THEN =b ELSE =a
1570 DEFFNmax(a,b,c,d)
1580 =FNmax2(FNmax2(a,b),FNmax2(c,d))
```

```
1590 DEFFNmin2(a,b)=-FNmax2(-a,-b)
1600 DEFFNmin(a,b,c,d)
1610 =-FNmax(-a,-b,-c,-d)
1620 :
1630 DEF PROCdisplay(size)
1640 FOR yo=1024-size TO 0 STEP -size
1650 FOR xo=0 TO 1280-size STEP size
1660 VDU24,xo,yo,xo+size-1,yo+size-1;16
1670 PROCsquare(xo,yo,size)
1680 NEXT xo:NEXT yo
1690 ENDPROC
1700 :
1710 DEF PROCsquare(x,y,s)
1720 xa=FNx(x):ya=FNy(y)
1730 xb=FNx(x+s):yb=FNy(y+s)
1740 z0=FNz(xa,ya):z10=FNz(xb,ya)
1750 z01=FNz(xa,yb):z11=FNz(xb,yb)
1760 q0=FNbelow(FNmin(z00,z01,z10,z11),
zinc5)
1770 q1=FNabove(FNmax(z00,z01,z10,z11),
zinc5)
1780 nz=INT((q1-q0)/zinc+.5)
1790 zcont=q1+zinc:GCOL0,1
1800 FOR z%=nz TO 0 STEP -1
1810 zcont=zcont-zinc
1820 PROCcontour(zcont,x,y,s)
1830 NEXT z%
1840 nz=INT((q1-q0)/zinc5+.5)
1850 zcont=q1+zinc5
1860 FOR z%=nz TO 0 STEP -1
1870 zcont=zcont-zinc5
1880 IF zcont>zmid THEN GCOL0,3 ELSE GC
OL0,2
1890 PROCcontour(zcont,x,y,s)
1900 NEXT z%
1910 ENDPROC
1920 :
1930 DEF PROCcontour(z,x,y,s)
1940 p1=FNin(z00,z,z10)
1950 p2=FNin(z00,z,z01)
1960 p3=FNin(z01,z,z11)
1970 p4=FNin(z10,z,z11)
1980 IF p1>=0 AND p2>0 AND p3>=0 AND p4
>=0 THEN N060
1990 IF p1>=0 AND p3>=0 THEN PROCjoin(x
```

*Continued on page 53*

# Program Development Made Easy

*Andrew Rowland describes some utilities which make editing Basic programs much more effective, using Edit on a Master, or View on a model B.*

For me, one of the joys of owning a Master is being able to edit programs in a text editor instantly, without all that tedious SPOOLing and EXECing. However, even the Master's Editor has some disadvantages: there is a limit on the length of program that can be transferred to the editor, and once line numbers are removed, you cannot return to Basic without EXECing.

This article presents four utilities: a pair so that BBC owners can edit programs in View with the same convenience as on a Master, and a pair to better facilitate using the Master's Edit. Both pairs may be used to edit without line numbers.

## EDITING WITHOUT LINE NUMBERS

The first thing I usually do on entering EDIT is to remove line numbers: it makes things a lot easier when copying or moving blocks around, you don't have to worry about squeezing enough new line numbers between two others, and I have a library of sub-routines saved without line numbers in ASCII format. These can then be inserted anywhere in a program while in the editor. It also encourages good discipline about not using GOTOs! You do have to avoid using RESTORE with line numbers, but with care, it is not necessary.

## ENTERING THE PROGRAMS

If you have a Master and use Edit, you will need listings 1 and 2, which produce the utilities XED and XBAS. Model B owners wanting the convenience of the

Master (or Master owners wedded to View) will find listings 3 and 4 fit the bill. This time the utilities are called VED and VBAS, and load your program into View for editing.

If you are entering the listings from the magazine, listing 4 is not reproduced as it is similar to listing 2, but with the following modifications:

```
10 REM Program .>VBASbas (4)
380 INY:INX
940 a$="SAVE VBAS "+STR$~install+" "
    +STR$~P%
```

## EDITING WITH LINE NUMBERS

The utilities can be modified as shown below to retain line numbers. For example, you may only want XED in order to avoid the 'No room' error that occurs with lengthy programs (maybe you should consider reducing the length of the program by CHAINing shorter ones or using overlays)! In this case, don't use XBAS - return to Basic in the usual way.

Listings 1 and 3: remove *JMP out* from line 610. Listings 2 and 4: delete lines 640 - 700. Master users will find listing 5 does the same job as listing 4 for less typing!

I do not use XED and XBAS exclusively, but use the normal route when I want line numbers present to find my place. View users may want the choice too. Create VED and VBAS normally, rename them to something appropriate, then perform the above modifications.

## Program Development Made Easy

---

### USING THE COMMANDS

Both sets of utilities work in the same way: type \*XED or \*VED in Basic's command mode and they will load the current program into Edit or View in the same way as the Basic command EDIT, but without line numbers. You should always save your work first, as the program will be corrupted if anything goes wrong. And if you are using View, it cannot cope with lines longer than 132 characters, so keep your program lines short - it improves readability anyway!

Then you may edit your program using all the word processor's facilities. Of course, you must ensure that no lines exceed 251 characters and that the last line is terminated by a carriage return. You cannot include Ctrl-@ (which appears black-on-white), as it is used as an end of file marker.

When your editing is over, you need to return to Basic. From Edit, press Shift/f4 (return to language) and enter 'XBAS' where you would usually type 'BASIC'. When the word *Escape* appears, your program is ready to list. It will have line numbers starting at 10 in increments of 10 - if you had any special numbering system, it will have been lost.

From View, you return to Basic by typing \*VBAS in View's command page.

### HOW THE UTILITIES WORK

Basic programs cannot be edited in a text editor: as you no doubt know, Basic has its own special format for programs, where Basic keywords like PRINT and ENVELOPE are replaced by a one byte code number called a *token*. There are also line numbers and line length bytes. Text editors need files containing only recognisable characters, so to edit an

existing program, it must first be converted into plain text (sometimes called ASCII), and afterwards converted back into Basic's internal format - this is called detokenising and tokenising.

Listing a program effectively detokenises it: the program is displayed in readable characters, just as you typed it in. XED and VED make Basic list the program too (by inserting LIST in the keyboard buffer), but instead of sending the output to the screen, the characters produced are stored in memory. A flag is set when a new line starts and five characters are missed, so that line numbers are not stored along with the rest (unless you disabled it). In addition, line feeds (ASCII code 10) which are sent at the end of lines are also skipped.

To do this, the OS routine used to write characters to the screen is diverted to a replacement routine via its vector at &20E, which is responsible for storing the characters in memory. It detects when the listing has ended by the appearance of the '>' prompt, and restores the vector.

Before this happens though, mode 7 is selected and the Basic program moved up in memory as far as it will go. The ASCII text is stored below the program starting at PAGE. This is to permit longer programs to be edited than the EDIT command can cope with (EDIT stores the ASCII text *above* the program so that if all the program cannot be detokenised, it is not corrupted and lost). Detokenised programs take up more room than Basic's internal format, but some overlap will not matter. Nevertheless, save your program first, especially if it *very* long.

To prevent problems, LIST and WIDTH are forced to zero. After this, the editor is



invoked and - if you are using Edit - you will be informed that there is text already in memory.

The way this is done may interest some readers. \*EDIT or \*EDIT <filename> also has an extended form: \*EDIT nn,nn where nn are two hexadecimal digits. These are not the start and end addresses of the text in memory but vectors to them. Thus if you invoke the editor with \*EDIT 00,02 then locations &00/&01 and &02/&03 should contain the start and end addresses of the text respectively.

On the other hand, View has to be tricked into thinking it has loaded a file. The command L ASC (load a file called ASC - a dummy name I use as a default) is inserted into the keyboard buffer as if it had been typed. The vector used by OSFILE to load files is diverted to a replacement routine. Then a \*WORD is issued, and the L ASC executed. However, the filing system never gets a chance to see the command: it is dealt with by the replacement routine which tells View how long the 'file' is (which is already in memory), resets the vector and exits.

### RETURNING TO BASIC

The utilities which return the ASCII program to Basic also employ a diverted vector, this time of OSWORD 0, the routine Basic uses to obtain a line of input. Whenever you enter a new line of a program at the '>' prompt, this routine is called by Basic. By diverting it, we can copy a line of our ASCII text into Basic's input buffer in page 7, and so fool it into thinking the line has been typed at the keyboard in the usual manner. Moreover, by ensuring the first line so copied is "AUTO", Basic will automatically add the line numbers as it goes. When the last

line has been 'entered' like this, the vector is restored to its former contents.

The text can be found easily by XBAS and VBAS because Edit and View always arrange the text in memory to start one page above OSHWM when 'Return to language' is used and place a zero byte at the end. It too, is moved as far up in memory as it will go, because although in general Basic programs are shorter than their ASCII equivalents, the fact that the line numbers are not present means that the tokenised program *can* get longer than the text as they are added, especially with long assembler listings which have little opportunity to benefit from tokenisation.

Finally, a note on listing 5, which is only for Master users who want to use View to create programs *with* line numbers. It uses the same mechanism as Edit does when returning to Basic - it places the address of the start of text in locations &00/01 and calls \*BASIC @.

Finally, a reminder that a version of XED that works with Bas128, called *BEDIT*, was published in BEEBUG Vol.9 No.4 on page 40.

### Listing 1

```
10 REM Program .>XEDbas (1)
20 REM Version B1.00
30 REM Machine Master/Compact
40 REM Author Andrew Rowland
50 REM BEEBUG July 1991
60 REM Program subject to copyright
70 :
100 wrchv=&20E:oswrch=&FFEE
110 osbyte=&FFF4:oscli=&FFF7
120 top=&12:page=&18:hmem=&6
130 listo=&1F:width=&23
140 pointer=&70:start=&72
150 :
```

```
160 FOR pass=0 TO 3 STEP 3
170 P%=&900:[OPT pass
180 .install LDA #22:JSR oswrch
190 LDA #135:JSR oswrch
200 \ alter WRCHV
210 LDA wrchv :STA oldv
220 LDA wrchv+1:STA oldv+1
230 SEI:LDA #entry MOD &100:STA wrchv
240 LDA #entry DIV &100:STA wrchv+1
250 CLI
260 \ move program up
270 LDA #132:JSR osbyte \ read HIMEM
280 STY himem+1:STX himem
290 DEY:LDX #0
300 .over STY pointer+1:STX pointer
310 LDY top:STX top:BEQ zero
320 .loop LDA (top),Y:STA (pointer),Y
330 DEY:BNE loop
340 .zero LDA (top),Y:STA (pointer),Y
350 DEC top+1:DEC pointer+1
360 LDX top+1:CPX page:BCS loop
370 LDX pointer+1:INX:STX page
380 LDA #2:STA pointer:STA start
390 LDA #131:JSR osbyte \ read OSHWM
400 STY pointer+1:STY start+1
410 \ WHEN0 0, WIDTH 0
420 LDX #0:STX listo:DEX:STX width
430 \ put "L." in k/b buffer
440 LDA #21:LDX #0:JSR osbyte:LDY #0
450 .loop LDA auto,Y:JSR poke
460 INY:CPY #3:BNE loop
470 RTS
480 .auto EQU$ "L."+CHR$13
490 \ perform *FX138,0,n
500 .poke TAX:TYA:PHA:TXA:TAY
510 LDA #138:LDX #0:JSR osbyte
520 PLA:TAY:RTS
530 \ *****
540 .entry STY ytemp
550 LDY flag:BNE secondtime
560 CMP #13:BNE eout:INC flag
570 .eout LDY ytemp:RTS
580 .secondtime CMP #10:BEQ out
590 LDY count:BEQ ok
600 CMP #ASC">":BEQ finished
610 DEC count:JMP out
620 .ok CMP #13:BNE notcr
```

```
630 LDY #5:STY count
640 .notcr LDY #0:STA (pointer),Y
650 INC pointer:BNE out:INC pointer+1
660 LDY pointer+1:CPY himem+1:BNE out
670 BRK:BRK:EQU$ "No room. Press BREAK
":BRK
680 .out LDY ytemp:RTS
690 :
700 .finished SEI
710 LDA oldv :STA wrchv
720 LDA oldv+1:STA wrchv+1:CLI
730 LDX #string MOD &100
740 LDY #string DIV &100
750 JMP oscli
760 .string EQU$ "EDIT "+STR$~start+",
"+STR$~pointer:EQU$ 13
770 :
780 .oldv EQU$ 0
790 .flag EQU$ 0
800 .ytemp EQU$ 0
810 .count EQU$ 5
820 JNEXT
830 a$="SAVE XED "+STR$~install+" "+STR$~P%
840 PRINT a$:OSCLI a$
```

## Listing 2

```
10 REM Program .>XBASbas (2)
20 REM Version B.00
30 REM Machine Master/Compact
40 REM Author Andrew Rowland
50 REM BEEBUG July 1991
60 REM Program subject to copyright
70 :
100 wordv=&20C:wrchv=&20E
110 osbyte=&FFF4:oscli=&FFF7
120 oswrch=&FFEE
130 zp=&70:block=&72:buffer=&74
140 FOR pass=0 TO 3 STEP 3
150 P%=&900:[OPT pass
160 .install \ disable ESCAPE
170 LDA #200:LDX #1:JSR osbyte
180 LDA #&7C:JSR osbyte
190 JSR setvecs:JSR findend:JSR movup
200 LDA #0:STA flag
210 LDX #string MOD &100
```

*Continued on page 54*

# Games Review

## Play It Again Sam 15

*Peter Rochford finds great entertainment in Superior Software's latest games compilation.*

<b>Product</b>	<b>Play it Again Sam 15</b>
<b>Supplier</b>	<b>Superior Software</b> P.O.Box 6, Brigg, S.Humberside DN20 9NH. Tel. (0652) 58585
<b>Price</b>	<b>£12.95 (tape)</b> <b>£14.95 (5.25" disc)</b> <b>£19.95 (3.5" disc)</b>

Things may come and things may go, but good old Sam goes on forever. Here we are at number fifteen in this continuing series of re-releases of past 'hits' from Superior Software. Actually, if it wasn't for Superior's support for the Beeb, there would have been little games software available for the Beeb over the last few years, so we shouldn't knock it.

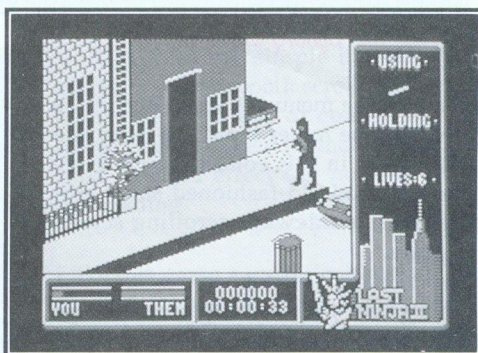
This compilation comprises four games and comes on three discs in both the forty and eighty track DFS versions. All games are compatible with the model B, B+ and Master 128. Detailed and well-written instructions are provided in the form of a fold-out leaflet.

First on offer, we have *Last Ninja II*, a martial arts game set in the heart of downtown New York. Your task is to find and defeat the evil Shogun, who for some unknown reason has left the shores of Japan and is hiding out in the Big Apple.

To find your man, you have to work your way through the streets and alleys

of the city, avoiding or destroying a vast array of assailants. At your disposal are a variety of weapons including shuriken stars, swords and clubs, and if all else fails, your fists. Apart from the combat, there are a number of puzzles to solve, if you are to progress through the game.

So much for the plot. The game itself has a rather bewildering number of keys and key combinations to contend with. I wasn't too impressed with this at first, and at times my Ninja warrior was performing some strange movements, definitely not what I always intended!

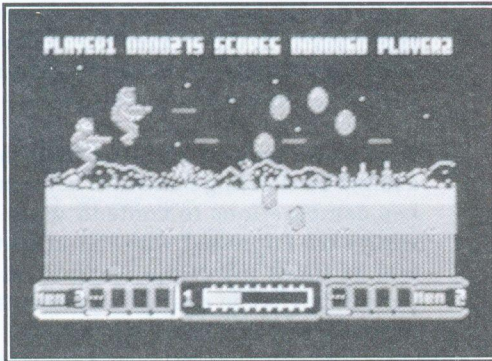


*Last Ninja II*

Graphics are quite good, with some nice animation touches, which shows that the good old eight bit Beeb is still a capable machine. Sadly, the four colour screen mode does tend to produce some confusion at times when two characters are close or plotted on top of one another. A similar loss of detail occurs when characters tend to blur or merge with a background.

## Games Review

Sound is OK but not outstanding, being confined to various squawks and shrieks. *Last Ninja II* is certainly an action packed game, and there is no time to hang around in it if you are going to be anywhere near successful. As I said earlier, the keys can be difficult to remember and master, and makes this game unsuitable for the young.



### *Cyborg Warriors*

Next on the menu is *Cyborg Warriors*, a new release rather than a re-issue like the others in this collection. Here we have a good old-fashioned shoot-em-up, with a sideways scrolling screen display.

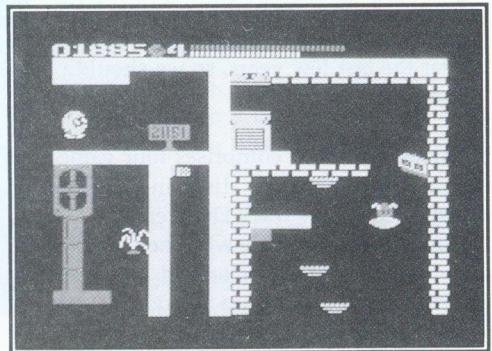
You guide your Cyborg robot across the surface of the alien planet and basically just shoot everything that is thrown at you. You can collect the tokens that drift past you, and this enables you to obtain more powerful weapons to destroy your many adversaries.

If you have seen R-type in the arcades, or played Nevryon from the 4th Dimension, you will have a good idea what this game is all about. It's very similar in

many respects, except that you are guiding a robot instead of a space craft.

There's lots of action in this game and it's lots of fun. Graphics are detailed and colourful with a pretty good sideways scrolling routine with very little flicker. Sound is good too, and when the action gets frenetic, there is a quite a wild racket going on: not the sort of game for late night playing! Key arrangements are simple, and it does not take long to get the hang of things.

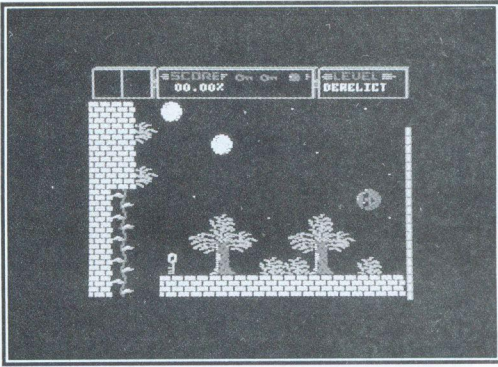
I am actually rather surprised that Superior has never released this game on its own. I would have thought as a single release, it would have done very well. As a part of this compilation, it is excellent value for money and a very worthy inclusion.



### *Network*

Moving on to the third game, we have *Network*. This is written by Peter Scott who has done some excellent work for Superior in the past. In this cross between an arcade adventure and a ladders and levels type game, you must collect the twenty parts of a machine called the Flynce.

The parts of the machine are hidden on various levels and you use lifts, extending platforms, springs and teleport machines to get you around. As you try to progress, naturally there are the usual bunch of aliens to thwart your efforts, and puzzles to solve too. There are over 100 screens to the game.



**Ricochet**

I was somewhat disappointed with this game from the word go. The graphics are not what I would call outstanding, and the game is a devil to get to grips with right from the first screen, never mind the other ninety-nine! I spent some not unreasonable time getting off the first screen, but was sure that I was not fully understanding what I was supposed to be doing. Hmmm. Anyhow, I think this game may be an acquired taste, or perhaps just appeal to those who have a sharper brain and are more dexterous than I. Certainly not my favourite game in this compilation.

Lastly, we come to *Ricochet*. This is an arcade adventure featuring SPRAT (Small Partially Robotic Alien). Different! The idea of the game is to guide our round, bouncing friend through each

level to find the hidden hourglass and teleport out to the next level. There are five levels in total, but each level consists of many screens. After each level you are given the password to the next, so that you do not have to go through those levels already completed.

SPRAT is a rather unusual character to guide around. He doesn't run and he doesn't jump as such. Rather, he rolls and bounces. To get him to move you can roll him along platforms, but to get through doors or up ladders, you hold down a key until he 'squishes' to half his normal size, then letting go of the key releases the energy and then he bounces, off walls, ceilings and anything else. Hence the name of the game, Ricochet.

Graphics are simple but effective, and with some smooth scrolling. There is plenty of good use of colour and the whole thing is nicely animated. Sound is used to good effect but nothing really outstanding.

I like this game a great deal. Just the fact that SPRAT has an unusual way of being moved around makes this all the more interesting than many other games of this genre. Added to that, there is a good challenge to the game.

Well there you have it. Fifteen down and I wonder how many more to go! This is one of the better Sam compilations as far as I am concerned. Apart from my own personal dislike of Network, the other three games I found excellent. Definitely recommended. **B**

# Recreational Mathematics

## The Mathematics of Encryption

by Michael Taylor

This is the last of four articles related to number theory. The accompanying program allows a message to be enciphered using the RSA (Rivest, Shamir and Adleman) method of encryption. It is intended to provide both entertainment and an explanation of the mathematics behind the new ciphers. The program may appear a little disappointing in that a restricted range of integers is used. However, it is intended to work quickly and easily, and the reader is encouraged to use procedures from last month to increase the range of integers and - when he buys an Archimedes - the speed!

In any case, readers would not want the pages of BEEBUG filled with procedures re-copied from last month's program. Rather, the program uses FNIndex, the function for finding powers used last time, but only with low moduli, below 46340, so that PROCProduct is not needed. If so desired readers can easily replace the new version of FNIndex, FNIndex2, with FNIndex and PROCProduct from last month's program and experiment with larger moduli, allowing more characters to be enciphered and in larger blocks. Also, they can accept the challenge of Bernard Hill (BEEBUG Vol.9 No.8) to adapt the precision arithmetic programs of BEEBUG Vol.3 No.1 and Vol.6 Nos. 5 & 6.

The whole advantage of the RSA method is that the enciphering algorithm can be published (hence references to *public* key encryption which appear in articles on this subject). This means that, say, an

agent in one country can encipher a message for transmission back to 'Control' and even if he and his equipment are captured, no one, not even he himself, can find out how to decipher any messages that he has sent. No courier need visit him with a key. It could be sent to him, by radio, for all to hear.

```
Simulation of the method of encryption of Rivest, Shamir and Adleman
Encipher or Decipher or Quit, E or D or Q? E
Two prime numbers are to be entered.
Their product must be less than INT(SQR(2^31-1)) = 46340,
and greater than 30^3 = 27000.
What is the value of the first prime factor of the modulus (e.g. 149)?
149
What is the value of the second prime factor of the modulus (e.g.233)?
233
The modulus is 149 x 233 = 34717 and its Euler Number is 148 x 232 = 34336
The exponent must be relatively prime to the Euler number of the modulus, and
so it must not share any factors with 148 or 232.
What is the exponent (e.g., a prime, 26987)?
26987
Write a message in CAPITAL LETTERS (space, comma and full stop also allowed)
with not more than 135 characters. e.g. Use copy key with:-
NOW IS THE TIME FOR ALL GOOD MEN, AND WOMEN, TO COME TO THE AID OF THE PARTY.
NOW IS THE TIME
1
0360989937103291053617522
```

### Encoding a message

The program is self-contained with instructions and can be typed in and run. For convenience, there is a message written in the program which can be easily input for enciphering using the Copy key. On deciphering, the Copy key can again be used to input the enciphered code so that the original message reappears. Of course, there is no need to type out the REM statements, but they comment on the program - about which little more need be said.

What is more interesting is the mathematical structure of RSA encryption, and this can be explained using the example written into the INPUT instructions in the program.

We will follow the path of one number, 21164 representing the word 'NOW' in the example message (the REM statements are sufficient comment on how the number was generated). This number 21164 is enciphered as the number 03609 and then deciphered back to 21164.

```

Simulation of the method of encryption of Rivest, Shamir and Adleman
Encipher or Decipher or Quit, E or D or Q? D
What modulus was used for encoding (the example given was 34717)?
34717
What is the exponent, which must be the inverse of the enciphering exponent,
modulo the Euler number of the modulus (for the example it must be 17731)?
17731
What is the code to be deciphered? If code is on the screen use Copy key.)
0360909937103291053617522
NOW IS THE TIME
```

### Decoding a message

In a real system the key that is sent consists of a large composite number, about 200 digits long, which has been made up as the product of two primes - which are *not* made public. In our little program, there is an example with each of the INPUT statements. As the program suggests we could use 149 and 233 whose product is 34717. It is this 34717 that is made public to anyone who wants to send a message. The other number that is made public is any number that is relatively prime to the Euler number of 34717: it is easiest just to select a prime number, in our case 26987. To encipher 'NOW', the number 21164 (which came from the word NOW) is raised to the power of 26987 (mod 34717). The resulting number to be transmitted is 03609.

The receiver must decode 03609. He does this by raising it to the power of a certain number, again modulo 34717. This number is 17731 and is only known by the potential receiver who originated the numbers in the first place. It is obtained by finding the inverse of 26987 modulo the Euler number of the modulus. For us

it is easy to find the Euler number of the modulus (using last month's program), and then to use it as modulus to find the inverse of 26987 which is 17731. In a real system, the number 34717 would be replaced by one about 200 digits long, and it would take far too long to factorise it and find its Euler Number.

How do we find the Euler Number of 34717? Since the modulus for transmission was made up from the product of two prime numbers, say  $pq$ , then its Euler number is  $(p-1)(q-1)$  (see *Recreational Mathematics in BEEBUG Vol.10 No.1*). In this case  $p=149$  and  $q=233$  and so the Euler number of the modulus

is  $148 \times 232$  which is 34336. We can only calculate this Euler number  $(p-1)(q-1)$  if we can factorise the modulus back to  $p$  and  $q$ . Our program 'Euler' would have no trouble with this - even if we had not already contrived 34717 as  $149 \times 233$  - but, as we have already said, if  $p$  and  $q$  were each 100 digits long, there is no computer in 1991 that could factorise  $pq$  to  $p$  and  $q$  and hence find the Euler number.

So, to decipher the transmitted code, 03609, we just raise it to the power of 17731 but to the same modulus 34717 as before, and recover the original number 21164.

We still have to explain why 17731, which is the inverse of 26987 modulo 34336 is indeed the exponent needed for deciphering the transmitted code. Here is an informal explanation.

Suppose we encipher  $A$  by raising it to the power  $E$  (mod  $m$ ). We then decipher it by further raising it to the power  $D$

## Recreational Mathematics

---

(mod  $m$ ). The result is  $(A^E)^D$  or  $A^{(E \cdot D)}$  (mod  $m$ ). We can argue that if  $E$  and  $D$  were inverses (mod  $\Phi(m)$ ) so that  $E \cdot D$  would be 1 modulo  $\Phi(m)$ , then  $A^{(E \cdot D)}$  modulo  $m$  would just be  $A$ .

Why must  $E$  and  $D$  be inverses modulo  $\Phi(m)$ ? Euler's theorem says that any number,  $A$ , raised to the power (mod  $m$ ) of the Euler number of  $m$ , equals 1 so long as  $A$  and  $m$  are relatively prime. Let the Euler number of  $m$  be  $\Phi(m)$ . Then,

$$\Phi(m)^A = 1 \pmod{m}$$

Dividing a power, such as  $A^B$ , by 1 has no effect on it (of course). From Euler's theorem  $A^{\Phi(m)}=1$  and so we can divide  $A^B$  by  $A^{\Phi(m)}$  (mod  $m$ ) and leave it unchanged. This is equivalent to subtracting  $\Phi(m)$  from  $B$ . We could repeat this process of subtracting  $\Phi(m)$  from the exponent indefinitely. This means that we can reduce  $A^B$  (mod  $m$ ) by reducing  $B$  (mod  $\Phi(m)$ ). In particular, if  $B$  is  $D \cdot E$  where  $E$  and  $D$  are inverses (mod  $\Phi(m)$ ), then  $A^{(E \cdot D)} = A^1 = A$  (mod  $m$ ). And we see why  $E$  must be relatively prime to the Euler number of the modulus: it must have an inverse (mod  $\Phi(m)$ ).

Thus if  $A$  is enciphered by raising it to the power  $E$ , it can be deciphered by further raising it to the power  $D$ , provided that  $D$  is the inverse of  $E$  modulo  $\Phi(m)$ .

In the example suggested in the program, if  $A$  is 21164,  $A^E$  is  $21164^{26987} \pmod{34717}$  which is 03609, and  $(A^E)^D = A^{(E \cdot D)}$  is  $03609^{17731} \pmod{34717}$  which is just 21164 again, since  $26987 \cdot 17731 = 1 \pmod{34336}$ .

By now the reader should be able to explain the example, given with bigger numbers at the end of last month's article. The reader was invited to

factorise the modulus 796652189. It can be tackled with the program Euler of three months ago which finds the factors 30829 and 25841. This means that its Euler Number is  $30828 \cdot 25840 = 796595520$ . We choose another prime number 254665717 whose inverse (mod 796595520) can now be found as 161722333. Thus 666666666 is encoded by raising it to the power of 254665717 (mod 796652189) to give 304117466. To decode we raise 304117466 to the power of 161722333 (mod 796652189) to recover 666666666. Do check this if you have last month's program Fermat (BEEBUG Vol.10 No.2) at hand.

A final comment: why is number theory not to be found in school mathematics? It links mathematics with its past, right back before 1600 BC, before the Greeks to the study of 'Pythagorean' triples on a Babylonian tablet; it is 'core maths' for computing; it delights 'pure' mathematicians - and it can be tackled by students almost independently of their skills - or lack of them - in other branches of mathematics.

### PROGRAM NOTES

The program is only a vehicle for the explanation of the mathematics of the RSA encryption method. The comments after the REM's should be sufficient explanation. The choice for  $p$  and  $q$  is restricted. In order to encipher just 26 capital letters, comma, full stop and space, 29 numbers are needed - and 30 are used for convenience.

To encipher them in blocks of 3,  $30^3$  or 27000 possible numbers are needed which means that the modulus,  $m = pq$  must be greater than this. However, since we have not bothered with the PROCProduct from last month's program, we must ensure that the product of any integers is no more than



$2^{31}-1$  which means that we must keep the modulus below  $\text{INT}(\text{SQR}(2^{31}-1))$  or 46340. Readers with either patience or an Archimedes, may like to change FNIndex2 to FNIndex from last month's program and add the procedure, PROCProduct, which it calls.

Some References on number theory and the RSA cipher are:

1. *Think of a Number* (1990), by Malcolm E. Lines, Adam Hilger (discusses modular arithmetic, the RSA system and much else of entertainment).
2. *Numbers, Groups and Codes* (1989), by J.F. Humphreys and M.Y. Prest, Cambridge University Press (this includes an account of the RSA cipher system. It would also make excellent reading for a 6th former thinking of mathematics or computing at university).
3. There is an article on two trap-door ciphers, one of them the RSA one: "*The Mathematics of Public-Key Cryptography*", by Martin E. Hellman, in *Scientific American*, Vol.241, No.2, August 1979, pp.130 to 139.
4. "*Cryptology*", by Bernard Hill, in *BEEBUG* (January/February 1991) Vol.9 No.8 (introduces both traditional ciphers and the new RSA system).

```

10 REM Program RSA
20 REM Version B1.0
30 REM Author Michael Taylor
40 REM BEEBUG July 1991
50 REM Program subject to copyright
60 :
100 MODE0
110 VDU19,0,7,0;0;0;
120 VDU19,1,0,0;0;0;
130 DIM B%(20),F%(3),L%(3)
140 PRINT'SPC4;"Simulation of the meth

```

```

od of encryption of Rivest, Shamir and A
dleman"
150 REPEAT
160 PROCShove(5,5)
170 INPUT" Encipher or Decipher or Qui
t, E or D or Q? "I$
180 IFI$="E" PROCEncipher ELSE IF I$="
D" PROCDecipher ELSE IF I$="Q" CLS:END
190 VDU13:PROCClear
200 UNTILFALSE
210 END
220 :
1000 DEF PROCEncipher
1010 Message$=""
1020 PRINT"" Two prime numbers are to b
e entered."" Their product must be less
than INT(SQR(2^31-1)) = 46340,"" and g
reater than 30^3 = 27000."
1030 PROCShove(6,6)
1040 PRINT"" What is the value of the f
irst prime factor of the modulus (e.g. 1
49)?"
1050 INPUT" Prime1%
1060 PROCShove(6,6)
1070 PRINT"" What is the value of the s
econd prime factor of the modulus (e.g.2
33)?"
1080 INPUT" Prime2%
1090 Modulus%=Prime1%*Prime2%
1100 PROCShove(6,6)
1110 PRINT"" The modulus is ";Prime1%;
x ";Prime2%;" = ";Modulus%;" and its Eu
ler Number is ";Prime1%-1;" x ";Prime2%-
1;" = ";(Prime1%-1)*(Prime2%-1)
1120 PROCShove(8,8)
1130 PRINT"" The exponent must be relat
ively prime to the Euler number of the m
odulus, and so it must not share any f
actors with ";(Prime1%-1);" or ";(Prime2
%-1);"." What is the exponent (e.g., a
prime, 26987)?"
1140 INPUT" Exponent%
1150 PROCShove(6,6)
1160 PRINT"" Write a message in CAPITAL
LETTERS (space, comma and full stop als
o allowed) with not more than 135 cha
racters. e.g. Use copy key with:--"" NO

```

## Recreational Mathematics

```
W IS THE TIME FOR ALL GOOD MEN, AND WOM
EN, TO COME TO THE AID OF THE PARTY."
1170 PROCShove(6,6):U%=POS:V%=VPOS:PRIN
TSPC136;"|":VDU31,U%,V%
1180 INPUTLINE"Message$:REM (to accep
t commas too, INPUTLINE)
1190 P$=LEFT$(Message$,135):REM (restr
ict length of 135 characters)
1195 PROCShove(8,8)
1200 PRINT"" ";
1210 P$=P$+" ":REM (pad with three s
paces)
1220 L%=LENP$
1230 FOR Z%=1 TO L%-3 STEP3
1240 L$=MID$(P$,Z%,3):REM (characters
in groups of three)
1250 FOR Y%=0 TO 2:
1260 D%=ASC MID$(L$,Y%+1,1)
1270 IF D%=32 F%=27 ELSE IF D%=44 F%=28
ELSE IF D%=46 F%=29 ELSE F%=D%-64
1280 F%(Y%)=F%:REM (three characters d
enoted by F%(1), F%(2) and F%(3))
1290 NEXT
1300 G%=F%(0)+F%(1)*30+F%(2)*30*30:REM
(from F%(Y%)'s, G% formed to base 30)
1310 H%=FNIndex2(G%,Exponent%):REM (enc
ipher G% as H%)
1320 IF LENSTR$H%<5 S$=STRING$(5-LEN(S
TR$(H%)), "0")+STR$(H%) ELSE S$=STR$H%
1330 PRINTS$;:REM (print string of dig
its for transmission)
1340 NEXT:PRINT""
1350 ENDPROC
1360 :
1370 DEF PROCDecipher
1380 PROCShove(6,6)
1390 PRINT"" What modulus was used for
encoding (the example given was 34717)?"
1400 INPUT"Modulus"
1410 PROCShove(7,7)
1420 PRINT"" What is the exponent, whic
h must be the inverse of the enciphering
exponent, modulo the Euler number of
the modulus (for the example it must be
17731)?"
1430 INPUT"Inverse%"
1440 PROCShove(6,6)
1450 PRINT"" What is the code to be dec
```

```
iphered? If code is on the screen use C
opy key.)"
1460 INPUT"Code$:REM (the received d
igits)
1470 PROCShove(6,6)
1480 PRINT"" ";
1490 Length%=LENCODE$
1500 FOR Z%=1 TO Length% STEP5
1510 Number$=MID$(CODE$,Z%,5):REM (cho
p into five digit groups)
1520 Number%=VAL(Number$)
1530 K%=FNIndex2(Number%,Inverse%):REM
(decipher each group of five digits)
1540 REPEAT
1550 J%=K%MOD30:REM (re-interpret in a
rithmetic to base 30)
1560 K%=K%DIV30
1570 IF J%=27 T%=32 ELSE IF J%=28 T%=44
ELSE IF J%=29 T%=46 ELSE T%=J%+64:REM
(re-generate ASCII numbers)
1580 PRINTCHR$(T%);:REM (print original
message)
1590 UNTILK%=0
1600 NEXT:PRINT""
1610 ENDPROC
1620 :
1630 DEF FNIndex2(U%,V%):LOCAL J%,K%,Y%
,Z%:REM (finds U%^V% mod M%)
1640 M%=Modulus%
1650 J%=U%MODM%:K%=V%
1660 S%=1:Z%=-1
1670 REPEAT
1680 Z%=Z%+1
1690 B%(Z%)=K%MOD2
1700 K%=K%DIV2
1710 IF Z%=0 Y%=J% ELSE Y%=(Y%*Y%)MODM%
1720 IF B%(Z%)=1 S%=(S%*Y%)MODM%
1730 UNTIL K%=0
1740 =S%
1750 :
1760 DEF PROCclear
1770 VDU11,13:PRINTSPC79:VDU11,13
1780 ENDPROC
1790 :
1800 DEF PROCShove(U%,V%)
1810 FOR Z%=1 TO U%:VDU10:NEXT
1820 FOR Z%=1 TO V%:VDU11:NEXT
1830 ENDPROC
```

B

# Passing Information via Function Keys

by Ashley Allerton

When the BBC Micro moves from Basic to another 'language', such as View or Wordwise, the program in memory is lost, and some way sometimes has to be found of passing necessary information across this boundary, e.g. to load a particular named file.

One way of accomplishing this is via one of the function keys, but this prevents the programming of this key for use in (say) the word processor - without undue complications, that is. If like me you want to make use of all 10 function keys unhindered, the following method can be used (using key 15 as an example):

```
100 OSCLI("FX4,2"):OSCLI("KEY15 *WORD
|MLOAD "+filename$+"|MMODE128|M*FX4,0|M*
KEY15|M")
110 *FX138,0,143
120 END
```

Line 100 sets the 'soft' keys to accept strings, and then feeds the necessary

returns to keyboard control. This 'software' pressing of the defined key causes the string assigned to it to be entered into the keyboard input buffer, where it is executed as though it had been typed in by the user. In the example, the equivalent keyboard commands would have been:

```
*WORD          (invoke View)
LOAD <filename> (load a specified file
where filename is the name assigned to
the variable filename%)
MODE128        (select mode 128)
```

By using the method described above, a Basic program could ensure that View is called with this information passed to it, and the same technique could be used whenever a Basic program transfers control to another 'language'.

In the command \*FX138,0,n 'n' corresponds to the soft keys as shown in Table 1.

F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	Copy	C/L	C/R	C/DN	C/UP
0	1	2	3	4	5	6	7	8	9	11	12	13	14	15
128	129	130	131	132	133	134	135	136	137	139	140	141	142	143

Table 1. Soft keys and related ASCII values

information into Key 15 (Cursor up): anything you like can be inserted here, such as the loading of a printer driver, changing screen colours or whatever the circumstances require. The soft keys are then returned to their usual state.

Line 110 enters a code into the keyboard buffer, which has the same effect as pressing 'Cursor up' on the keyboard, and this is picked up when the program listed above ends, and the computer

In the table the first row of figures shows the key number (as used in \*KEY) of each programmable key. The bottom row shows the corresponding values used in the \*FX138 command.

Key 10 (n=138) is the Break key, which can also be used, but as this first performs a Break operation, there are various side-effects, which you may or may not want. This key is therefore omitted from Table 1. B

# DFS Disc Recovery Utility

*This program by Trevor Pullen enables you to recover accidentally deleted, and possibly corrupted, disc files. It was first published in Vol.4 No.8 and is repeated for the benefit of newer readers, particularly as that issue is no longer available.*

It is all too easy to lose a valuable program or other file by deleting it. A moment of error can cost you hours of work. Discs can become corrupted too, with equally disastrous results. The program listed here, *Recover*, will provide a much-needed solution to this problem. It can be used to rescue programs or files of any length up to the maximum allowed by the disc or the DFS.

## UNDERLYING PRINCIPLES

A disc is divided into concentric tracks, each track being divided into sectors (10 for single density, 16 or 18 for double density) configured to hold 256 bytes of data. When a file is saved it goes onto the disc as a series of 256 byte blocks stored sequentially. For the computer to retrieve the file, it must know at which track and sector the file starts and how long the file is. This information is stored in the catalogue in the first sector along with other information such as the start and execution addresses for each program file.

When a \*DELETE command is given, the catalogue reference is removed but the file itself is left untouched. However, if the disc is subsequently compacted or another file is saved then the area where the 'lost' file is stored may well be overwritten and the file lost for ever. Thus, if a file is accidentally deleted, make no further saves to the disc and under no circumstances compact it until the required file has been rescued.

The computer will also lose track of the location and length of the files on the disc if the catalogue becomes corrupted. The catalogue is most prone to corruption because it is the most frequently accessed part of the disc - the rest of the disc may well still be OK. Once all the files have been rescued it is usually only necessary to re-format the disc to make it re-usable.

So, to recover a 'lost' file we must manually locate its start and end. This information can then be used to patch the catalogue, or to allow the file to be lifted from the disc and saved to a new one, automatically generating the required new catalogue entry. If rescue of a single deleted file is required then a fresh disc is not really necessary, as the original one can be used.

## USING THE PROGRAM

Once you have typed in and saved the program as *Recover*, run it and, as prompted, specify the drive holding the file to be rescued. This will cause *Recover* to display the contents of the first sector of that disc, along with the sector and track number and other information.

You should now scan through the disc, sector by sector, looking for your file. The '<' and '>' keys will display the previous and next sectors. Alternatively, you can go straight to a particular sector by using the up and down cursor keys to increase and decrease the track number

and the right and left cursor keys to increase and decrease the sector number, and then press the '<' or '>' key.

Only legal ASCII codes (32 to 126) are displayed, so identifying Basic programs can be quite difficult. It is always a good idea to include an identifying REM statement at the start of all your programs, as in our listings. *Recover* will insert a solid white block in the first character position of the sector displayed if it thinks you may be at the start of a Basic program. However, this is not an infallible guide. If you are not sure, you could recover the sectors anyway, and have a look at the file at a later stage.

Having located the start sector press 'S'. This will reset the displayed program length to &100 bytes (one block), and display the file start location on the disc. Now you need to find the end of the file. This is often the sector immediately before the beginning of the next file. Finally to rescue the file press 'R'. After a short pause, whilst the file is loaded into memory, a prompt for the new filename will be issued and you will be asked to insert the destination disc (of the same format as the source disc) into drive 0.

If the catalogue has been severely damaged *Recover* will not be able to read the format of the disc and will probably print the 'Disc Error 18' message. In such a case, the program can be fooled into reading the format, if a good disc of identical format is inserted into the drive and the program is restarted by pressing Escape. Once the first sector has been read, the corrupt disc can be reinserted and you can progress as before.

In the case of repeated disc error messages on all or most tracks, it is likely

that the whole disc is severely corrupted and recovery of files is probably impossible. You should be wary of continuing to use such a disc.

The moral here must be to avoid problems before they happen if at all possible. When developing programs use several discs and save updated copies frequently.

*Note:* the original program used OSWORD &7E to determine the size of the disc. However, there is a serious bug in the new Master MOS (3.50) which prevents this from working. We will cover this bug in the next issue of BEEBUG, since under some circumstances (luckily rare), it can have damaging results. The listing given here should work correctly.

```
10 REM          >Recover
20 REM Program Disc Recovery
30 REM Version B2.1
40 REM Author  Trevor Pullen
50 REM BEEBUG  July 1991
60 REM Program Subject to Copyright
70 :
80 ON ERROR GOTO 140
90 MODE 7:end=FALSE:PROCassemble
100 REPEAT:PROCinitialise:UNTIL format
110 REPEAT:PROCscan:UNTIL exit
120 PROCrecover
130 :
140 IF end GOTO 190
150 IF ERR=17 RUN
160 IF ERR=204 error=TRUE:PROCsavecat
170 IF ERR=190 OR ERR=197 OR ERR=198 O
R ERR=199 OR ERR=201 REPORT:PRINT;" Inse
rt a new disc.""Hit any key to continue
";:X=GET:PROCsavecat
180 REPORT:PRINT;" at line ";ERL
190 VDU 23;11,255;0;0;0;0:*FX4,0
200 END
210 :
220 DEFPROCassemble
230 DIMspace 30
```

## DFS Disc Recovery Utility

```
240 FOR pass=0 TO 2 STEP 2
250 P%=space
260 [
270 OPT pass
280 .print
290 LDY #0
300 .again
310 LDA &3000,Y:CMP #32:BMI change
320 CMP #126:BPL change:JMP aclr
330 .change
340 LDA #46
350 .aclr
360 JSR &FFE3:INY:ENE again:RTS
370 ]
380 NEXT
390 ENDPROC
400 :
410 DEFPROCinitialise
420 exit=FALSE:recov=FALSE
430 read=TRUE:error=TRUE
440 start=&3000:len=1:stkr=0:ssec=0
450 sector=0:track=0:CLS:*FX4,1
460 VDU 26;23;11,255;0;0;0
470 FOR Y=0 TO 1:PRINTTAB(13,Y);CHR$(1
41);"Disc Recovery":NEXT Y
480 VDU28,0,24,39,2,12
490 REPEAT INPUT'"Which drive (0,1,2,
3): "sdrv:UNTIL sdrv>=0 AND sdrv<=3
500 drv=sdrv:PROCgetformat
510 IF format=FALSE ENDPROC
520 CLS:VDU26;23;11,0;0;0;0
530 PROCaccessdisc
540 PRINTTAB(0,3)"'ESC' To Re-RUN";TAB
(0,4)"'E' To Exit";TAB(21,3)"'R' To Reco
ver file";TAB(21,4)"'S' To Set start"
550 PRINTTAB(22,9)"Track:0";TAB(22,10)
"Sector:0";TAB(22,11)"Length:&0";TAB(22,
12)"Abs. Sec:&0"
560 ENDPROC
570 :
580 DEFPROCscan
590 PRINTTAB(22,17)"Disc Format:"
600 IF trk=80 PRINTTAB(22,18)"80 Track"
ELSE PRINTTAB(22,18)"40 Track"
610 IF sec=17 PRINTTAB(22,19)"Double D
ensity" ELSE PRINTTAB(22,19)"Single Dens
ity"
620 PRINTTAB(18,14)"Start Trk,Sec: ";s
trk;",";ssec;" "
630 PRINTTAB(30,11);~len*256;" "
```

```
640 REPEAT:A=GET:UNTIL A=139 OR A=138
OR A=137 OR A=136 OR A=44 OR A=46 OR A=6
0 OR A=62 OR A=83 OR A=115 OR A=82 OR A=
114 OR A=69 OR A=101
650 IF A=69 OR A=101 end=TRUE:CLS:GOTO
140
660 IF A=82 OR A=114 exit=TRUE:ENDPROC
670 IF (A=83 OR A=115) stkr=track:ssec
=sector:len=1
680 *FX21,0
690 IF A=60 OR A=44 OR A=136 OR A=138
dir$="B" ELSE dir$="F"
700 IF A=83 OR A=115 ENDPROC
710 jump=TRUE:IF A>=136 jump=FALSE
720 IF dir$="F" PROCforward
730 IF dir$="B" PROCbackward
740 PRINTTAB(28,9);track;" ";TAB(29,10
);sector;" ";TAB(32,12);~(track*(sec+1)+
sector);" "
750 IF jump PROCaccessdisc
760 IF ?&7A=0 PRINTTAB(2,6);SPC(37) EL
SE PRINTTAB(2,6);"Disc Error : &";~?&7A;
" "
770 ENDPROC
780 :
790 DEFPROCrecover
800 CLS
810 PRINTTAB(2,4)"Please wait...LOADIN
G"
820 VDU28,0,24,39,2;23;11,255;0;0;0
830 recov=TRUE:track=stkr:sector=ssec
840 C%=-1:proglen=len:copypen=len
850 REPEAT:read=TRUE:C%=C%+1
860 IF C%>0 AND sdrv=0 PRINT'"Insert S
ource Disc. Hit any key";:X=GET
870 drv=sdrv:PROCsetdrive
880 IF copypen>73 loop=73 ELSE loop=co
pylen
890 IF C%>0 track=stkr:sector=ssec
900 FOR S%=0 TO loop
910 start=&3000+256*S%
920 PROCaccessdisc:PROCforward
930 NEXT S%
940 copypen=copypen-loop
950 stkr=track:ssec=sector
960 IF C%=0 PROCsavecat ELSE PROCsavem
em
970 UNTIL copypen=0
980 PRINT'"Do you wish to run again -
Y/N ";:X$=GET$
```

```

990 IF X$="N" end=TRUE ELSE RUN
1000 ENDPROC
1010 :
1020 DEFPROCsavecat
1030 drv=0:PROCsetdrive:CLS
1040 IF error PRINTTAB(2,4)"Bad Filenam
e - Press 'C' to continue":REPEAT X=GET:
UNTIL X=67
1050 error=FALSE:CLS
1060 INPUTTAB(2,4)"Please enter new Fil
ename: "fn$
1070 PRINT""Insert Destination Disc int
o Drive 0. Hit any key";:X=GET
1080 IF sdrv<>0 PRINT""Please wait...CO
PYING"
1090 $&A00=fn$:!&70=&A00:!&72=PAGE
1100 !&7A=&3000
1110 !&7E=(&3000+256*proglen)
1120 A%=0:X%=&70:Y%=0:CALL &FFDD
1130 ENDPROC
1140 :
1150 DEFPROCsavemem
1160 IF sdrv=0 PRINT""Insert Destinatio
n Disc. Hit any key";:X=GET
1170 track=0:sector=1
1180 drv=0:PROCsetdrive
1190 start=&7A00:read=TRUE
1200 PROCaccessdisc
1210 ft=256*(?&7A0E AND &3)+?&7A0F+C%*7
4
1220 track=ft DIV (sec+1):sector=ft MOD
(sec+1)
1230 read=FALSE:FOR S%=0 TO loop
1240 start=&3000+S%*256
1250 PROCaccessdisc:PROCforward
1260 NEXT S%
1270 ENDPROC
1280 :
1290 DEFPROCforward
1300 IF track=trk-1 AND sector=sec ENDP
ROC
1310 IF A=139 len=len+sec+1 ELSE len=le
n+1
1320 IF A=139 AND track<trk-1 track=tra
ck+1:ENDPROC
1330 flag=FALSE
1340 IF sector<sec sector=sector+1 ELSE
sector=0:flag=TRUE
1350 IF flag=TRUE AND track<trk-1 track
=track+1

```

```

1360 ENDPROC
1370 :
1380 DEFPROCbackward
1390 IF track=0 AND sector=0 ENDPROC
1400 IF A=138 AND len>sec+1 len=len-sec
-1 ELSE IF len>1 len=len-1
1410 IF A=138 AND track>0 track=track-1
:ENDPROC
1420 flag=FALSE
1430 IF sector>0 sector=sector-1 ELSE s
ector=sec:flag=TRUE
1440 IF flag=TRUE AND track>0 track=tr
ack-1
1450 ENDPROC
1460 :
1470 DEFPROCaccessdisc
1480 PROCosword(track,sector)
1490 IF recov ENDPROC
1500 VDU28,0,24,15,8;30
1510 CALLprint:VDU 26
1520 IF ?&3000=13 AND (?&3001=0 OR ?&30
01=&FF) PRINTTAB(0,8);CHR$(255)
1530 ENDPROC
1540 :
1550 DEFPROCosword(t,s)
1560 ?&70=drv:!&71=start:?&75=3
1570 IF read ?&76=&53 ELSE ?&76=&4B
1580 ?&77=t:?&78=s:?&79=&21
1590 X%=&70:Y%=0:A%=&7F:CALL &FFF1
1600 ENDPROC
1610 :
1620 DEFPROCgetformat
1630 drv=sdrv:PROCsetdrive
1640 PROCosword(0,1):trk=0
1650 A%=256*((256*(start?6 AND3))+start
?7)
1660 IF A%=&5A000 THEN trk=80:sec=17
1670 IF A%=&32000 THEN trk=80:sec=9
1680 IF A%=&30000 THEN trk=40:sec=17
1690 IF A%=&19000 THEN trk=40:sec=9
1700 format=TRUE
1710 IF trk=0 PRINT"Unrecognised disc f
ormat":format=FALSE
1720 IF trk=0 PRINT""Press any key to c
ontinue":X=GET
1730 ENDPROC
1740 :
1750 DEFPROCsetdrive
1760 OSCLI"*DRIVE "+STR$(drv)
1770 ENDPROC

```

# SPECIAL

## 40% Members Discount on



### Beebug C & Stand Alone Generator

Beebug C is the much acclaimed C programming language for the BBC Micro and Master 128. Although normally only available on more powerful computers, Beebug C is a full implementation of the Kernighan & Ritchie standard. Features include:

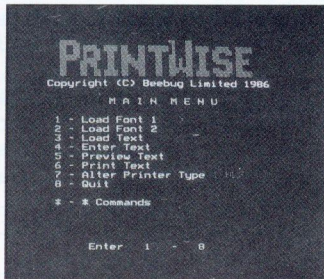
- ★ Runs on BBC B, B+ and M128.
- ★ Comprehensive set of ANSI functions.
- ★ OS functions vdu, osbyte, mode etc.
- ★ Command line interpreter.
- ★ Floating-point maths.
- ★ Linker for multi-source programs.
- ★ Maths functions.
- ★ Expandable run-time library.
- ★ Stand alone generator.
- ★ Full macro handling facilities.
- ★ Supplied on 2 ROMs & library disc

Normal price: **£79.69** inc VAT

Offer price: **£47.81** inc VAT

Stock code: 0074 40T, 0075 80T

Please add £3.50 carriage.



### Printwise

Printwise is a low-cost publishing aid allowing you to create professional looking magazines, leaflets, posters etc - the possibilities are endless. Simply take your text file, use embedded commands to specify the font styles you require, and let Printwise do the rest.

- ★ 9 authentic fonts from 4pt to 40pt.
- ★ Font designer for creating new fonts.
- ★ Fonts may be used on the same line.
- ★ Italics, bold, condensed, reversed etc.
- ★ Proportional spacing.
- ★ Subscript and superscript.
- ★ Left, centre, right & full justification.
- ★ Suitable for Epson compatible printers.

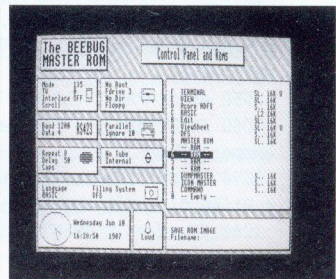
Printwise is easy to use and requires no programming skills. It may be used with text files created on Wordwise, View, InterWord, Mini Office and almost any text editor.

Normal price: **£30.66** inc VAT

Offer price: **£18.40** inc VAT

Stock code: 0085 40T, 0086 80T

Please add £2.50 carriage.



### Master ROM

The Master ROM is a powerful 32K ROM packed with features to enhance the facilities of the Master 128.

**Disc Menu** - A single command takes you to a full feature disc menu displaying all the items in the current directory. You can change directories or run, copy, delete, rename selected files.

**Control panel** - This displays all the computers status settings and the ROMs fitted. The cursor keys may be used to adjust any of the settings, which may be saved for future loading at any time.

**Disc commands** - A whole range of useful ADFS commands, including: \*FIND, \*FORMAT, \*VERIFY, \*BACKUP, \*MERGE, \*WIPE etc.

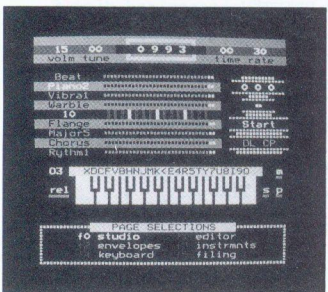
Plus: **16K-64K Printer buffer**  
**Simple 16K-64K RAM disc**  
**Diary and automatic alarm**

Normal price: **£39.84** inc VAT

Offer price: **£23.90** inc VAT

Stock code: 0087 ROM

Please add £2.50 carriage.



### Studio 8

Studio 8 is a real-time studio system with digital recorder, rhythm and drum machines which give hours of entertainment.

**Studio** - Allows playing and recording in real time with keyboard and sequencer.

**Editor** - A full-screen editor allowing the precise editing of notes.

**Envelope editor** - allows the definition of up to 16 amplitude and pitch envelopes.

**Instrument definer** - Create up to 32 instruments by combining pitch and amplitude envelopes, volume & sustain.

Plus many more features.

Normal price: **£22.48** inc VAT

Offer price: **£13.49** inc VAT

Stock code: 0009 80T

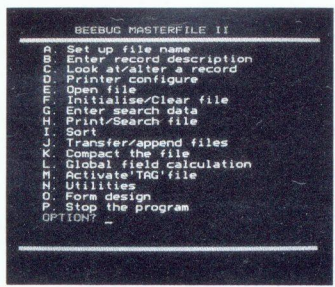
Please add £2.50 carriage.



Offer ends  
31st July 1991

# OFFER

## BBC & Master software



### Masterfile II

Masterfile II is the best selling general purpose database for the BBC Micro and Master 128. It has many powerful features, but is easy-to-use and ideally suited for home, business or school. It is supplied ready to run and no programming skills are required.

- ★ Menu driven for ease-of-use.
- ★ Fast 'tag' sorting.
- ★ Flexible print layouts.
- ★ Global calculations.
- ★ File size limited only by disc capacity.
- ★ File append options.
- ★ Search for a particular match.
- ★ Sorting on any field or fields.
- ★ Direct fast access to any record.
- ★ Form designer.
- ★ Label printing facility.
- ★ Record insertion.
- ★ Example database and tag files.

Normal price: **£22.48** inc VAT  
 Offer price: **£13.50** inc VAT  
 Stock code 0024 40T, 0025 80T,  
 0081 ADFS for Master 128 only  
 Please add £2.50 carriage.



### Command

The Command ROM is a powerful communications package that may be both menu or command driven. Command may be used with Hayes and other intelligent modems.

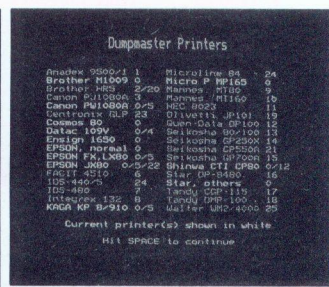
**Text terminal** - Use this to access scrolling text services such as Telecom Gold. XMODEM file transfer allows files to be sent around the world.

**Viewdata terminal** - A full feature terminal giving access to Prestel and other Viewdata services. Frames may be saved to disc, printed, tagged etc. and a full feature Viewdata editor is provided.

**Telephone directory** - Set up the name, number and log-on for services you wish to recall at a later date.

**Command driven** - A wide range of commands may be used in Basic to create applications for your own individual needs.

Normal price: **£39.84** inc VAT  
 Offer price: **£23.90** inc VAT  
 Stock code: 0073 ROM  
 Please add £2.50 carriage.



### Dumpmaster II ROM

Dumpmaster provides fast screen dumps using up to 8 shades from any screen mode. It includes a 'snapshot' facility to dump screens from programs while they are running. A wide range of printers are supported, including Epson compatibles and the Star LC10 colour.

Normal price: **£31.68** inc VAT  
 Offer price: **£19.00** inc VAT  
 Stock code 0053. ROM  
 Please add £2.50 carriage.

### Exmon II

This acclaimed ROM adds over 60 commands to manipulate and debug machine code programs. Features include:

- ★ set breakpoints ★ trace ★ set registers
- ★ full-screen RAM editor ★ single-step
- ★ disassembler ★ dual screens ★ search
- ★ line assembler ★ program relocater

Normal price: **£32.70** inc VAT  
 Offer price: **£19.60** inc VAT  
 Stock code: 0004 ROM  
 Please add £2.50 carriage.

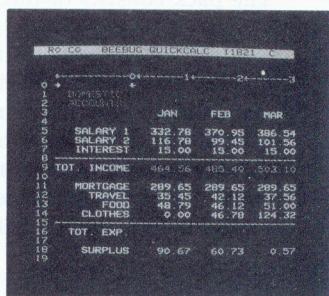
### Quickcalc

Quickcalc is an easy-to-use disc based spreadsheet enabling you to use the calculating power of your computer without any need to program. It is ideal for personal accounts, stock control, and general financial planning.

#### Features include:

- ★ 20 x 50 default spreadsheet.
- ★ Load, save and print spreadsheet.
- ★ Replicate into row, column or area.
- ★ Simple histogram capability.
- ★ Individual column widths may be altered at any time.
- ★ Min, max and sum functions.
- ★ Very easy to use.

Normal price: **£18.39** inc VAT  
 Offer price: **£11.00** inc VAT  
 Stock code: 0029 80T DFS  
 Please add £2.50 carriage.



# BEEBUG Education

by Mark Sealey

<b>Product</b>	<b>Time Traveller</b>
<b>Supplier</b>	<b>ESM</b> <b>Duke Street,</b> <b>Wisbech,</b> <b>Cambridge PE13 2AE.</b> <b>Tel. (0945) 63441</b>
<b>Price</b>	<b>£40.00 ex. VAT</b>

## INTRO

Helping younger children to understand the notion of how chronology and the passage of time can be represented in written or tabular form is often quite difficult. Familiarity with 'time-lines', timetables and other such layouts helps. An interactive computer program, which encourages experimentation and editing of data, can take pupils a long way in this direction.

ESM has collaborated with Homerton College, Cambridge and NCET to produce an interesting pack called *Time Traveller*, which - if used properly - could do much for pupils of all ages to increase their confidence at placing and sequencing events in history, and help them understand how dates can be ordered and recorded, not to mention provide opportunities for them to learn something about the events themselves.

## TIME TRAVELLER

The pack consists of three 40 track DFS discs in a sturdy plastic A4 folder with wallets, two substantial booklets, attractively produced 'Help' and 'Getting started' cards, as well as a registration card for ESM in furtherance of their lifetime guarantee: the program disc is copy protected, but the publishers undertake to replace it should it ever fail to work.

## USING THE PROGRAM

Booting this program disc, which is coloured differently from either of the other two data discs (one blank, one containing an existing database), leads to the main screen via a title screen.

Time Traveller is a form of menu-driven, historical database, which will allow text referring to events to be entered against the appropriate year any time from 9999 BC to 2100 AD. Intervals can be of years, decades and centuries, and one of the most appealing features of the product is the way in which it allows you to move - with assured visual impact - between them.

The program will 'count' intervals in years, so that Infant and lower Junior projects might reckon the spans of ancestors' lives as well as sixth forms calculate the years in each century when there were no wars!

The main screen makes use of colour and starts with the year on the left and a single 'tag' such as "Napoleon died" or "Death of Akhnaten" of up to 31 characters to its right. Some dozen or so of these years can appear on screen at once, and you scroll vertically forwards and backwards in time. One year - at the centre of the screen - is always "selected".

Across the foot of the screen, though, is a series of simple options allowing you to manipulate the data as appropriate. The legends for each of these options are highlighted in turn by moving the left and right cursor keys, and then selected by pressing Return.

## THE OPTIONS

The first of these menu options is the one to display the lines that link events in the database. Lines appear showing, say, the span of the Thirty Years War in the seventeenth century, or the duration of a lengthy strike. Such links are inserted in the editor, which is the last option (see later).

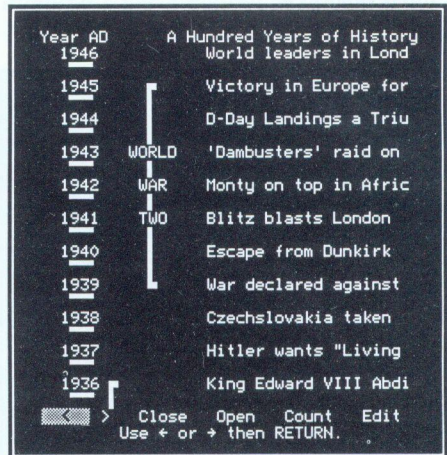
"Close" and "Open" dynamically expand or contract the time intervals in a clever and graphic way. Imagine that your database was showing the passage of time in decades, say: 1610, 1620, 1630 etc and 1640 was selected. When, for example, you opted for 'Close', 1640 and its text would remain while the rest of the screen would clear, and years and their text scroll in from the top and bottom of the screen in single years: 1638, 1639, 1640, 1641 etc. Conversely, 'Open' would slide away these single years (or decades) and 1440, 1540, 1640, 1740 and 1840 would roll up and down into view.

This is particularly useful in helping children to appreciate the ways in which dates can be sequenced, the more so because the base year stays at the vertical centre of the screen all the time.

The next option is one that counts dates and intervals of time. 'Count' is highlighted and selected and a '0' appears at the top right of the screen. As you scroll through the database, the number of years elapsed appears in place of this figure. It works across the BC-AD divide, too, which can be useful for that sort of calculation.

The option which leads to the most extensive activity is 'Edit'. This allows the links between years mentioned earlier to be entered, amended and deleted. Text to a maximum of one page

per year (16 lines, just less than 500 characters) can be entered, edited and deleted by means of the in-built text-handler, which is simple but adequate.



### Consecutive 'year' entries displayed

Function keys are responsible for controlling basic formatting etc. Although only the one line of text usually appears against each year on the main screen, the more extended information prepared in the Editor can amplify and provide background data on any year to a maximum of 100 pages per disc.

This information is revealed, incidentally, by selecting 'Open' when an interval of only one year is in force and you are on the year about which you require more information. The years are underlined to indicate that more information does exist.

You may want, for example, to place the months of the year on one of these pages, with events that happened placed alongside them for browsing. Alternatively, these could be the answers to a 'quiz' entry for the year on the main screen: "1941 - name the major European battles, dates and outcomes", for instance.

Selected periods, entries and details can be printed out from the editor - either to the width of one screen or the entire entry. Pupils often like the idea of adding what they have written using a program like this to project folders or books etc.

```
Year AD      A Hundred Years of History
1969 Man walks on the Moon

The world held its breath as
American astronaut Neil
Armstrong walked into the
history books. At 3.56 am BST,
on Monday July 21, Neil
Armstrong climbed down the
steps of the Eagle and stepped
onto the moon.
His words will be long remem-
bered: "That's one small step
for man, one giant leap for
mankind."
He was then joined by Edwin
"Buzz" Aldrin and together they
conducted vital experiments.

*****
Close  Edit  Count  Edit  Name
Use + or → then RETURN.
```

### Extended text entry for 1969

## EXTRAS

A separate 'system' utility, which is only obtainable once you have finished using the main Time Traveller program, allows you to alter such parameters as screen background colour, printer settings and to prepare fresh data discs.

Terminating Time Traveller, by the way, is achieved by the slightly less than satisfactory method of pressing the Break key. Although this is well handled, it is not something children should be encouraged to do as a rule, and there is a warning in the manual about the damage that can be done if Break is pressed during a disc access.

## MANUALS

There are two well laid out manuals accompanying Time Traveller. There is a

user guide dealing with routes through the suite itself, which could perhaps be clearer for the first time user, who is expected to use the crib card after becoming familiar with Time Traveller.

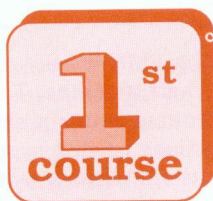
Secondly, there is a teacher's handbook, which - as seems to be more and more the practice nowadays - makes suggestions for wider use and includes a section on the sample database (the hundred years from 1891 to 1990), management and use in the classroom, preparatory and follow up activities and short chapters dealing with historical vocabulary and with the educational background and learning issues.

Teachers who take the trouble to read right through this booklet before using the package will find it pays off in the long run.

## CONCLUSIONS

*Time Traveller* is a simple idea presented attractively and imaginatively. With a maximum of just 200 records and the text dimensions referred to just now, it perhaps lacks the capacity for in-depth work at upper Secondary level - especially compared, say, with Soft Teach's Time Lines.

Although the product as supplied will not work over a network and is perhaps a trifle expensive, there are sufficient facilities, nearly all of which are well executed and error trapped, to make this a package which can be recommended for history and humanities work at all ages. It has one or two specific features (such as the 'Open' and 'Close' options) that can help to make difficult concepts more concrete. Given some of the wrong headed ideas about the teaching of history that abound at present, this can only be a good thing. **B**



# VDU and FX Calls (4)

by Mike Williams

I hope last month's instalment in this series of articles didn't leave you too downhearted. Unfortunately, some aspects of computing can be difficult to master, often because there is a lot of detail to understand. This month we will be dealing with much simpler matters, with a first look at \*FX calls.

The VDU drivers are the routines in the operating system which handle screen displays. All characters sent to the screen are intercepted first by the VDU drivers, and it is for this reason that a number of character codes can be made to produce other effects.

FX calls are also calls to routines in the operating system, and are known collectively as OSBYTE calls. Whereas a VDU instruction sends one or more characters to the VDU drivers, an FX call is like a jump to a subroutine in the operating system which performs a specific function. Although the FX format provides a simple route to these functions, a little more background information will not come amiss, and will help later.

All FX calls, are calls to the OSBYTE routine at &FFF4 in memory. The contents of the accumulator (set in A%) determine which OSBYTE routine (FX call) is to be used. Thus a call like:

```
*FX4
```

is the equivalent of:

```
A% = 4
```

```
CALL &FFF4
```

The majority of the more frequently used FX calls are nearly always accessed using

'\*FX', but the CALL format (or the similar USR function) does need to be used with some of the more complex forms of OSBYTE calls.

There is one further point on which to remind ourselves before we get started. Unlike VDU, \*FX is a star command, and thus no other instruction can follow a \*FX call on the same line. This can occasionally be inconvenient (indeed, if overlooked it can be the embarrassing source of an error). The alternative is to use the OSCLI function.

Thus referring to the example above, this could also be written:

```
OSCLI("FX4"): . . .
```

and any further instructions could then follow on the same line. This can be particularly useful in IF-THEN constructions.

Right, let's get started. Theoretically, there could be 256 different FX calls to cover (\*FX0 to \*FX255), but not all possible values are used, and a good few are mainly for machine code programmers, and thus really outside the scope of these articles. I will concentrate on those calls which I believe are most useful, and which will hopefully benefit from some more expansive discussion compared with the User Guide (or Welcome Guide depending on your BBC micro).

## PRINTER OUTPUT (FX3 & FX5)

This group of calls can cause some confusion, and the standard guides are incomplete in their documentation of

this call. The interaction between \*FX3 calls and the use of VDU2/VDU3 adds further confusion here. Normally, VDU2 and VDU3 are used to start output to a printer, and to stop output to a printer. Output will appear both on the screen and as hard copy as \*FX3,0 is the default setting. Using other \*FX3 calls enable the output stream to be redirected.

One problem which often arises is that of outputting to a printer only, and not to the screen as well. VDU1 can be used to specify that the next character to be sent should be sent to the printer only. This is all right for a few characters, but no good at all for longer text. The solution is to specify the poorly documented:

\*FX3,10

immediately before printer-only output is required, and to issue a:

\*FX3,0

as soon as normal output is to be resumed. Furthermore, if this form of \*FX3 is used, then this also enables and disables the printer - no VDU2 or VDU3 calls are required at all. It also disables the screen avoiding the need to use VDU1 to send characters to the printer only.

The related \*FX5 call is used to determine what type of printer you have in use, or to be more precise, how the printer is connected to your computer. These days, most printers use the parallel printer port connection, and thus a \*FX5,1 needs to be in effect before you attempt any printing. However, this is normally the default setting. Master and Compact users can save their choice of printer type in CMOS RAM, so that their choice is always known to their system. Model B users could include the appropriate instruction in a !BOOT file so that inserting a 'Start

Up' disc and pressing Shift-Break will similarly initialise their machine. The !BOOT file can, of course, contain any and all of the instructions needed to set up a system to the user's requirements after power up.

One often overlooked, but useful call here is \*FX5,0. Unlike many FX calls, the 0 does not signify a 'default'; in this instance it selects the 'printer sink' as the output device. This can be very useful for testing programs which are designed to output to a printer without wasting metres of paper.

Once \*FX5,0 has been specified, all output that would have gone to the printer is instead displayed on the screen (in printer layout as far as possible), and then 'lost' into the printer sink. It is particularly useful for testing programs using the \*FX3,10 call described earlier. With that in effect as well as \*FX5,0 all printer output is now redirected to the screen, but without affecting any output that would have gone to the screen anyway.

If that sounds a little confusing, wait until the next time you are testing a program which outputs to the printer and then try it out. It's only really suitable for character text output; graphics sequences intended for your printer will have a quite unpredictable effect if redirected to the screen (it's those VDU drivers again). Do remember too, to re-issue the appropriate \*FX5 call, after you have finished with the printer sink, to restore printer output as normal.

There is one other \*FX call which can also affect printer output and that is \*FX6, to set the so-called 'printer ignore'

character. The default setting is \*FX6,10 meaning that any new line (or linefeed) character (ASCII 10) in the output stream is ignored (and not passed on to the printer). This works fine for printers which automatically issue a linefeed character themselves on receipt of a carriage return.

In practice, this is not usually the best arrangement. The DIP switch settings on your printer will usually allow you to set it so that no automatic linefeed is issued - in other words the printer relies solely on the computer to tell it what to print. For the computer to work effectively with this setting, you must change the printer ignore character, usually to the null (ASCII zero) character. This is achieved with:

```
*FX6,0
```

and this call should either be issued immediately after the appropriate \*FX5 call for your system, or set as the default (Master and Compact). Even this is not foolproof, particularly when printing graphics and other fancy print effects - another alternative is to try \*FX6,255.

The effect of not doing this can be particularly noticeable when performing a graphics dump by the apparently random linefeeds which occur (where, by chance, a graphics character is the same as a linefeed character). Conversely, the sudden appearance of printout all on one line can often be traced to a mismatch between the \*FX6 setting on your micro, and the DIP switch setting on your printer.

## SOFT KEY OPTIONS

If you examine the coding of published programs, you will often see \*FX4 being used. This function controls the use of the four cursor keys (for up, down, left

and right), and the Copy key. Normally these keys perform cursor editing, but they can be used in two further ways.

Executing \*FX4,1 results in these five keys generating ASCII codes, just like other keys on the keyboard, and this is usually what a programmer requires (as we will see in a moment). Alternatively, executing \*FX4,2 enables these keys to be programmed as function keys 11 to 15 just like the standard function keys f0 to f9, plus Break (function key 11).

For example, you might want the cursor keys to be used to control a 'cross wire' on the screen in a graphics program. This could be accomplished as follows:

```
100 MODEL:*FX4,1
110 GCOL3,3:X=640:Y=512
120 X1=X:Y1=Y
130 VDU23,1,0;0;0;0;
140 PROCcross(X,Y)
150 REPEAT
160 code=INKEY(100)
170 IF code=136 THEN X1=X-4
180 IF code=137 THEN X1=X+4
190 IF code=138 THEN Y1=Y-4
200 IF code=139 THEN Y1=Y+4
210 PROCcross(X,Y)
220 PROCcross(X1,Y1)
230 X=X1:Y=Y1
240 UNTIL code=135
250 *FX4,0
260 VDU23,1,1;0;0;0;
270 END
280:
1000 DEF PROCcross(x,y)
1010 MOVE x-8,y:DRAW x+8,y
1020 MOVE x,y-8:DRAW x,y+8
1030 ENDPROC
```

If you enter and run this program you will see a small cross displayed in the centre of the screen. If you use the cursor keys, the cross will move correspondingly up,

down, left and right. If you press Copy the program will terminate. Let's have a look in more detail at some of the features in this program.

It uses mode 1 which means that the smallest unit of movement is 4 graphics units both vertically and horizontally. Using \*FX4,1 enables the cursor keys to generate ASCII codes as previously described.

Line 110 uses the GCOL instruction to do two jobs. The first value '3' selects Exclusive OR plotting which is essential for screen movement. Without going into detail, it means that plotting something on the screen makes it appear, while plotting the same object in exactly the same position makes it disappear. By repeating this two step process, but using a different position each time, an object can be made to appear to move on the screen. The second GCOL value, also '3' here, selects the colour for plotting, in the example 'white'.

Line 110 also initialise two variables, which indicate the current position of the cross wires, in the centre of the graphics screen. Two further variables, which are used temporarily for a 'new' position, are also initialised to the same values.

The following lines then switch off the flashing cursor (as described previously using a VDU call) and plots the cross wires for the first time. This uses a simple procedure for readability. The cross wires are drawn as two straight lines 16 graphics units long.

We then have the main loop, implemented here as a REPEAT-UNTIL loop. The object of the loop is to cycle fairly rapidly looking for any input on each pass. The ASCII code of any key

pressed is assigned to the variable code. The value in the INKEY function is not critical - increasing it will reduce sensitivity to any key press, reducing it will make the program more responsive.

In the next four lines, the program checks to see if any of the four cursor keys has been pressed to indicate a new position for the cross wires. However, rather than updating the values of X and Y directly, the program assigns the new position coordinates to variables X1 and Y1. The reason for this is that we need to remove the cross wires from the screen in the old position, and then replot them in the new position. Once this has been accomplished the values of X and Y can be updated to the new position ready for a next pass round the loop.

One further point, relevant to graphics and movement, is that it always pays to keep the time between removing the object from the screen and replotting as small as possible. This keeps any flicker to a minimum.

The loop, and in this case the program, can be terminated by pressing the Copy key. It would also be very easy to add further similar lines to the program, following line 200, allowing other key presses to be detected and acted upon as required. Note that before the program finally ends, a \*FX4 command is executed to return the cursor key functions to their normal state, and the flashing cursor is also restored to view. This is always desirable. Whenever any VDU or FX call is used to change from the default state of your machine, it is always advisable to reset the system to its defaults on exit. To convince yourself, just leave out line 250, and after running the program try and do some cursor editing.

*Continued on page 42*



# Wordwise User's Notebook (2)

*We have two items in this month's Wordwise Notebook: a short piece (following previous correspondence) on a method for generating a plus-or-minus sign ( $\pm$ ) in Wordwise from James Sugden, and a further contribution from David Else with a Segment program for Wordwise Plus users.*

## PLUS OR MINUS

I was reading recently about producing a  $\pm$  sign in View (see *Postbag* BEEBUG Vol.9 No.9), and was prompted to think about doing it for Wordwise. I started by trying a variation on the published method for View, viz by typing a plus sign, followed by backspace, and then an underline character. It works but the result in my opinion is very crude.

I tried several ways but I think the best result is obtained by using a superscript plus sign, then a backspace followed by a subscript minus sign. The codes for an Epson printer are:

Superscript	Escape 83,0
Subscript	Escape 83,1
Cancel above	Escape 84

and backspace is of course 8. A one-off plus or minus sign can thus be achieved with:

```
f1ES83,0f2+f1OC8f1ES83,1f2-f1ES84f2
```

where f1 and f2 represent the keys for start and end embedded codes.

A more elegant method, especially if two or more such signs are required is to define it as a print sequence, and this version takes care of correct justification by printing a space and then a backspace first:

If the sequence:

```
f1RPS9,8,27,83,0,43,27,83,1,8,45,27,84f2
```

is placed near the beginning, then whenever a  $\pm$  sign is required simply type:

```
<pad char |>f1OPS9f2
```

which will produce a plus or minus sign.

Tel. 0482 523854  
Membership No. E0029253  
Your ref. M/D4/09

15 Caphurst Road,  
Colwyn Bay,  
Clwyd,  
LL22 6JP.  
20th May 1991

Mr.M.Williams,  
BEEBUG Limited,  
117 Hatfield Road,  
St. Albans,  
Hertfordshire,  
AL1 4JS.

Dear Mr Williams,

### WORDWISE RELATED ITEMS OF INTEREST

Thank you for your letter of 8th May. In my letter of 21st April 1991 I omitted to include the fact that once my BOOT file had been run,

*Standard letter layout expected by stripping program*

## ADDRESS STRIPPING

The Wordwise Plus segment program listed here is designed to strip the recipient's address from a standard letter layout and then reformat this for printing onto an address label (or envelope). The program assumes a standard layout, such as that discussed in last month's Wordwise Notebook, where your own address appears above (and to the right) of the address of the recipient (see illustration above).

The program relies on the '19' followed by two digits (for the year), which comes before the recipient's address to strip away the top half of the document, and the 'Dear' at the start of the letter to strip away the remainder.

The address is then printed with the start of each line moving one character to the right as it should be (in my opinion). It

## Wordwise User's Notebook

also asks if you wish to 'Strip the text', and only if you reply with a 'Y' or 'y' does it proceed to strip the text from the letter.

To do this, the letter should be loaded into memory as normal with Wordwise, and the program loaded into one of the other segments. Pressing the appropriate function key will then activate the program. While writing a number of letters, the program can remain in its allotted segment, and either be used to print each address after each letter, or each letter in turn can be loaded at the end of a session, and the addresses printed one after the other.

```
PRINT
DOTHIS
VDU129,157,131,141
PRINT"Strip the text?      "
TIMES2
X$=GCK$
VDU7
IFX$<>"Y"A.X$<>"y"THEN GOTOa
SELECT TEXT
CURSOR TOP
```

```
FIND "Dear"
FKEY3
CURSOR BOTTOM
FKEY3
DELETE MARKED
TYPE "f1ES64f2"
CURSOR TOP
FKEY3
FIND"19##"
CURSOR AT0
CURSOR DOWN
FKEY3
DELETE MARKED
TYPE CHR$13
CURSOR TOP
TYPE "f1ES64f1ES40f1ES51,45f1*FX155,36f2"
CURSOR DOWN
A%=0
REPEAT
CURSOR AT0
TYPE "f1LM20+"+STR$(A%)
TYPE "f2"
CURSOR DOWN
A%=A%+1
UNTIL A%=9
.a
DISPLAY
END
```

**B**

### 1st Course (continued from page 40)

There are two final points worth making about this program. I have chosen to increment or decrement the co-ordinates of the cross wires in steps of 4 (see lines 170 to 200), the minimum sensible in mode 1. If you feel that the resulting movement of the cross wires on the screen is two slow, then increase the value in each of these lines, but only in multiples of four.

Secondly, as the program stands there is nothing to stop the cross wires being moved off the screen altogether. This is not really a good idea, and can be easily avoided. For example, the left-hand edge of the screen is where  $X=0$  and thus  $X$  should only be decremented if  $X>0$ . Such a condition has a value of TRUE (-1) or FALSE (0), and we can use the numerical equivalent to provide a simple modification. Thus the four lines in question would become:

```
170 IF code=136 THEN X1=X+4*(X>0)
180 IF code=137 THEN X1=X-4*(X<1276)
190 IF code=138 THEN Y1=Y+4*(Y>0)
200 IF code=139 THEN Y1=Y-4*(Y<1020)
```

Notice that because TRUE is equivalent to -1, the incrementing plus and minus signs have become reversed. The values of 1276 and 1020 represent the largest values that can be reached horizontally and vertically starting from zero and incrementing in steps of 4 (i.e. in the ranges 0 to 1279 and 0 to 1023).

We have wandered a long way from FX calls here, but on this occasion I thought it worthwhile to give an example of an FX call in use, and to explore it in some detail. More on FX calls next month.

**B**

# Printing Scientific Characters with Word Processors (Part 3)

by Gareth Leyshon

This month we look at some of the technical details of the program given last month, and the changes which may be needed to cope with different printers.

## OTHER PRINTER CODES

Lines 12000-12070 contain data essential to the program's correct working. At line 12020 you must store:

1. the number of bytes your printer requires to define the matrix of a UDG (9 for a KX-P1081, 11 for many Epson-compatible printers);
2. the number of lines to a page (66 for standard printer paper, or 70 for A4 size); and
3. the number of special control characters you wish to implement (default 13).

Line 12040 stores three sets of printer codes. `CHR$1+CHR$27+CHR$1,Z$+CHR$38+CHR$1,Z$+CHR$64,Z$+CHR$67+CHR$1` is the standard beginning of any Escape sequence and may be abbreviated to 'Z\$' at all other points. This should be followed (after a comma) by your printer's sequence for 'define UDG', 'clear all UDGs' and then the sequence for 'set page length'. For example, line 12040 would become:

```
12040 DATA CHR$1+CHR$27+CHR$1,Z$+CHR$38+CHR$1,Z$+CHR$64,Z$+CHR$67+CHR$1
```

for an Epson FX-80.

Line 12060 gives the code for your 'default' printer setup, followed by the code to revert from 'default' mode to the

printer's standard mode - the 'pre-print' and 'post-print' codes. Set both of these to null characters (`CHR$0,CHR$0`) if your printer does not have this facility. Lines 13000-13270 contain the data for intercepting command characters and sending codes to the printer. The number of these is set in line 12020, and they are of the form:

```
DATA intercept character code, replacement control sequence
```

where the intercept code is the ASCII code and the replacement sequence of the form `Z$+CHR$(m)+CHR$(n)` etc. (see Technical Notes). If you are considering adding a code to select either an international character set or super/subscript mode, consider whether the job can more easily be done by defining another UDG to look like the desired character. Note that the sequences given in lines 13130 to 13200 have no equivalents on an FX80.

## TECHNICAL NOTES

1. Pound patch: There are at least two ways to perform a pound-patch with Edit-Plus. One (implemented in this version) is to treat '£' as a special character so that it prints on screen with no modification but when printing out, is replaced with the appropriate code. Character 185 is set up to look like a backwards apostrophe (') on screen but has the printer definition for a hash. Thus when a hash is printed on screen it is patched to character 185, and the real 185 is patched to the pound code (a backwards apostrophe to the printer).

## Printing Scientific Characters with Word Processors

The same effect can be achieved by NOT redefining anything in the printer (useful if its buffer is full) but changing the pound's screen appearance to a back-apostrophe, and making the hash look like a pound sign (though screen and printer won't tally with the keyboard in this case). You can, of course, redefine another character to look like a hash on both screen and printer.

If you need to place printer codes directly into the program, I recommend certain spots for VDU commands - these are given in Table 1. Note that when using the PRINT CHR\$( ) construction, a carriage return will be sent to the printer at the end of it unless suppressed by a semicolon.

Some printers print italics using an italic character set with codes greater than 127. In this case, redefining high-number characters may disrupt the italic character set, so you are advised to use italics with caution. Take care that you do not exceed the maximum number of UDGs the printer can store. If this happens, you may see italic characters appear after the Xs on the printout during definition. If you suspect the buffer is not clear, momentarily switch off the printer. Extensive use is made of the EVAL( ) function (on strings). This is mainly due to the need to store control codes of unknown format and length in DATA statements. When a string of CHR\$( ) expressions joined by + is evaluated, it gives a string of control codes. If a variable such as the Z\$ header is to be included in a string to be evaluated, this 'sub-string' must already be evaluated. That is, if we have A\$="Hello", B\$="A\$", C\$="B\$", we find

that EVAL(C\$)="A\$". To get Hello we must first say B\$=EVAL(B\$) or possibly, EVAL(EVAL(C\$))="Hello".

If the structure of Edit-Plus does not allow you to insert the right codes to define UDGs on your printer, you may wish to incorporate VDU statements into the program. This table gives the best positions to do this:

- a) Before printing any UDG definitions:  
Place between EVAL(K\$) and CHR\$3 on line 1020.
  - b) Before printing each UDG definition:  
Change line 1180 and put the codes in N\$.
  - c) Before sending each byte of a UDG definition.  
The first byte is put into N\$ in line 1180, subsequent bytes in line 1190.
- N.B. Here, I% is the character in question and J% the data byte.
- d) Immediately after the last UDG definition:  
In the VDU of line 1040.

Pre- and post-print codes for the body of the text should be incorporated into DATA statements as explained in the article.

**Table 1**

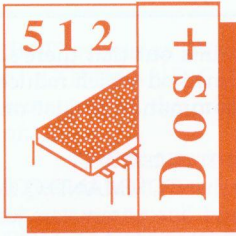
You may wish to add some control characters to the list from 13000 onwards which need to toggle between an 'ON' state and an 'OFF' state, like the bold and underline functions. One way of doing it is to use two separate commands for on and off. If you wish to set a toggle flag, use one of the variables G%, H%, O%, LOCALising it in line 1310.

Toggle it in line 1540, using, say:

H%=H%EOR (- (R%=x) )

where x is the position of the toggle command in the DATA list. It will alternate between 0 and 1. H% should be included in the control code in the DATA statement to be EVALuated during the run. The toggle will normally form the third byte of a sequence, but where two

*Continued on page 49*



# 512 Forum

by Robin Burton

Some possible Forum topics tend to be ignored

because I think of them as obvious, or basic facts everyone probably already knows.

Perhaps they're not. I guess from the response to a Forum a couple of months ago that some subjects are perhaps worth covering in more detail after all.

## COMMAND.COM REVISITED

You'll recall that recently I mentioned loading a second copy of COMMAND.COM as a fix for some programs that normally won't run in the 512. It's not a new discovery, quite the opposite in fact, and it was explained in one of the first Forums nearly three years ago. Obviously some of you had forgotten it, or perhaps hadn't even heard of it, so it turned out to be well worth covering again. It also prompted a letter which provides the subject for this month's Forum.

First, thanks to Doug Blyth from Epsom for writing to tell me that *TAXCALC*, a PC program from Which? magazine responds to this fix and now runs in his 512. I should mention that Doug had written to me quite some time ago asking if I knew if this program could be made to work in the 512, but at the time I wasn't able to help. The reason was simply that Doug didn't send a copy of the program with his query, while I (incorrectly as is now obvious) had assumed he'd tried reloading COMMAND.COM because it's the first thing I try if a program fails to run. It just shows that we can all forget basics and

that the cure for a problem might be so simple we overlook it.

I'm also indebted to another member, David Harper, first for confirming that this fix also works for version 4.00 of the shareware spreadsheet *As-Easy-As*, and for reminding me about another point I haven't covered in the Forum.

There wasn't enough space in the last issue to cover COMMAND.COM, but in any case the point that David raises deserves a separate article of its own to do it justice. Here it is, with some extra detail and explanation added for the benefit of newer 512 users, plus some example files in case you want to try things for yourself.

## COMMAND LINE SWITCHES

A useful feature of DOS Plus which almost everyone uses from time to time is the command line switch. The most frequently used are probably '/P' after a 'TYPE' or a 'DIR' command to output the display in paged mode, a '/W' for a wide directory display, or '/S', which enables access to system files which otherwise would be ignored in operations such as 'DIR' or 'COPY'.

What DOS Plus users might not realise however, is that these switches are not a DOS standard, by which I mean they aren't provided by PC-DOS or MS-DOS. One illustration is something referred to in BEEBUG Vol.9 No.7, the MS-DOS utility MORE. MS-DOS users are obliged to employ piping through an extra program to produce exactly the same effect as the much more convenient '/P' used in DOS Plus. In fact command line switches are common to all Digital

Research operating systems, from the 8-bit CP/M of ten years ago up to and including the latest version of DR-DOS.

What does this have to do with re-loading COMMAND.COM? Well, you'll remember that 'EXIT', terminates a second copy of COMMAND and returns you to a single DOS command shell, freeing 30K of memory as it does so. David points out that if you use the '/C' switch when the extra copy of COMMAND is loaded the 'EXIT' is automatic, saving you the need to (remember to) do it. Here's how it works.

Suppose we want to run a program which requires a second copy of COMMAND.COM to be loaded and we want this second copy to be terminated automatically at the end of the program run. First let's point out that it doesn't matter whether the program in question is just a utility that does a single job and then terminates, or is a more substantial application such as *As-Easy-As*, which you might use for hours before returning to DOS. In all cases the method is exactly the same.

For illustration we'll use a text editor, and with the usual flair and imagination for which the Forum is well known we'll call the program *EDIT*. The file to be edited? What else but 'TEXT.DOC'! Assuming drive A: as the current drive, doing things manually (i.e. the long way) the sequence of commands would be:

```
A>COMMAND
A>EDIT TEXT.DOC
```

at which point the application would load, followed by the file and we'd then proceed to edit it. When we finished the job we'd save the file again then exit the editor back to the DOS command line. Next we must enter:

```
A>EXIT
```

to remove the second copy of COMMAND.COM and free its 30K of

memory. David points out that there's a much more direct method which reduces the above three commands to just one entry, like this:

```
A>COMMAND /C EDIT TEXT.DOC
```

If you didn't know that COMMAND.COM could take command line parameters just like many other DOS programs, you do now. In fact, it's virtually the same technique as that used by DOS itself when the system first starts up. The major difference when you boot the system is that you never get to see the command line, but if you did, it would be 'COMMAND AUTOEXEC'.

Of course, in this case DOS has first made sure that there is an AUTOEXEC.BAT file, and if there isn't then COMMAND.COM is simply loaded without a command tail. You might have noticed when the system starts up (though you'll only be aware of it using floppies) that after the initial XIOS copyright display the system very clearly scans the disc again, and there's a distinct delay while it finds out whether or not there's an AUTOEXEC file to load. Of course this is just information for interest, it's not directly useful, so now back to our example.

To understand how the '/C' switch works we'll take the command entry apart and see what happens to it. The first point is that one copy of COMMAND.COM is already resident of course, and this permanent copy processes all manual or batch file command input, so the command is initially treated just like any other that calls a program from disc.

As usual, the first item on the line (up to the first space) is the name of the executable file which is to be loaded, in this case COMMAND. As normal DOS looks for and loads a file with that filename and a 'COM', 'CMD', 'EXE' or

'BAT' extension. When the second copy of COMMAND.COM is loaded the remainder of the original line, consisting of:

```
/C EDIT TEXT.DOC
```

is passed to the newly loaded program as its own command tail. The second COMMAND.COM therefore potentially has three parameters to process. The first one is the part we're most interested in here, the '/C'. That tells COMMAND that when the next action or operation ends, the program should remove itself from memory.

The remainder of the original command line is then processed by (the second) COMMAND.COM in the same way as it would be if it had been entered directly from the keyboard. The next parameter is again the name of a program, which in turn is loaded in the normal manner and according to the usual rules. The final parameter, the filename, is then passed to this new program as its command tail.

You can see that, although the command line is entered in one, in fact it is processed piecemeal so that each parameter is treated almost like a separate command. It is therefore important to remember that the position of the '/C' is critical. It must immediately follow COMMAND, otherwise it might be passed to the application as one of its parameters, with numerous potential results, none of which would be the right one.

Naturally all this is transparent to the user sitting at the machine, so when editing is complete and we finally leave the editor, the extra copy of COMMAND.COM then actions the original '/C' parameter, which it has remembered throughout. The extra copy of COMMAND.COM therefore terminates immediately with no further user input and all the previously occupied RAM is freed.

Another point which you'll notice if you use this technique is that the usual copyright display seen when a second copy of COMMAND is loaded, which appears as:

```
DOS Plus Version 2.1
```

```
Copyright Digital Research (c) 1985,1986
```

isn't displayed when the '/C' switch is used. This is obviously of little importance during manual entry, but it clearly points the way to other possibilities offered by the '/C' switch.

### BATCH FILES AGAIN

A potential problem in batch files in the 512 is the fact that some programs will only work if a second COMMAND.COM is loaded. Without the '/C' switch the need for a second COMMAND shell appears to restrict the use of such programs in batch files unless you load COMMAND manually before you start the batch file. You can prove this yourself if you don't see why by creating and running the example file I'll explain in a moment.

Before you start, however, issue a 'BACKG' command and note how much free RAM you have in your machine if you're not sure. In a standard 512 this should be 358K, in an expanded machine, 614K. Obviously, if you use TSRs as in your 512 these will reduce the free memory figures slightly.

Having done this, from the command line type:

```
COPY CON DEMO.BAT
```

followed by Return. This will open the file 'DEMO.BAT' for output to the current directory on the current drive. Next, type in the lines shown below, following each line with a Return.

```
COMMAND
```

```
ECHO Hello Mum
```

```
EXIT
```

On or after the end of the last line press Ctrl-Z and another Return, which will close the file and return you to the DOS prompt. Make sure that COMMAND.COM is in a current path, then run the file by entering 'DEMO'.

You'll see that a second COMMAND.COM loads by the fact that it announces itself with the copyright message shown above. However, that's all you'll see for the moment, because execution of the batch file is then immediately suspended and you're presented with a normal DOS prompt (from which you can execute programs as normal).

The reason for this effect is that the initial call to the batch file was made through the original, permanent copy of COMMAND.COM. However, as soon as the second copy is loaded and for as long as it remains active the original COMMAND.COM is no longer in control, hence it cannot continue with execution of the batch file.

To prove this, next enter 'BACKG' again and you'll see from the reduction in free memory (30K less than before) that the second copy of COMMAND is in RAM. 'BACKG' is of course a transient program, so running it clearly demonstrates that you're using a normal command line, but the rest of this test will also show that you're doing so from within a batch file.

You can at this stage execute any command you'd normally use on the command line, including execution of yet more batch files. What's more, one of those batch files could be 'DEMO' (again) if you like. This you can also prove, but before trying this, read the rest of the article, for reasons which will become clear later.

For now just type 'EXIT', followed by Return and you'll see that immediately

following your entry the line 'Hello mum' appears, followed by the echoed display then yet another 'EXIT', this one being the one in the batch file. You can see that the 'EXIT' in the batch file, which you might have expected to remove the second copy of COMMAND.COM, serves no useful purpose in fact, because it's too late by the time it is processed!

Given the explanation of what happened when the file began execution, you can probably guess what's now happened. As soon as the 'EXIT' is entered manually the second copy of COMMAND terminates. After this control returns to the original, permanent copy, which then resumes processing of the batch file from the point it had reached when control was taken away from it.

Now create another batch file called, say 'DEMO2', but this time include a '/C' and a 'BACKG' on the COMMAND line, like this:

```
COMMAND /C BACKG
ECHO Hello Mum
EXIT
```

When you run it you'll immediately see the difference. 'BACKG' is run from the second COMMAND.COM as the free RAM display will prove, but that copy then terminates immediately and execution of the batch file will resume automatically.

The benefits of this are obvious, you can after all run programs that need a second copy of COMMAND.COM from batch files successfully without manual intervention. Secondly, and as a side effect of the '/C' parameter, you will avoid cluttering the screen with the DOS Plus copyright display in the middle of the execution of a batch file.

### LIMITATIONS

To round off this month I'll return to the point about running batch files from a



second shell, itself created within a batch file. Given normal (that is, sensible) use there's no problem at all, apart from that of the reduction in free RAM in unexpanded machines. However, as an experiment I tried running 'DEMO.BAT' repeatedly and on the 21st run I received the message:


```
Internal error 08
Not enough memory to load command.com
```

Not surprising really! 19 'EXIT's later I was nearly back to where I'd started, each 'EXIT' producing the expected effect, but I should warn you that making DOS tie itself in knots like this is almost certain to crash the system sooner or later. Sure enough, on the 20th 'EXIT' it did crash, quite gently really, with a permanent display of:

```
Not enough memory to load command.com
```

At this point nothing had any further effect, except that repeated Ctrl-Cs caused the message to be repeated. The system was obviously quite prepared to continue like this ad infinitum.

The only surprise to me was that the system proved to be far more tolerant of this type of abuse than I'd expected. I'd anticipated a crash at any moment after the fourth nested 'DEMO' run. I didn't bother to find out how many I could actually get away with safely, it's certainly more than three and quite enough for any sensible purpose, so I'll leave it to you to investigate the limit if you're interested.

Obviously what happens ultimately is that somewhere in DOS a data store overflows (probably one of the system stacks) with the result that the system loses its way and finally can't remember how to get out again. 

### **Printing Scientific Characters (continued from page 44)**


different ESC codes switch a function, use ESC followed by the off code plus the (difference between off and on) times the toggle variable.

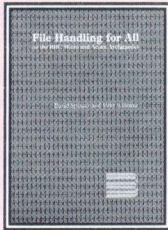
#### **CONCLUSION**

None of the methods I have presented in these articles is perfectly WYSIWYG. It's a case of choosing the most suitable method. Where you are writing an essay which needs just a few symbols, View can provide a limited number of on-screen characters which actually look right, though I don't know how to set the character definitions for screen and printer without running a program or executing a VDU code file beforehand.

Inter-Word makes the pre-print code handling easier, but you don't have the clarity of having the characters represented on screen. However, if the

majority of the new characters can be achieved using super- or subscripts, knowing how to get round the NLQ 'full sized only' constraint could make View or Inter-Word more attractive.

Edit comes closest to truly representing the final printout in terms of characters - especially when they are not obtainable by super- or subscripts - but you do not have the benefits of on-screen justification, underlining or boldening. However, the use of single characters rather than three-character codes for underlining, etc., does reduce the problem of the coded page looking confused. I feel that if you have a number of new characters in your document, the loss of visible underlining is worth the gain in symbols. Edit-Plus should simplify the use of Edit for this considerably. 



## File Handling for All on the BBC Micro and Acorn Archimedes by David Spencer and Mike Williams

Computers are often used for file handling applications yet this is a subject which computer users find difficult when it comes to developing their own programs. *File Handling for All* aims to change that by providing an extensive and comprehensive introduction to the writing of file handling programs with particular reference to Basic.

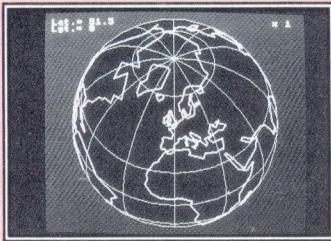
*File Handling for All*, written by highly experienced authors and programmers David Spencer and Mike Williams, offers 144 pages of text supported by many useful program listings. It is aimed at Basic programmers, beginners and advanced users, and anybody interested in File Handling and Databases on the Beeb and the Arc. However, all the file handling concepts discussed are relevant to most computer systems, making this a suitable introduction to file handling for all.

The book starts with an introduction to the basic principles of file handling, and in the following chapters develops an in-depth look at the handling of different types of files e.g. serial files, indexed files, direct access files, and searching and sorting. A separate chapter is devoted to hierarchical and relational database design, and the book concludes with a chapter of practical advice on how best to develop file handling programs.

The topics covered by the book include:

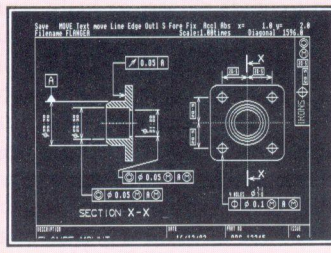
- Card Index Files, Serial Files, File Headers, Disc and Record Buffering, Using Pointers, Indexing Files, Searching Techniques, Hashing Functions, Sorting Methods, Testing and Debugging, Networking Conflicts, File System Calls

The additional disc contains complete working programs based on the routines described in the book and a copy of Filer, a full-feature Database program originally published in BEEBUG magazine.



### 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
- 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



### ASTAAD

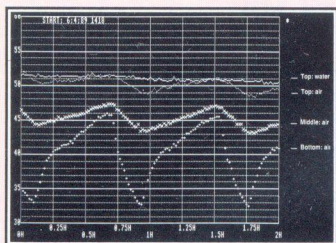
**Enhanced ASTAAD CAD program for the Master, offering the following features:**

- \* 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
EDIKIT (EPROM)	1451a	£ 7.75			
EDIKIT (40/80T DFS)	1450a	£ 5.75	EDIKIT (3.5" ADFS)	1452a	£ 5.75
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

*Please add p&p*

## 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
- LABEL PROCESSOR** - for designing and printing labels on Epson compatible printers
- SHARE INVESTOR** - assists decision making when buying and selling shares.

## 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.

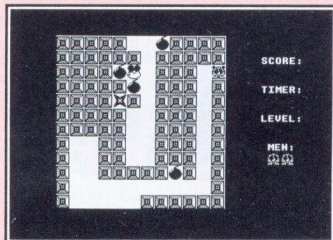
**PITFALL PETE** - Collect all the diamonds on the screen, but try not to trap yourself when you dislodge the many boulders on your way.

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

**MINEMFIELD** - 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.



## 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.

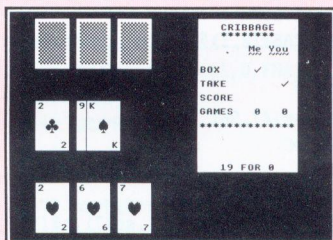
**ELEVENENSES** - another popular version of Patience - lay down cards on the table in three by three grid and start turning them over until they add up to eleven.

**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.



	Stock Code	Price		Stock Code	Price
<b>Arcade Games</b> (40/80 track DFS)	PAG1a	£ 5.95	<b>Arcade Games</b> (3.5" ADFS)	PAG2a	£ 5.95
<b>Board Games</b> (40/80 track DFS)	PBG1a	£ 5.95	<b>Board Games</b> (3.5" ADFS)	PBG2a	£ 5.95
<b>File Handling for All Book</b>	BK02b	£ 9.95			
<b>File Handling for All Disc</b> (40/80T DFS)	BK05a	£ 4.75	<b>File Handling for All Disc</b> (3.5" ADFS)	BK07a	£ 4.75
<b>Joint Offer book and disc</b> (40/80T DFS)	BK04b	£ 11.95	<b>Joint Offer book and disc</b> (3.5" ADFS)	BK06b	£ 11.95

Please add p&p. UK: £1.00 first item (50p for every additional item), Europe and Eire: £1.60 first item (80p every additional item), Elsewhere: £2.60 first item (£1.30 every additional item)

## Chinese Tangrams (continued from page 10)

```

2580 DEF PROCsqu(p%)
2590 MOVE x%,y%
2600 PLOT 0,c%-s%,s%+c%
2610 PLOT p%,-2*c%,-2*s%
2620 PLOT p%,2*s%,-2*c%
2630 PLOT p%,2*c%,2*s%
2640 PLOT p%,-2*s%,2*c%
2650 ENDPROC
2660 :
2670 DEF PROCrhom(p%)
2680 MOVE x%,y%
2690 PLOT 0,-s%/2,c%/2
2700 PLOT p%,-c%,-s%
2710 PLOT p%,c%+s%,s%-c%
2720 PLOT p%,c%,s%
2730 PLOT p%,-c%-s%,c%-s%
2740 ENDPROC
2750 :
2760 DEF PROCend
2770 IF G%<>48 ENDPROC
2780 VDU4,23,1,0;0;0;0;0;
2790 PRINTTAB(18,1)SPC21
2800 VDU5:PROCfillmt(2)
2810 REPEAT UNTIL INKEY=99
2820 PROCfillmt(0)
2830 FOR I%=1 TO 7
2840 PROCset:NEXT
2850 VDU4:COLOUR 3
2860 PRINTTAB(18,1)"Key 1-7 to select T
an"
2870 VDU5,7
2880 PROCget(47,56)
2890 ENDPROC
2900 :
2910 DEF PROCinit
2920 DIM x%(7),y%(7),a(7),sc(7),n$(12)
2930 VDU19,1,2;0;
2940 I%=0:q=SQR2
2950 FOR i%=1 TO 7:READ sc(i%):NEXT
2960 FOR j%=1 TO 12:READ n$(j%):NEXT
2970 ENDPROC
2980 :
2990 DATA 128,128,64*q,64,64,32*q,64*q
3000 DATA The Square,Candle-stick,Scott
ie Dog,Siamese Cat,Sad Vulture,Man Runni
ng,Man Kicking,Man Sitting,A Polite Man,
Man Riding,Sailing Boat,Napoleon ?
3010 DATA "1 2 3 4 5 6 7",648,84,
0,772,168,180,876,88,0,980,120,135,1052,
120,225,1154,120,270,1232,140,-90

```

```

3020 DATA "Construct Tangram here",52,0
,135,420,0,-135,228,464,0,0,504,45,476,5
04,-45,0,272,90,496,272,118
3030 REM USE FOR VARIABLE RESTORE
3040 DATA -16,64,-45,164,64,45,72,244,0
,-64,200,225,116,200,135,28,200,0,164,20
0,180
3050 DATA 64,-88,0,136,104,90,68,264,18
0,0,44,270,0,172,270,68,316,0,72,468,116
3060 DATA 212,0,315,24,0,315,180,24,135
,-16,140,225,-100,128,270,-72,28,45,372,
16,159
3070 DATA 124,32,135,120,220,225,124,12
4,180,112,412,270,244,412,90,180,348,-45
,-48,0,171
3080 DATA 124,176,270,32,168,225,224,21
2,315,144,0,0,336,76,45,336,168,0,0,40,2
25
3090 DATA 24,212,0,64,120,-45,0,388,-22
,24,0,225,4,384,157,36,76,-45,204,116,90
3100 DATA 176,244,-8,108,248,171,16,104
,-98,0,0,84,304,164,217,224,412,-27,188,
136,216
3110 DATA 256,64,193,312,84,13,88,28,13
,0,0,193,68,224,104,244,244,-59,132,132,
149
3120 DATA 36,280,180,96,280,0,264,428,-
70,48,0,135,232,344,180,120,452,-13,0,11
6,315
3130 DATA 0,56,277,188,20,315,148,176,2
25,136,0,0,-60,136,231,156,264,52,348,32
,157
3140 DATA 64,40,-3,196,164,90,64,260,27
0,68,36,180,252,176,135,252,80,0,164,0,4
5
3150 DATA 116,172,315,104,0,234,108,328
,135,48,172,271,0,180,225,116,104,322,17
6,312,180
3160 :
3170 DEF PROCerr
3180 VDU22,7:PRINT'
3190 REPORT:PRINT" at line ";ERL
3200 ENDPROC
3210 :
3220 DEF PROCpage
3230 *FX3,2
3240 *KEY0 *T.|MF.A%=0TO(TOP-PA.)S.4:A%
!&E0=A%!PA.:N.|MPA.=&E0|MO.|MRUN|M
3250 *FX138,0,128
3260 ENDPROC

```

B

## Workshop - Three Dimensional Graphs (continued from page 14)

```

+p1*s,y,x+p3*s,y+s):ENDPROC
2000 IF p2>=0 AND p4>=0 THEN PROCjoin(x
,y+p2*s,x+s,y+p4*s):ENDPROC
2010 IF p1>=0 AND p2>=0 THEN PROCjoin(x
+p1*s,y,x,y+p2*s):ENDPROC
2020 IF p1>=0 AND p4>=0 THEN PROCjoin(x
+p1*s,y,x+s,y+p4*s):ENDPROC
2030 IF p2>=0 AND p3>=0 THEN PROCjoin(x
,y+p2*s,x+p3*s,y+s):ENDPROC
2040 IF p3>=0 AND p4>=0 THEN PROCjoin(x
+p3*s,y+s,x+s,y+p4*s):ENDPROC
2050 ENDPROC
2060 r1=1-p1:r2=1-p2:r3=1-p3:r4=1-p4
2070 d1=SQR(p1*p1+p2*p2)+SQR(r3*r3+r4*r
4)
2080 d2=SQR(r2*r2+p3*p3)+SQR(r1*r1+p4*p
4)
2090 IF d1<d2 THEN PROCjoin(x+p1*s,y,x,
y+p2*s):PROCjoin(x+p3*s,y+s,x+s,y+p4*s):
ENDPROC
2100 PROCjoin(x,y+p2*s,x+p3*s,y+s)
2110 PROCjoin(x+p1*s,y,x+s,y+p4*s)
2120 ENDPROC
2130 :
2140 DEF PROCjoin(a,b,c,d)
2150 MOVE a,b:DRAW c,d:ENDPROC
2160 :
2170 DEF FNin(a,b,c)
2180 IF SGN(b-a)=SGN(b-c) THEN =-1
2190 =(b-a)/(c-a)
2200 :
2210 DEF FNx(xo)=xscale*xo+x0
2220 DEFFNy(yo)=yscale*yo+y0
2230 :
2240 DEF PROCplot3D
2250 xscale=639/((x1-x0)*cos)
2260 yscale=639/((y1-y0)*cos)
2270 k=k/SQR(xscale*yscale)
2280 sx0=-x0*xscale:sx1=(x1-x0)*xscale
2290 sy0=-y0*yscale:sy1=(y1-y0)*yscale
2300 dx=(x1-x0)/n:dy=(y1-y0)/n
2310 zscale=200/(max-min)*(1+k*SQR((x1-
x0)^2+(y1-y0)^2))
2320 REM back right
2330 FOR Y=n TO 0 STEP -1:y=y0+Y*dy
2340 sy=Y*dy*yscale
2350 PROCscoords(sx1,sy,FNz(x1,y))

```

```

2360 IF Y=n THEN oldx=px:oldy=py:NEXT
2370 PROCjoin3D
2380 NEXT Y
2390 REM main graph
2400 FOR Y=n TO 0 STEP -1:y=y0+Y*dy
2410 sy=(y-y0)*yscale
2420 FOR X=0 TO n:x=x0+X*dx
2430 sx=(x-x0)*xscale
2440 PROCscoords(sx,sy,FNz(x,y))
2450 IF X=0 THEN oldx=px:oldy=py:NEXT
2460 PROCjoin3D:NEXT X:NEXT Y
2470 REM left front
2480 FOR Y=n TO 0 STEP -1:y=y0+Y*dy
2490 sy=Y*dy*yscale
2500 PROCscoords(0,sy,FNz(x0,y))
2510 IF Y=n THEN oldx=px:oldy=py:NEXT
2520 PROCjoin3D
2530 NEXT Y
2540 REM box base
2550 PROCscoords(0,sy1,FNz(x0,y1))
2560 oldx=px:oldy=py
2570 PROCscoords(0,sy1,min):PROCjoin3D
2580 PROCscoords(0,0,min):PROCjoin3D
2590 PROCscoords(0,0,FNz(x0,y0)):PROCjo
in3D
2600 PROCscoords(0,0,min):oldx=px:oldy=
py
2610 PROCscoords(sx1,0,min):PROCjoin3D
2620 PROCscoords(sx1,0,FNz(x1,y0))
2630 PROCjoin3D
2640 ENDPROC
2650 :
2660 DEF PROCscoords(sx,sy,z)
2670 px=640+(sx*cos-sy*cos)
2680 py=(sy*sin+sx*sin+zscale*z)/(1+k*S
QR((sx-sx0)^2+(sy-sy0)^2))
2690 ENDPROC
2700 :
2710 DEF PROCjoin3D
2720 LOCAL X%,Y%,s
2730 IF px=oldx THEN MOVE oldx,oldy:DRA
W px,py:oldx=px:oldy=py:ENDPROC
2740 s=(py-oldy)/(px-oldx)
2750 FOR X%=oldx TO px STEP 2
2760 Y%=s*(X%-px)+py
2770 MOVE X%,0:PLOT 15,X%,Y%
2780 PLOT 69,X%,Y%
2790 NEXT:oldx=px:oldy=py:ENDPROC

```

```

220 LDY #string DIV &100
230 JMP oscli
240 .string EQU8 "BASIC":EQU8 13
250 :
260 .setvecs LDA wordv:STA oldv
270 LDA wordv+1:STA oldv+1
280 LDA wrchv :STA oldv+2
290 LDA wrchv+1:STA oldv+3
300 SEI:LDA #entry MOD &100:STA wordv
310 LDA #entry DIV &100:STA wordv+1
320 LDA #rts MOD &100:STA wrchv
330 LDA #rts DIV &100:STA wrchv+1
340 CLI:RTS
350 :
360 .findend
370 LDA #131:JSR osbyte \ read OSHWM
380 INY \ add INX for VBAS
390 STX zp:STX zp+2
400 STY zp+1:STY zp+3:LDY #0
410 .loop LDA (zp+2),Y:BEQ foundit
420 INC zp+2:BNE loop
430 INC zp+3:BNE loop
440 .foundit RTS
450 :
460 .movup
470 LDA #&84:JSR osbyte \ read HIMEM
480 TXA:SEC:SBC #16:BCS over:DEY
490 .over STX buffer:STY buffer+1
500 LDY #0
510 .loop LDA (zp+2),Y:STA (buffer),Y
520 OPT FNdec(zp+2):OPT FNdec(buffer)
530 LDA zp+3:CMP zp+1:BNE loop
540 LDA zp+2:CMP zp :BNE loop
550 LDA (zp+2),Y:STA (buffer),Y
560 LDA buffer+1:STA zp+1
570 LDA buffer :STA zp:RTS
580 \ *****
590 .entry CMP #0:BEQ forme
600 JMP (oldv) \ not OSWORD 0
610 .forme STX block:STY block+1
620 TAY:LDA (block),Y:STA buffer
630 INY:LDA (block),Y:STA buffer+1
640 LDA flag:BNE secondtime:LDY #0
650 .firsttime
660 LDA auto,Y:STA (buffer),Y
670 INY:CPY #4:BNE firsttime
680 INC flag:CLC:RTS

```

```

690 .auto EQU8 "AU":EQU8 13
700 :
710 .secondtime LDY #&FF
720 .loop INY:CPY length:BEQ finished
730 LDA (zp),Y:STA (buffer),Y
740 BEQ finished
750 CMP #&0D:BNE loop
760 INY:TYA:CLC \ eol
770 ADC zp:STA zp:BCC rts
780 INC zp+1:CLC
790 .rts RTS
800 :
810 .finished SEI
820 LDA oldv :STA wordv
830 LDA oldv+1:STA wordv+1
840 LDA oldv+2:STA wrchv
850 LDA oldv+3:STA wrchv+1:CLI
860 \ re-enable ESC
870 LDA #200:LDX #0:JSR osbyte
880 SEC:RTS
890 :
900 .oldv EQU8 0
910 .length EQU8 251
920 .flag EQU8 0
930 ]NEXT
940 a$="SAVE XBAS "+STR$~install+" "+S
TR$~P%
950 PRINT a$:OSCLI a$
960 END
970 :
1000 DEFFNdec(address)
1010 LOCAL jump
1020 IFaddress<&100 jump=4 ELSE jump=5
1030 [OPT pass
1040 DEC address:LDX address
1050 INX:BNE P%+jump:DEC address+1
1060 ]=pass

```

**Listing 3**

```

10 REM Program .>VEDbas (3)
20 REM Version B1.00
30 REM Author Andrew Rowland
40 REM BEEBUG July 1991
50 REM Program subject to copyright
60 :
100 wrchv=&20E:filev=&212

```

```

110 oswrch=&FFEE:osbyte=&FFF4
120 oscli=&FFF7:top=&12:page=&18
130 himem=&6:listo=&1F:width=&23
140 pointer=&90:start=&92:block=&94
150 :
160 FOR pass=0 TO 3 STEP 3
170 P%=&900:[OPT pass
180 .install LDA #22:JSR oswrch
190 LDA #135:JSR oswrch
200 \ alter WRCHV
210 LDA wrchv :STA oldv
220 LDA wrchv+1:STA oldv+1
230 SEI:LDA #entry MOD &100:STA wrchv
240 LDA #entry DIV &100:STA wrchv+1
250 CLI
260 \ move program up
270 LDA #132:JSR osbyte \ read HIMEM
280 STY himem+1:STX himem
290 DEY:LDX #0
300 .over STY pointer+1:STX pointer
310 LDY top:STX top:BEQ zero
320 .loop LDA (top),Y:STA (pointer),Y
330 DEY:BNE loop
340 .zero LDA (top),Y:STA (pointer),Y
350 DEC top+1:DEC pointer+1
360 LDX top+1:CPX page:BCS loop
370 LDX pointer+1:INX:STX page
380 LDA #1:STA pointer:STA start
390 LDA #131:JSR osbyte \ read OSHWM
400 INY:STY pointer+1:STY start+1
405 LDY #0:LDA #13:STA (start),Y
410 \ WHENO 0, WIDTH 0
420 STY listo:DEY:STY width
430 \ put "L." in k/b buffer
440 LDA #21:LDX #0:JSR osbyte:LDY #0
450 .loop LDA auto,Y:JSR poke
460 INY:CPY #3:BNE loop
470 RTS
480 .auto EQU "L."+CHR$13
490 \ perform *FX138,0,n
500 .poke TAX:TYA:PHA:TXA:TAY
510 LDA #138:LDX #0:JSR osbyte
520 PLA:TAY:RTS
530 \ *****
540 .entry STY ytemp
550 LDY flag:BNE secondtime

```

```

560 CMP #13:BNE eout:INC flag
570 .eout LDY ytemp:RTS
580 .secondtime CMP #10:BEQ out
590 LDY count:BEQ ok
600 CMP #ASC">":BEQ finished
610 DEC count:JMP out
620 .ok CMP #13:BNE notcr
630 LDY #5:STY count
640 .notcr LDY #0:STA (pointer),Y
650 INC pointer:BNE out:INC pointer+1
660 LDY pointer+1:CPY himem+1:BNE out
670 BRK:BRK:EQU "No room. Press BREAK
":BRK
680 .out LDY ytemp:RTS
690 :
700 .finished SEI
710 LDA oldv :STA wrchv
720 LDA oldv+1 :STA wrchv+1
730 LDA filev :STA oldv
740 LDA filev+1:STA oldv+1
750 LDA #fentry MOD &100:STA filev
760 LDA #fentry DIV &100:STA filev+1
770 CLI
780 \ preserve lowest 2 bytes
790 LDY #0:LDA (start),Y:STA ytemp
800 INY:LDA (start),Y:STA ytemp+1
810 \ push "L ASC" into k/b buffer
820 LDA #21:LDX #0:JSR osbyte
830 LDY #0
840 .loop LDA view,Y:JSR poke
850 INY:CPY #6:BNE loop
860 LDX #string MOD &100
870 LDY #string DIV &100
880 JMP oscli
890 .view EQU "L ASC"+CHR$13
900 .string EQU "WORD":EQUB 13
910 :
920 .fentry CMP #5:BEQ cat
930 CMP #&FF:BEQ load
940 JMP (oldv)
950 .load LDA oldv:STA filev
960 LDA oldv+1:STA filev+1
970 .cat LDA block:PHA
980 LDA block+1:PHA
990 TXA:PHA:TYA:PHA
1000 STX block:STY block+1

```

```
1010 LDA #0:LDY #2
1020 .loop STA (block),Y:INY
1030 CPY #&12:BNE loop
1040 LDY #&A:LDA pointer:SEC
1050 SBC start:STA (block),Y
1060 LDA pointer+1:SBC start+1
1070 INY:STA (block),Y
1080 \ restore 2 lowest bytes
1090 LDY #0:LDA ytemp:STA (start),Y
1100 INY:LDA ytemp+1:STA (start),Y
1110 PLA:TAY:PLA:TAX
1120 PLA:STA block+1:PLA:STA block
1130 LDA #1:RTS
1140 :
1150 .oldv EQUW 0
1160 .flag EQUB 0
1170 .ytemp EQUB 0
1180 .count EQUB 5
1190 JNEXT
1200 a$="SAVE VED "+STR$~install+" "+ST
R$~P%
1210 PRINT a$:OSCLI a$
```

## Listing 5

```
10 REM Program .>VBASbas (M) (5)
20 REM Version B1.00
30 REM Machine Master/Compact
40 REM Author Andrew Rowland
50 REM BEEBUG July 1991
60 REM Program subject to copyright
70 :
100 osbyte=&FFF4:oscli=&FFF7
110 FOR pass=0 TO 3 STEP 3
120 P%=&A00:[OPT pass
130 LDA #180:LDX #0:LDY #&FF
140 JSR osbyte \ read OSHWM
150 INX:STX 1:STZ 0
160 LDX #string MOD &100
170 LDY #string DIV &100
180 JMP oscli
190 :
200 .string EQU$ "BASIC@":EQUB 13
210 JNEXT
220 a$="SAVE VBAS A00 "+STR$~P%
230 PRINT a$:OSCLI a$
```

B

## 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 us all ads (personal and business) to MEMBERS' ADS, BEEBUG, 117 Hatfield Road, St. Albans, Herts AL1 4JS. The normal copy date for receipt of all ads will be the 5th of each month.*

Prism modem 2000 for sale (no software or instructions) - Offers? Elite 5.25", Revs 5.25" boxed, complete offers, WANTED: BEEBUG magazines Vol.1 No.1 to Vol. 2 No.9 and BEEBUG discs before May '86 or will swap for above. Tel. (0705) 678410 anytime.

Epson RX80 printer £60, BBC Master ROMs, Printmaster £8, ADT £10, Interword £25, Master £12, on disc; (Superart, Pagemaker, Mouse) £30, Elite £8, Hyperdriver £5, Astaad £5, Master Reference manuals 1&2 £15, Voltmac joystick and splitter box £8, Watford User Port Splitter £10, 3 Quad cartridges £7 each. Tel. (0475) 38502.

BBC B issue 7, with Torch Z80 CP/N double disc attachment, almost unused Torch perfect,

comes with full software perfect writer, perfect filer etc. BBC A issue 7, with upgrade kit of chips to bring up to B specification (unused). Electron, used, with joystick attachment and modem connector (serial-port type attachment) £ . Tel. 091-271 5966.

WANTED: I've been given an ATPL board that I want to fit to my old BBC B Micro, the trouble is someone cut all the wires taking it out! and I have no instructions. Could anyone owning an ATPL ROM/RAM board please photocopy and send me the fitting and operating instructions? Also does anyone own a ROM/RAM board made by Peartree Ltd, (it has a socket to take the O.S ROM, four empty sockets and four yellow wires with a blue plug on the end of each). Once again I have no paper

work, can anyone help with a photo copy of the fitting instructions. I will be happy to pay any cost involved if anyone can help on this. Tel. (0322) 664761 eves or 071-494 1365 office hours.

Archimedes 310 colour system, 1Mb RAM, 2x 3.5" disc drive, RISC OS, PC Emulator, Amstrad 5M2400 modem and Hearsay, will split £100 o.n.o. Tel. (0494) 764643 after 6pm.

WANTED: BEEBUG Vols.1-8 complete, plus Vol.9 Nos.1,2 and 3, reasonable prices paid. Tel. (0792) 201848.

A3000 colour system, 1Mb, RISC OS, E-Type, Fun School 2, little used, mint condition, must sell £490 or offer. Tel. 081-360 8076 or 071-956 5488 office hours.



Econet level 3 file server (hard disc, BBC B, 2nd processor) £350, several BBC's each with SWRAM and Econet interface £150 each, all open to offers, support given if required during school hours 8.30-5.00. Tel. (0225) 464313 extn. 154.

Olivetti daisy wheel electronic typewriter, Octet 121 with KSR interface for RS423 socket on BBC, user and technical handbooks and extra daisies, superb quality printer with Wordwise etc. 16" carriage, hardly used, works perfectly, original price was £1020 accept £200 o.n.o. Tel. (0872) 73651.

Master 512 add-on board complete with Gem and mouse £180, "Mastering DOS Plus" £5, "The DOS Plus reference guide" £5, "PC Mag-DOS power tools" with disc £12, ATPL sideways ROM expansion board for B £10. Tel. 081-668 8776.

Electron teletext adaptor by Morley Electronics complete with PSU, lead to Plus one, instruction booklet, in working order, now you can have teletext on Electron, for quick sale £70 collectors item. Tel. 081-903 0904 eves.

WANTED: ATPL board for BBC B +128 preferably with battery back-up. Tel. 081-555 9069.

BBC system, dual disc drive DS80TK, £80. Tel. (0273) 832548 eves.

WANTED: View printer driver generator, with documentation, 3.5" disc for Master Compact, also Viewstore, manual not essential. Tel. (0763) 208365.

M128, includes Spellmaster, Lisp and Basic Booster ROMs and ROM cartridges, manuals etc. £250, two 80T disc drives with PSU's £55 each, 80T 3.5" drive £55, Acorn Atom memory chips (2114) £5 for 16, TRS-80 pocket computer plus printer/tape interface unit £50. Tel. (0276) 32358.

Pace Linnett modem £70, Hearsay £40 as new. Tel. (0452) 612580 eves only.

M128 £250, Marconi tracker ball £35, Philips CM8533 colour monitor £75, dual 80T disc drives £95, spare drive £35, various games, books, utilities from £7.50. Tel. (0474) 363503.

Magic modem complete with Commssoft ROM and manual, all fittings ready to go on model B or Master £30, Cumana 5.25" disc drive 80T, single sided with PSU's, recently overhauled at Cumana workshop £30. Tel. (0844) 52547.

Modem - Pace "Nightingale", 300/300 and 1200/75 baud rates, offered with BEEBUG "Command" communications ROM and cable

for BBC B, everything in mint condition and boxed £45. Tel. (0294) 52250 eves.

BBC B version 1-20, ATPL ROM /RAM board, DFS toolkit and tape to disc drive, all instruction manuals, joysticks, Acorn data recorder, Amber 2400 (2 1/4") printer, Ingersoll B&W television, numerous discs, tapes and magazines, for serious offers Tel. (0344) 422961.

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

BBC Public Domain software library has over 20Mb of software available!  
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.

RAM boards for BBC, Aries B32 board £32, Aries B20 board £20. Tel. (0223) 891960 anytime.

Bargain: FX80/NLQ board, printer included, BBC lead and manuals £25, two large boxes misc. Software, Firmware and books, BBC Master - Overview, VP, Adventures etc. £25 the pair. Tel. (0734) 576885.

Software sale BEEBUG Master ROM £20, Label Master & 2000 labels £12, The Publisher ROM £25, Wordwise + £20, Star LC10 colour dump ROM £15, Star Printmaster ROM £10, Kinship Disc £25, Family History System £15, Speech disc £5, W.E. diagnostic kit £5, BEEBUG Sleuth ROM (both BBC B only) £15, Genie in a Box £20, Context Bank Manager (Master) £15, Morley Teletext adaptor + 2 ROMs £45, GIS Advanced Teletext receiver £60, Delta 14/B joystick £5, Acornsoft Exile £5, Acornsoft Elite £5, Master Reference Manuals parts 1&2 £10, all above with manuals, original discs and ROMs, owner upgrading. Tel. (0753) 645332.

Acornsoft Micro Prolog ROM for BBC computers, with manual £10 + £1 postage. Tel. 081-807 8157.

M128, Cumana 40/80T twin DD, Spellmaster ROM, Voltmace joystick, keyboard cover, disc cleaner, disbox, games, reference manuals and BEEBUG magazines £350 o.n.o. Tel. (04022) 29921 anytime.

Dabhand Guide to View, good condition £8, Dabhand Guide to Viewsheets/Store,

good condition £8, Amstrad DMP3000 printer, good order £25, Complete set of Micro User Vol.1/1 - Vol.5/8 inc. Offers? Tel. (0442) 64003.

WANTED: Co-pro adaptor suitable for BBC Master working or not working, offers to 2 Gibbons Fields, Mullion, Helston, Cornwall, TR12 7EA.

For sale: Atari 520 ST FM and over 10 titles of software very good condition £300 o.n.o. Tel. (04023) 44179.

AMX Pagemaker £20, Z80 2nd processor and manual 2 cpm books £60, EMR Midi Interface £40, Master ref. 1&2 and advanced user £15, Watford Digitiser £40, Morley teletext and PSU ATS ROM utilities disc £45, BEEBUG C with Stand Alone generator £35, Dabs 512 User Guide and Tech guide with discs £20, Dabs Shareware sets 1&2 £15, BBC B separart with mouse £20. Tel. 061-445 6316.

BBC B issue 7 with 1770 DFS, machine only, also Watford steel plinth and Aries B32 shadow/sideways RAM £130 for Beeb, £30 for Aries & £10 for plinth o.n.o. Tel. (0438) 354177 ask for John.

BEEBUG Printwise disc and manual, Dabs Hyperdriver ROM and manual, Merlin Database ROM disc and manual, Micro Aid Cashbook disc and manual, Viewsheet ROM and manual, Viewstore ROM and manual, Leisure Genius Waddington's Monopoly, disc version, L.G. Computer Scrabble disc version, C.C. Wordwise plus and manual, Care cartridge for the Master, Advanced BBC Toolkit ROM and manual, Quicksilver BBC music processor cassette and manual, BBC soft home finance cassette. EMR software for the Master; Midi Interface unit, Miditrack Music Editor, Miditrack Performer, Scorewriter Utility, all with manuals. Books; Dabhand Guide to Viewsheet and Viewstore 340 page book (mint), Epson FX/Kaga printer commands revealed 116 page book, Disc Programming Techniques for the BBC Micro. No reasonable offer refused. Tel. (0705) 371018.

Viewstore and Viewspell both in original packaging with manuals £15 each or £25 for both, plus postage, little used replaced by Overview. Tel. (0792) 865691.

WANTED: Utility disc for Master hard disc ADFS. Tel. (0792) 865691.

WANTED: any info or HELP with a 21Mb hard disc drive 'Ice Microcube' by ICE Ltd. 54 Hartland Road, London E15 4AH. Tel. 081-534 0873.

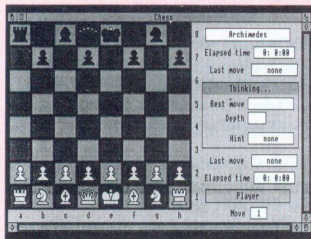
# RISC USER

## The Archimedes Magazine & Support Group

Now in its fourth year of publication, Risc User continues to enjoy the largest circulation of any magazine devoted solely to the Archimedes range of computers. It provides support for all Archimedes users at work (schools, colleges, universities, industry, government establishments) and home.

Existing Beebug members, interested in the new range of Acorn micros, may either transfer their membership to the new magazine 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 suppliers on the complete range of BBC micros. RISC User has a massive amount to offer to enthusiasts and professionals at all levels.



Society of Hog Casing Manufacturers - Annual Report

The society of hog casing manufacturers is pleased to announce that due to the increased consumption of the British sausage there was a marked increase in the production of casings in the period January to March 1986.

In part this was caused by a reduction of canned frankfurter sales and the increased public awareness of the high fat content in the indigenous product due to the recent television publicity given to the British product by the Minister of State the Hon. Jim Hacker MP.

Of particular note is the marked increase in the production of synthetic casings compared with the same period last year. The results are shown in the following table:

Production of hog casings (thousand metres)	1985		1986	
	Jan-Mar	Year	Jan-Mar	Year
Farm manufactured	67.2	255.0	65.0	255.0
Abattoir manufactured	92.0	290.3	84.0	290.3
Synthetic	98.5	1,127.1	193.5	1,127.1



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

### VIRUS ALERT

An extremely useful piece of software which will automatically warn you if your machine catches a VES.

### CHESS FOR THE ARCHIMEDES

A comparison between two contenders in the Archimedes chess stakes.

### DRAWFONT

A powerful application which provides the means to manipulate outline fonts in Draw.

### EASIWRIEER

A review of this comprehensive document processing and desktop publishing package.

### GOING INTERNATIONAL

A mini-series looking into wordprocessing in foreign languages.

### DEBUGGING THE WIMP

A look into the use of a number of readily available tools for debugging Wimp programs written in Basic or C.

### INTO THE ARC

A regular series for beginners currently treating the subject of using the Basic Editor.

### USING ANSI C

A new series of articles on programming Desktop applications in C.

### MASTERING THE WIMP

A major and very popular series on the Wimp programming environment.

### WP/DTP

A regular column which offers advice on using different DTP and WP packages.

### PROGRAMMER'S WORKSHOP

A major series covering all advanced aspects of the Archimedes and incorporating the Assembler Workshop.

### ARCADE

An occasional column offering a round-up of the latest games for the Archimedes.

### 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	£ 9.10
Rest of Europe and Elre	£14.00
Middle East	£17.00
Americas and Africa	£19.00
Elsewhere	£20.00



## MORE ON MASTER MOS DFS BUG

Mr. Robertson is not alone in finding the 'DFS' bug when using the new Master OS (see *Postbag* BEEBUG Vol.10 No.2), but I am not sure which one he is using. I have found the bug using MOSplus V1.14 from Dabs Press. Not only can I not write to DFS when switching from ADFS by typing '\*DISK', but the computer will not always switch back to ADFS from DFS by typing '\*ADFS'. I have found a solution to the latter problem by pressing Ctrl-F-Break (rather than Ctrl-A-Break).

Another problem is '\*BACKUP', when in DFS mode with MOSplus switched on (I have it installed in a Quad cartridge), which always returns the error "Bad backup". Switching off MOSplus avoids the problem, so I can only assume that the computer is trying to access the Backup utility in MOSplus, which is for ADFS only, and thus trying to read my DFS disc in ADFS mode.

While so far I have been negative in my letter, I must point out that MOSplus is extremely good when you want to format an ADFS disc, and perform other ADFS utilities.

Andrew Truscott

*Alan Wrigley, our Technical Editor and a long time user of MOSplus, has not encountered the same problems at all. He suggests that it is the absence of an ADFS format disc in the drive on entering \*ADFS which may be causing the situation described by Mr.Truscott.*

## ASTAAD FOR THE BEEB

Is ASTAAD worth getting for a BBC B, or is it designed so much for the Master that I am better off getting something else? Ideally I would like something like the old ASTAAD programs published in the early issues of BEEBUG, but I understand that the tapes on which these programs are recorded are no longer available.

David Regis

*ASTAAD (standing for Any Sized Text, printed at any Angle, Anywhere on the Drawing), is a CAD program for the BBC micro first published in BEEBUG Vol.2 No.7 with an extensive update in Vol.2 No.9. No copies of the original magazines (nor the accompanying cassettes) remain.*

*In Vol.7 No.5 (and the following three issues) we published a substantially enhanced version for the Master 128, and it is this version (with documentation) which is included in our Best of BEEBUG product list.*

*However, the original model B version (complete) was also included on the magazine disc for Vol.7 No.5, and the corresponding magazine contained a keystrip for that version. These back issues are still available (£1.20 for volume 7 magazines, £4.00 for a volume 7 magazine disc plus postage in each case) - see inside back cover for full order details. ASTAAD for the Master 128 costs £5.95 to members (plus £1.00 p&P).*

## TAB AND TAB

With reference to your series of First Course articles on VDU calls (see BEEBUG Vol.9 No.10) on the subject of PRINT TAB versus VDU31, did you know of one snag using the form TAB(x), in that it prints spaces in all the locations before it, unlike TAB(x,y) which jumps to the specified position without disturbing anything already on the line?

Bernard Beeston

*I must admit to overlooking this difference in the two types of TAB function when writing the article. To be specific, TAB(x) will print spaces from the current print position up to the Tab position specified, and is somewhat similar in effect to the SPC(x) function, though that prints a specified number of spaces, rather than printing up to a particular position. Personally, I have little or no use for this TAB format, but find TAB(x,y) very useful when working with non-scrolling screen layouts.*

M.W. 

# Magscan

Comprehensive  
Magazine Database  
for the BBC Micro and  
the Master 128

An updated version of Magscan, which contains the complete indexes to all BEEBUG magazines from Volume 1 Issue 1 to the latest Volume 9 Issue 10

Magscan allows you to locate instantly all references to any chosen subject mentioned anywhere in the 90 issues of BEEBUG magazine.

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.

```
BEEBUG MAGSCAN          VOLUMES 1-9

Enter Volume No. (1-9 or *) >* <

Enter article type      >*<
* - All types
A - General Article
B - Programming Article
C - Review
D - News
E - Hint
F - Points Arising
G - Application Program
H - Utility Program
I - Games Program
J - Miscellaneous

Enter String 1          >Basic   <
Enter String 2          >Program <
Logic OR/AND (O/A)     >AND<

Hard Copy (Y/N)       >N<
```

Specifying a Magscan search

```
BEEBUG MAGSCAN          VOLUMES 1-9

Volume : 1 2 3 4 5 6 7 8 9
Type   : All
String 1 : BASIC
String 2 : PROGRAM
Logic   : AND

Edikit (Part 5)
Basic Program Utility/Toolkit ROM
Programming Utilities
Vol 9 No 1 Page 30

Thanks for the Memory - Bas128 (Part 1)
Main Memory Resident Version of Basic
Sideways RAM Program Storage
Vol 9 No 3 Page 20

Hint: Improved Move-Down Routine
Using Additional Program Lines
Basic/PAGE/Memory Restrictions
Vol 9 No 4 Page 61
```

Entries retrieved from Magscan files

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

Phone your order now on (0727) 40303

or send your cheque/postal order to the address below. Please quote your name and membership number. When ordering by Access, Visa or Connect, please quote your card number and the expiry date.

**Magscan complete** pack, contains disc, manual and release notes: **£9.95**+p&p

Stock codes: 0005b 5.25"disc 40 track DFS 1457b 3.5" ADFS disc  
0006b 5.25"disc 80 track DFS

**Magscan update**, contains disc and release notes: **£4.75** +p&p

(for update, please return old disc, label or evidence of purchase)

Stock codes: 0011a 5.25"disc 40 track DFS 1458a 3.5" ADFS disc  
0010a 5.25"disc 80 track DFS

Postage: a - 60p UK, £1 Europe + Eire, £2.40 Elsewhere b - £1.50 UK, £2.50 Europe + Eire, £4.80 Elsewhere

# HINTS HINTS HINTS HINTS HINTS

*and tips and tips and tips and tips and tips*

This month's collection of hints contains an excellent idea for DFS users - clearly information not readily available otherwise. More hints and tips for this page are always welcome, and all hints published are paid for.

## COLOURED DISC TITLES

*K.Hollingsworth*

The following routine can be used to add a mode 7 coloured title to each of your discs to give them a professional look.

```
100 DIM buffer 11,sector 256
110 ?buffer=0
120 buffer?1=sector MOD 256
130 buffer?2=sector DIV 256
140 buffer?3=0: buffer?4=0
150 buffer?5=3: buffer?6=&53
160 buffer?7=0: buffer?8=0
170 buffer?9=&21: buffer!10=0
180 A%=&7F
190 X%=buffer MOD256:Y%=buffer DIV256
200 CALL &FFF1
210 ?sector=131: sector?1=65
220 sector?2=66: sector?3=67
300 buffer?6=&4B
310 CALL &FFF1:END
```

The text for your title (as ASCII codes) and any teletext colour codes, should be specified as shown in lines 210 and 220 (in the example, 'ABC' in yellow). The message after the colour code should not exceed 7 characters.

To do the same for the other side of the disc change line 110 to read:

```
110 ?buffer=2
```

In addition, if you change line 210 to:

```
210 ?sector=21
```

the catalogue will not appear at all.

## SAVING SIDEWAYS RAM

*Andrew Truscott*

If you have Computer Concepts' Spellmaster, a good way of saving the contents of any sideways RAM bank is to use the \*DSAVE command provided by Spellmaster to save dictionaries to disc, rather than the more cumbersome \*SRSAVE command (provided on the Master). With this command anything can be saved from sideways RAM (and ROMs as well).

## USING WATFORD SIDEWAYS RAM BOARDS

*K.Hollingsworth*

There is some confusion surrounding the use of Watford Electronic's sideways RAM boards. The 128K version has RAM in banks 0-7, and static RAM in bank E (14).

The way to emulate the \*SRLOAD command (of the Master 128) is to type in:


```
?&FF3x=0
```

where 'x' is the socket number ('E' must be used for socket 14), followed by:

```
*LOAD <image> 8000
```

where <image> is replaced by the name of the file which you wish to load. All that is now required is to press Break, and the ROM is active.

A further snippet of information gleaned from the rather meagre manual is that the Silicon Filing System image must be loaded into bank E, otherwise it will not work. Also when running the test program, the message "Suspect R5" may appear. The usual cause is that the Read/Write Protect switches are on; switching them off removes the problem.

Better documentation would also help enhance this otherwise excellent add-on. 

# BEEBUG MEMBERSHIP

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 subscribe to RISC User at a special reduced rate.

## BEEBUG SUBSCRIPTION RATES

£18.40	1 year (10 issues) UK, BFPO, Ch.1
£27.50	Rest of Europe & Eire
£33.50	Middle East
£36.50	Americas & Africa
£39.50	Elsewhere

## BEEBUG & RISC USER

£27.50
£41.50
£50.50
£55.50
£59.50

## BACK ISSUE PRICES (per Issue) 1 July 1991

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
Binders	£4.20		

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. Note that there is no VAT in magazines.

## POST AND PACKING

Please add the cost of p&p when ordering individual items. See table opposite.

Destination	First Item	Second Item
UK, BFPO + Ch.1	£ 1.00	£ 0.50
Europe + Eire	£ 1.60	£ 0.80
Elsewhere	£ 2.40	£ 1.20

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

BEEBUG MAGAZINE is produced by BEEBUG Ltd.

Editor: Mike Williams  
 Assistant Editor: Kristina Lucas  
 Technical Editor: Alan Wrigley  
 Technical Assistant: Glynn Clements  
 Production Assistant: Sheila Stoneman  
 Advertising: Sarah Shrive  
 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, BEEBUG Limited.

## 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.

**BEEBUG Ltd (c) 1991**

Printed by Arlon Printers (0923) 268328 ISSN - 0263 - 7561

# Magazine Disc

July 1991  
DISC CONTENTS

**CHINESE TANGRAMS** - a fun program implementing a classical puzzle designing a variety of patterns - tangrams - using a set of seven shapes: five triangles of different sizes, one square and one rhombus.

**BEEBUG WORKSHOP: THREE DIMENSIONAL GRAPHS** you can now visualise mathematical functions of two variables with this program for plotting three dimensional surfaces as perspective views, or as colour plots.

**PROGRAM DEVELOPMENT MADE EASY** - six utilities which allow BBC users to edit Basic programs in View, and Master owners to use Edit more efficiently, with separate model B and Master versions where appropriate.

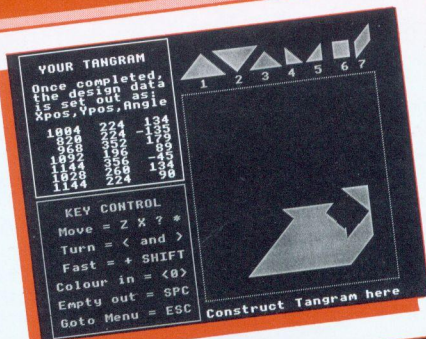
**RECREATIONAL MATHEMATICS: THE MATHEMATICS OF ENCRYPTION** - the last of the four programs related to number theory allows a message to be enciphered using the RSA method of encryption (public key encryption).

**DFS DISC RECOVERY UTILITY** you can recover accidentally deleted or even corrupted disc files of any length up to the maximum allowed by the disc or the DFS.

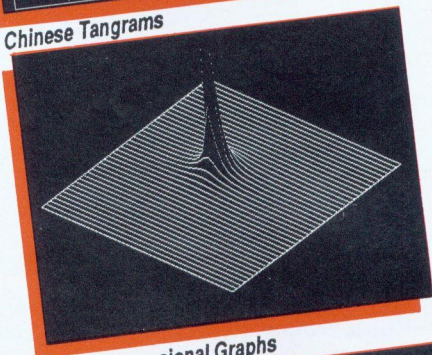
**WORDWISE PLUS NOTEBOOK** - a short segment program which strips the recipient's address from a standard letter and reformats this for printing onto a label or an envelope.

**PRINTING SCIENTIFIC CHARACTERS WITH WP (3)** - a repeat of last month's program which allows you to design scientific characters for use with Edit (the Master's built-in text processor) modified to cope with different printers.

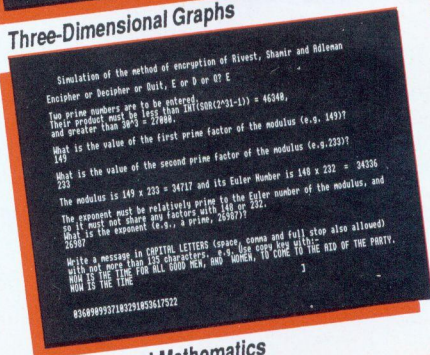
**MAGSCAN DATA** - bibliography for this issue (Vol.10)



Chinese Tangrams



Three-Dimensional Graphs



Recreational Mathematics

**ALL THIS FOR £4.75 (5.25" & 3.5" DISC) + 60p P&P (30p FOR EACH ADDITIONAL ITEM)**  
Back issues (5.25" disc since Vol.3 No.1, 3.5" disc since Vol.5 No.1) available at the same prices.

**DISC (5.25" or 3.5") SUBSCRIPTION RATES**

6 months (5 issues)
12 months (10 issues)

**UK ONLY**  
£25.50  
£50.00

**OVERSEAS**  
£30.00  
£56.00

Prices are inclusive of VAT and postage as applicable. Sterling only please.

**BEEBUG, 117 Hatfield Road, St.Albans, Herts AL1 4JS**

# Special Offers to BEEBUG Members July 1991

## BEEBUG'S OWN SOFTWARE

Code	Product	Members Price	inc Vat	Code	Product	Members Price	inc Vat
1407a	ASTAAD3 - 5" Disc (DFS)	5.95		PAG1a	Arcade Games (5.25" 40/80T)	5.95	
1408a	ASTAAD3 - 3.5" Disc (ADFS)	5.95		PAG2a	Arcade Games (3.5" )	5.95	
1404a	Beebug Applics I - 5" Disc	4.00		PBG1a	Board Games (5.25" 40/80T)	5.95	
1409a	Beebug Applics I - 3.5" Disc	4.00		PBG2a	Board Games (3.5")	5.95	
1411a	Beebug Applics II - 5" Disc	4.00		1600a	Beebug magazine disc	4.75	
1412a	Beebug Applics II - 3.5" Disc	4.00		0077b	C - Stand Alone Generator	14.56	
1405a	Beebug Utilities - 5" Disc	4.00		0081b	Masterfile ADFS M128 80 T	16.86	
1413a	Beebug Utilities - 3.5" Disc	4.00		0024b	Masterfile DFS 40 T	16.86	
1450a	EdiKit 40/80 Track	5.75		0025b	Masterfile DFS 80 T	16.86	
1451a	EdiKit EPROM	7.75		0074b	Beebug C 40 Track	45.21	
1452a	EdiKit 3.5"	5.75		0075b	Beebug C 80 Track	45.21	
0005b	Magscan Vol.1 - 8 40 Track	12.50		0084b	Command	29.88	
0006b	Magscan Vol.1 - 8 80 Track	12.50		0073b	Command(Hayes compatible)	29.88	
1457b	Magscan Vol.1 - 8 3.5" ADFS	12.50		0053b	Dumpmaster II	23.76	
0011a	Magscan Update 40 track	4.75		0004b	Exmon II	24.52	
0010a	Magscan Update 80 track	4.75		0087b	Master ROM	29.88	
1458a	Magscan Update 3.5" ADFS	4.75		1421b	Beebug Binder	4.20	

## OTHER MEMBERS' OFFERS

These offers are available for a limited period only. To avoid disappointment please order early, as offers are only available while stocks last and demand is always high. Orders are dispatched on a first come first served basis. To order phone (0727) 40303 or write to BEEBUG, 117 Hatfield Rd, St. Albans, AL1 4JS.

### Nevryon

£9.40 (inc VAT) +p&p

Normal price £14.51 (inc VAT)

A fast shoot'em up arcade game offering superb graphics and 8 levels of furious opponents. You fly a spaceship which contains immense firepower released from the top and bottom of your ship, and you must save the people from Nevryon by destroying all alien invaders.

Stock code 1133b

### Play It Again Sam 14

£7.90 (inc VAT) + p&p

Normal price £11.35 (inc VAT)

A four game compilation offering: **Predator**, where you can fight your way through the American Jungle, **Ballistix** - an original new ball game, **Super Soccer** - an action packed arcade game, and **Star Port** - a brand new arcade adventure with 60 action packed screens.

Stock code 5540b

## Special Offers to BEEBUG members from our Summer Clearance Sale

Check the special leaflet mailed out with this month's magazine for bargains on software and hardware.

Here are some examples of the offers (ex Vat):

<b>Easiword</b>	£ 25.00
<b>System Delta</b>	£ 41.00
<b>System Gamma</b>	£ 31.00
<b>View 3.0 Manual</b>	£ 2.50
<b>AcornSoft C</b>	£ 42.48
<b>Reconditioned Master 128</b>	£ 195.00
<b>Recond. Archimedes 310</b>	£ 400.00
(Recond machines have 3 months warranty)	
<b>and lots more.</b>	

Offer ends 31st July 1991.

Post & Packing	UK	Europe	Americas, Africa Middle East	Elsewhere
a	£ 1.00	£ 1.60	£ 2.40	£ 2.60
b	£ 2.00	£ 3.00	£ 5.00	£ 5.50
c	£ 3.10	£ 6.50	£10.50	£11.50
d	£ 3.60	£15.50	£24.50	£25.50