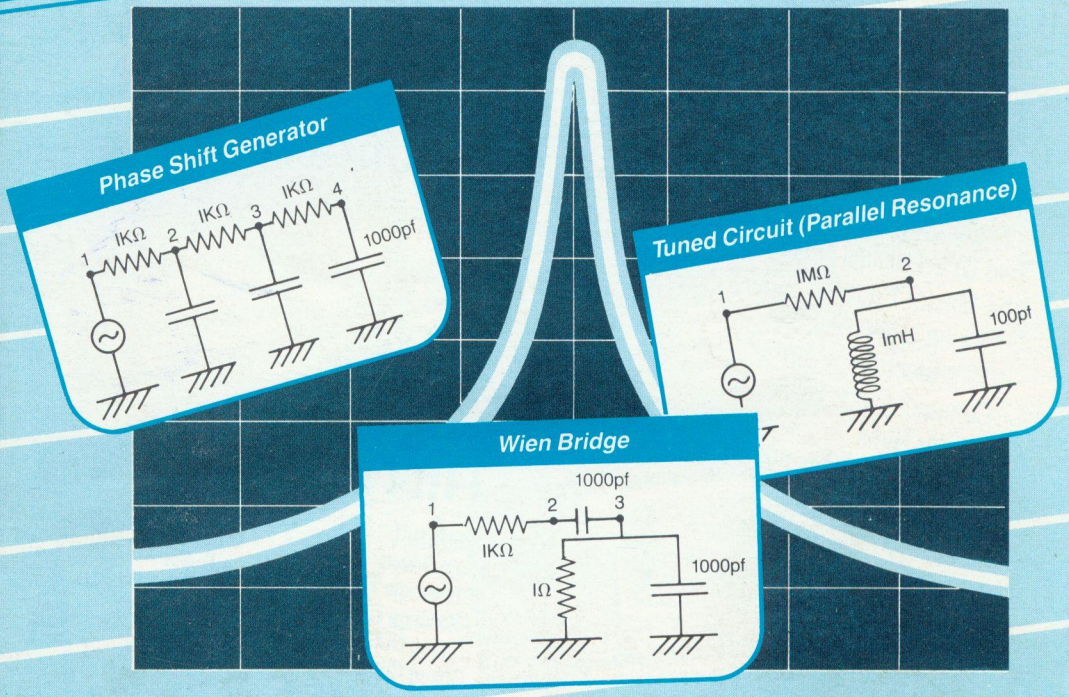
Vol.6 · No.8 · January/February 1988

BEEBUG

FOR THE BBC MICRO & MASTER SERIES



Circuit Analysis

- WORDPOWER REVIEW
- MUSIC COMPOSITION
- ARCHIMEDES A440
- M/C CROSS REFERENCER

BEEBUG Vol.6 · No.8 · January/February 1988

FEATURES

Hopalong 8
Circuit Analysis 14
Printing and Transposing Music 19
Barry Christie Visuals - The BEEBUG Sprite Editor 23
Workshop - Thanks for the Memory (Part 2) 28
Two DFS Utilities: *FREE and *MAP 30
The Master Pages - Talking to the ADFS 41
Care Electronics' Smart Cartridge 44
Compact Clock Reviewed 45
Master Hints 46
Macine Code Cross Referencer 50
First Course - Getting Animated 54
Exploring Assembler (Part 7) 58
Terras 5 62
The Comms Spot 66

REVIEWS

Archimedes A440 Reviewed 6
Word Power 11
Adventure Games 27
MEWSoft's Personal Organiser 47
Genie Junior & SideWriter 49
Two 'C' Compilers Reviewed 56

REGULAR ITEMS

Editor's Jottings 4
News 4
Supplement 33-40
Master Hints 45
Points Arising 67
Hints and Tips 68
Postbag 69
Subscriptions & Back Issues 70

HINTS & TIPS

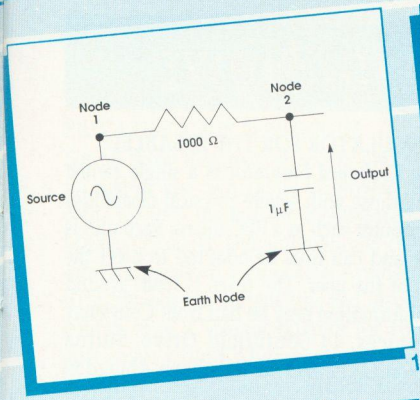
GENERAL
Fast Drives
Challenger Fix
Allocating Disc Space
Timing Out
Colour Composite Video
Minature Disc Labels
Two into One

MASTER
Conditional Printer Buffer
Appending Basic Revisited

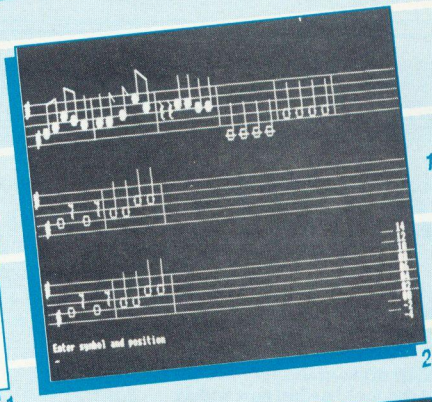
PROGRAM INFORMATION

All programs listed in BEEBUG magazine are produced direct from working programs. They are listed in LISTO1 format with a line length of 40. However, you do not need to enter the space after the line number when typing in programs, as this is only included to aid readability. The line length of 40 will help in checking programs listed on a 40 column screen.

Programs are checked against all standard Acorn systems (model B, B+, Master, Compact and Electron; Basic I and Basic II; ADFS, DFS and Cassette filing systems; and the Tube). We hope that the classification symbols for programs, and also reviews, will clarify matters with regard to compatibility. The complete set of icons is given



1.



2.

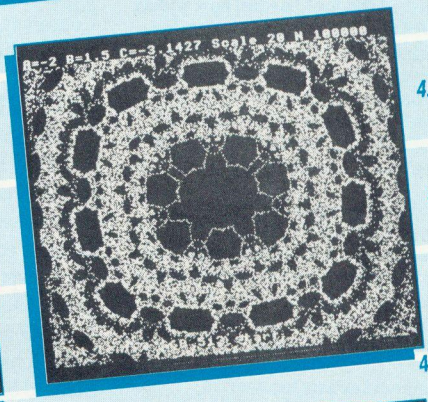
1. Circuit Analysis by Micro

2. Printing and Transposing Music

3. Sprite Editor



3.



4.

4. Hopalong

5. Word Power

6. Archimedes A440 Reviewed

¡ELIMINA MEJOR LOS ESCAPES! Protectores azules anti-hum
más ajustable. Asegúrese de que la cinta azul está bien a

ANNU BATTRE SKYDD MOT LACKAGE! Nu med bättre lackag
och förbättrad passform. Återförslut sedan blojan och kon
fast på blojan.

PIÙ PROTEZIONE DALLE FUORIUSCITE DI PIPI! Per una migli
di pipì, tirare il pannolino verso l'alto e distendere i bord

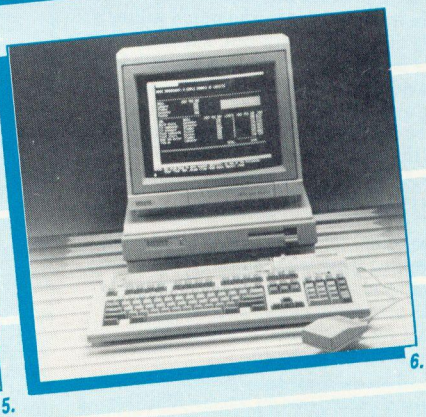
MEHR AUSLAUFSICHERHEIT! Blaue Auslaufsperre an der Ta
Sie die Klebebänder und -stellen vollkommen frei von Pud

MEILLEURE PROTECTION ANTI-FUITES! Pour contrôler, ouvr

Το WORDPOWER περιγράφεται λεπτομερώς στο τετραεδιδο
έχετε δει. Με λίγα λόγια το WORDPOWER είναι ένα δυναμικό
ονομάζουν το καλύτερο που υπάρχει για BBC ή Electron.

Ленинградской киностудии нужно было снять горные в
Шаня. План путешествия был такой: в мае экспедиция в
горах Тянь-Шаня и потом вернется в Ленинград.

5.



6.

below. These show clearly the valid combinations of machine (version of Basic) and filing system for each item, and Tube compatibility. A single line through a symbol indicates partial working (normally just a few changes will be needed); a cross shows total incompatibility. Reviews do not distinguish between Basic I and II.

Computer System

Master (Basic IV)

Compact (Basic VI)

Model B (Basic II)

Model B (Basic I)

Electron

Filing System

ADFS

DFS

Cassette

Tube Compatibility

Tube

Editor's Jottings

1987 proved to be a more than usually interesting year for Acorn and companies like ourselves who strive to provide a good service to the tens of thousands of users with Acorn micros. The major event of the year was undoubtedly the launch of the Archimedes, now widely recognised as the world's fastest micro computer. We in turn marked the significance of this event for the future of the BBC micro market by starting a separate magazine and user group devoted exclusively to the Archimedes, RISC User. With three issues completed and a fourth well on the way we can only say that we have been more than pleased with the response.

However, 1987 was not good news for everyone, particularly the 47 or so staff made redundant by Acorn not long before Christmas, an event which followed shortly after the unexpected departure of Acorn's then Managing Director Brian Long. And Acorn made other decisions that indicated that there are still lessons for the company to learn.

Despite that, we are all enthusiasts for Acorn's machines, and there are few users who would willingly part with their faithful and trusted BBC micro for the offerings of other manufacturers. The Archimedes is important in providing an upward path for those who wish to move on, and yet still retain all that is good about BBC micros. This is a healthy situation, and we very much hope that as more software becomes available for the Archimedes, we shall see Acorn's new flagship go from strength to strength.

This is just as important for all the faithful model B owners, not forgetting those with Masters and Compacts. A healthy market place for Acorn supports and maintains an active and flourishing user population so that we can all look forward with confidence to the future with Acorn.

With the expansion of BEEBUG and the launch of RISC User, we are urgently in need of a Technical Editor to join our editorial team. The work is hard and demanding, but at the same time very varied and interesting. If you are interested you will need to convince us that you have both a reasonable technical knowledge of the BBC micro range, and that you are capable of writing reviews and articles for the magazine, and editing the work of others. The right person would be able to play a major role in producing both BEEBUG and RISC User. If you are interested, then please apply enclosing a full cv direct to the Editor, and convince us you are the person we need.

..News..News..

KEYBOARD EMULATOR FOR THE DISABLED

The *Courtenay Keyboard Emulator* is a single switch device allowing those with severe physical disabilities to control a computer such as a BBC micro. Each unit is made to order and can thus be adapted to meet the specific needs of the user. The price is from around £348. Further information may be had from Courtenay Technical Services, 14 Southfield Drive, Sutton Courtenay, Oxon OX14 4AY or telephone Abingdon (0235) 848587.

COMPUTER EDUCATION GROUP

In addition to the other organisations mentioned in BEEBUG Education (Vol.6 No.7), there is the well-established *Computer Education Group* which has been in existence for over 20 years. C.E.G. organises a biennial conference, and the next one will be held at the University of Nottingham in April 1988. For more information on C.E.G. and this years conference write to the Computer Education Group, North Staffordshire Polytechnic, Computer Centre, Blackheath Lane, Stafford ST18 0AD.

TOPOLOGICAL TWIST

Rapidly making a name for themselves in the Adventure Games market, Toplogika has produced a second edition of best selling mathematical adventure game *Giantkiller* costing £18.40 on 5.25" disc and £21.70 on 3.5" disc for the Compact. Giantkiller is written by that well known Adventure Game writer, Peter Kilworth (see this month's Adventure Game column).

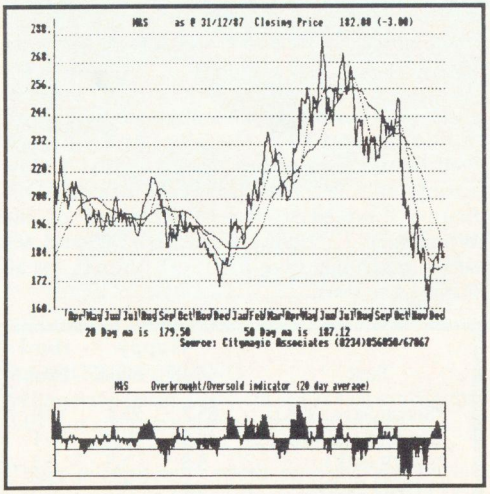
Topologika's latest release, on two discs, is *Polyominoes*, a suite of five programs for investigating the properties of Polyominoes. The price is the same as for Giantkiller, and Topologika claims that it is suitable for seven year olds and upwards.

Moving even more into the educational market, Toplogika is expecting to release a package called *Curves* in March/April. Curves is a multi-media cross-curricular pack for primary/middle schools, and will cost around £29.00. For more information contact Toplogika on (0733) 244682.

News..News..News..News..News..

PRIVATISE YOUR MASTER 128

For all those interested in stocks and shares (whether for real or just as a game) Citymagic Associates have come up with the answer, a software package for the Master 128 which will automatically collect share price information from Ceefax (168 equities and FT30 and FTSE100 indexes) for analysis and display. You will, of course, need a Teletext adaptor for your Master. The software, called *Sharematic II*, costs £35 from Citymagic Associates, 40 Manor Road, Goldington, Bedford MK41 9LQ or phone Bedford (0234) 856050.



BBC SOFT HAS A GOOD TWITCH

Just the program for young ornithologists is *Bird Spy* from BBC Soft. This specialist database program allows sightings of birds to be recorded, displayed and analysed. The software can be used for other applications, and adapted for all types of survey work. It has been produced in association with the BBC Radio series 'Looking at Nature'.

Another recent release from BBC Soft, Geordie Racer, is a detective game based on the BBC Television series 'Look and Read'. Colourful graphics test lateral thinking and memory power without which the hard-won treasure is irretrievably lost. Both applications, which are intended for school and home use, are supplied on 40 track disc at £14.95. For more information phone BBC Enterprises on 01-576 0339.

LATEST GAMES

If you want to change the course of history then *World War I* from software house Lothlorien could be the answer. In this new wargame scenario, you have the chance to change the colour of the world map. Play either against the computer or against another opponent. The game comes on disc only for the BBC micro at £12.95.

Bug-Byte's latest games compilation, imaginatively entitled *Compilation II*, contains Cricket, Ice Hockey, Hunkydory and Jack Attac - all on disc for just £7.95. Both Lothlorien and Bug-Byte are part of the former Argus Press Software Group, now known as Grand Slam Entertainments after a management buy out. Phone 01-439 0666 for further information.

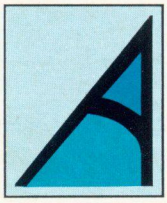
PEN DOWN FOR A FULL HOUSE

Logotron, noted for their Logo and supporting software has released two further discs of utilities for use with PenDown. Much of the new software has been contributed by PenDown users. Included on the two discs are new fonts, a font merger, adventures, and dictionaries for project work at a cost to existing PenDown users of £9.00 plus VAT.

Another development from Logotron is *3-D Logo* for the teaching of three dimensional geometry. The 3-D Logo pack costs £15 for the BBC and Master series, while at the same time Logotron Logo has been reduced from £60 to £44. Logotron may be contacted on Cambridge (0223) 323656.

PINEAPPLES UPGRADED

Pineapple Software has recently updated its popular *PCB* printed circuit drafting program (see BEBUG Vol.5 No.8). As well as improvements to the original PCB ROM, a second ROM is now available which gives full automatic track routing capabilities. This update is available to registered PCB users at £55. Pineapple Software is at 39 Brownlea Gardens, Seven Kings, Ilford, Essex IG3 9NL or telephone 01-599 1476. B



The ARCHIMEDES A440 reviewed

The Archimedes 440 is now in the shops in limited numbers at least. Lee Calcraft takes a look at what this top of the range machine has to offer.

The entry price of the Archimedes 440 is around £2600. This is almost three times the cost of an 310, so what extra features do you get for your money? Externally the difference between the the 440 and the 300 series machines is minimal. All make use of identical system boxes, and the only difference in the keyboards is that the 440 has grey coloured function keys instead of the distinctive red colour used on the 330 series. At the rear of the machine there is again very little difference. All plugs, sockets and panels are identical except that the phono video socket on the 300 series has been replaced on the 440 by a pair of BNC sockets used for the attachment of a special high frequency mono monitor (video and sync).

Internally of course the machine looks quite different. It has a completely new PCB designed to accommodate 4 Mbytes of RAM. The strict upper limit for the 300 series is 1 Mbyte. There are three other obvious differences in the internal hardware. The machine houses a 20 Mbyte hard disc (of which more later), a small cooling fan, and a four-slot backplane. A backplane is essentially a board extension socket. The 300 series machines may be upgraded to take a two-slot backplane at a cost of around £40, but the 400 series machines provide four as standard. Of the four, slot number 2 may be used to hold a co-processor, and Acorn plan to have a floating-point co-processor board on the market by the last quarter of 1988. The 300 series machines, by contrast, have no co-processor interface, and may not be upgraded to take one.

The manuals and firmware of the 440 are again identical to those supplied with 300 series machines. The 440 is currently supplied with Arthur 1.2, and with Basic V version 1.02. Its processor runs at the same speed as that of 300 series machines, and although the RAM chips in 400 series machines are somewhat faster, they are not clocked at a faster rate. The machines therefore perform identically, except where storage operations are involved.

HARD DISC

The 440's internally mounted 20 Mbyte Tandon hard disc provides a welcome increase in both storage speed and capacity. It is a delight to use, and runs more quietly than many hard disc machines. The accompanying speed tests give some idea of its performance on 3 different types of test: saving and loading 80K screens using the *ScreenSave and *ScreenLoad commands, saving and loading an 80K block of RAM using *SAVE and *LOAD, and finally the PCW "Store" benchmark, which involves writing a 20 byte string to a file 1000 times in succession. I have given comparison timings for both 600K and 800K floppies. And you may well be more surprised by the relatively slow speeds attained with the 600K format ADFS floppy than with the high speed of the Winchester. I should add that the 300 and 400 series machines give identical timings for all floppy disc tests.

Test	Floppy 600K	Floppy 800K	Hard Disc
*ScreenSave (80K)	85.7	25.3	10.0
*ScreenLoad (80K)	40.8	18.0	5.6
*SAVE (80K)	5.5	3.5	0.4
*LOAD (80K)	5.2	3.2	0.3
Store Benchmark	18.9	6.4	3.1

Fig 1. A440 Timings (times in seconds)

NEW SCREEN MODES

Archimedes 300 series machines boast 21 screen modes, of which the top three (i.e. modes 18, 19 and 20) require a special multi-sync monitor. The 440 has two extra screen modes, labelled 22 and 23 (number 21 is "reserved for future expansion"). The two new modes are mono only, and require a further "special" monitor (in this case, one with a 96MHz line scan). Of the two, mode 23 is a text only mode, giving 144 characters on 54 lines (though you need a special 8x16 font for this). Mode 22 more

usefully combines text (at 160 characters by 122 lines) with graphics, where it provides a resolution of 1280x976. This is very close to the 1280x1024 graphics units used on all versions of the BBC micro. The 48 missing vertical lines are taken equally from the top and bottom of the screen. As you will appreciate this is a very impressive resolution, and will be ideal for CAD and other graphics use, though lack of a special monitor prevented me from testing it out.



RAM CONFIGURING

When you enter Basic on a default-configured 440 you are presented with the message:

```
Starting with 3698940 bytes free
```

There is obviously room here for some quite long programs! Exactly how much RAM is allocated depends, as it does with the 300 series, on the *CONFIGURE options. On the 300 series these allocations are all made in page units of 8K (except for FontSpace, which uses 4K pages). In 400 series machines, the size of page used for the *CONFIGURE command is four times as large (except for FontSpace which remains at 4K). Thus if you execute:

```
*CONFIGURE ScreenSize 10
```

you will allocate 320K of RAM to screen use on a 440.

The default *CONFIGURE options on the 440 are adjusted accordingly, and are given in the accompanying table. Software designed to run on the full range of Archimedes machines will need to take account of the new configured page sizes. Unfortunately there is no very easy way of distinguishing between the various members of the Archimedes range. The normally useful INKEY(-256) call gives the same result for all members, and is only useful for distinguishing between an Archimedes and earlier BBC micros. To use this test, type:

```
PRINT INKEY(-256)
```

All Archimedes machines give the result 160. The test given in listing 1 will however distinguish between the two Archimedes machine series. It reads the MEMC status register, and prints out the machine series number by checking whether the memory page size is 8K (300 series) or 32K (400 series).

```
  10 REM Tests 300 or 400 series
  20 PRINT300-100*FNfour;" Series machi
ne"
  30 END
  40 :
  50 DEFFNfour
  60 SYS &1A ,0,0 TO reg
  70 =(reg AND 8)=8
```

Function	Config	Allocation
Font Size	6	24K
Screen Size	0	160K
RAM Filing	0	0
System Size	0	32K
RMA Size	2	64K*
Sprite Size	1	32K

*RMA Size 64K nominal. With all resident modules engaged, the actual space allocated to the RMA is 192K.

Fig 2. A440 default RAM allocations

CONCLUSION

The added RAM, hard disc, ultra high resolution modes and the potential offered by the co-processor interface make the 440 a highly desirable piece of kit. Its high price will of course mean that only educational and business establishments will in general be able to justify its purchase. Home users who feel the need for the 4 Mbytes of RAM and high resolution offered by the 400 series machines will need to wait for the emergence of the 410 in the spring. This has no hard disc, and has 1 Mbyte of RAM upgradable to 4 Mbytes, but costs just £1608.85 for the entry level system.

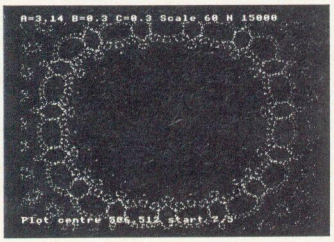
Hopalong

Explore the unexpected world of colourful and fascinating patterns revealed in Kate Crennell's short program.

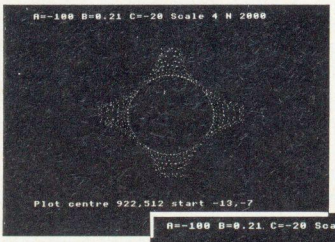
This program plots coloured points apparently 'hopping' around quite randomly. As more points are plotted, you will see that they are actually arranged in intriguing and fascinating patterns, which often resemble the form of cellular organisms as seen down a biologist's microscope (Fig. 1).

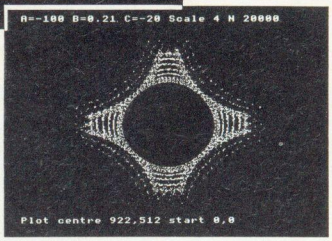


Simply changing a few parameters gives an infinite variety of patterns, fascinating to watch, because they appear to get stuck repeating themselves, and then suddenly start to grow another outer layer of lacy wings around their edges (Fig. 2).



Some are almost symmetrical (Fig. 3), others quickly become 'chaotic'. Some are initially rather dull (Fig. 4a- top right) but after 20,000 points a more interesting pattern suddenly develops (Fig. 4b - below).

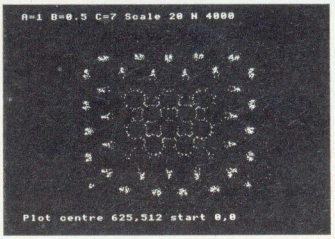




Plotting time on a Master is 12 seconds per 1000 points, and takes about twice that on a model B.

MAKING THE FIRST PLOT

Type the program in carefully and save away to tape or disc. When you first run the program just press Return to make the first plot (Fig. 5) using the default values. As it progresses, your parameters are shown at the top and bottom of the screen; Escape stops the plotting immediately, or you can wait until all the points you asked for have been plotted. Pressing 'C' will plot the same number of points again, starting where it left off.



To change the plot, press M to see the menu. The line with the parameter to change has the cursor next to it and is displayed in green; use the cursor keys to move up and down the menu, and press Return to select the action or change parameters. When you see the prompt **Enter new value,** type the number you want and press Return again.

EXPLANATION OF PLOT PARAMETERS

A, B and C are constants in the equation, used by the program to determine (iteratively) the co-ordinates of each successive point, and may

be any numbers you like. Try the values given in table 1 to get some ideas. See the section "Mathematics" for a more detailed explanation.

Starting X,Y are the values of X and Y in the equation when you start the plot; by default they are 0,0 but you may want to continue a plot from a particularly exciting place so they can be anything. To reset both of them to zero, move to **Reset start x,y** and press Return.

Plot origin and scale determine where the pattern starts on the screen, and its size. Hopalong tries to position the plot in the middle for you automatically by looking at A, B and C, but you may have to change the scale yourself.

No. of points is the number of points to plot this time.

Chg col after determines the number of plotted points after which the colour changes. The default is 1000, but some plots may look better with 50, others with 5000.

Quit stops the program.

Incidentally, the name HOPALONG was coined by A.K.Dewdney in 1986 (see reference below).

MATHEMATICS

The maths behind these plots is similar to that of the Mandelbrot Set (see BEEBUG Vol.5 No.1) except that these pictures come from iterating real numbers. The equations are:

```
Y - SGN(X)*SQR(ABS(B*X)-C)  ==> X
    and                     A-X ==> Y
```

The exact pattern depends not only on the starting values for X,Y and on the constants A, B and C, but also on which computer you are using. The Master and the BBC model B give similar but not always identical patterns, because they have different versions of Basic.

The equation is on line 210; the FOR loop is all on one line for speed. Other simple equations can be substituted for this one to make different sets of amusing patterns. You might try:

```
y - SIN(B*x-C)
```

with values of A close to odd multiples of PI, and any value for B and C. C=PI/2 is the same as changing from SIN to COS in the equation.

SAVING AND DUMPING PLOTS

Typing **S** when the plot is finished *SAVEs the screen into a file, named PLOTn, where n starts at 0 for each run.

Typing **D** will dump the screen to a printer if you insert a call to your dump routine at line 1000 (either your own routine or a call to a commercial dump such as Printmaster or Dumpmaster).

TABLE OF EXAMPLE PARAMETERS

A	B	C	Points	Scale	Fig.	Comments
-3.14	1.414	2.718	6000	40	1	biological cells
3.14	0.3	0.3	15000	60	2	a posy
-3.14	0.3	0.3	20000	100	3	
-100	0.21	-20	2000	4	4ab	also 20000 points
1	0.5	7	4000	20	5	default pattern
15	7.1	10	2000	16		a Valentine
1	51	17	2000	8		also 50000 points, scale 4
1.57	0.3	0.3	3000	100		

FURTHER READING

1985 British Computer Society, **Computer Bulletin** pages 18, 19 refer to the original idea due to Barry Martin, University of Aston.

1986 September **Scientific American** 'Computer Recreations' A.K.Dewdney pages 15 to 20 has coloured pictures.

```
   10 REM Program Hopalong
   20 REM Version B1.0
   30 REM Author  Kate Crennell
   40 REM BEEBUG  Jan/Feb 1988
   50 REM Program subject to copyright
   60 :
  100 ON ERROR GOTO 370
  110 PROCInit
  120 MODE 7:PROCMenu
  130 MODE1:REM to turn the main screen
BLUE add here VDU19,0,4,0,0,0
```

```
  140 PRINT"A=";A;" B=";B;" C=";C;" Scal
e ";S:PRINTTAB(0,30)"Plot centre ";INTXC
;",";INTYC;" start ";INTX;",";INTY;
  150 VDU29,XC;YC;28,29,0,39,0
  160 TS=T*S:REM Scale plot by S and rot
ate by -45 degrees to fit screen better
  170 GCOL0,C%
  180 REM Start iterations
  190 FOR I%=U% TO N%-1 STEP K%
  200 GCOL0,C%:COLOURC%:CLS:PRINTI%;
  210 FOR J%=V% TO K%:X1=Y-SGN(X)*SQR(AB
S(B*X-C)):Y=A-X:X=X1:PLOT69,TS*(X+Y),TS*
(Y-X):V%=1:NEXT
  220 U%=0:C%=C%MOD3+1
  230 NEXT:J%=0
  240 CLS:PRINTI%+J%;:VDU7
  250 REPEAT
  260 VDU28,0,0,39,0:CLS:COLOUR3:PRINTTA
B(0,0)"M-enu, C-ont. D-ump, Q-uit, or S-
ave";
  270 G%=GET:CLS:PRINT"A=";A;" B=";B;" C
=";C;" Scale ";S;" N ";I%+J%;
  280 IFG%>95 G%=G%-32
  290 A$=CHR$G%
  300 IF A$="D" PROCDump:GOTO 250
  310 IF A$="S" PROCSave:GOTO 250
  320 IF A$="Q" PROCq:END
  330 IF A$="M" GOTO 120
  340 IF A$="C" VDU26:GOTO 140
  350 UNTIL FALSE
  360 :
  370 IF ERR<>17 MODE7:REPORT:PRINT;" at
 line ";ERL:END
  380 IF E%=0 THEN U%=I%:V%=J%:GOTO 250
  390 GOTO 120
  400 :
 1000 DEFPROCDump:ENDPROC:REM call your
screen dump here
 1010 :
 1020 DEFPROCSave:OSCLI"SAVE PLOT"+STR$(
P%)+" 3000 8000"
 1030 P%=P%+1:ENDPROC
 1040 :
 1050 DEFPROCq:VDU26,12:*FX4,0
 1060 ENDPROC
 1070 :
 1080 DEFPROCInit
 1090 A=1:B=0.5:C=7:REM equation constan
ts
 1100 X=0:Y=0:REM initial values
 1110 XC=640:YC=512:REM origin of plot
 1120 S=20:REM scale of plot
 1130 K%=1000:N%=4000:REM change color a
fter K%, N%=no points to plot
 1140 T=1/SQR(2):E%=0:P%=0:REM P% is sav
ed plot number
 1150 XC=640-A*S*T:YC=512:REM origin of
plot
 1160 BL$=STRING$(8,CHR$32)
 1170 ENDPROC
 1180 :
 1190 DEFPROCMenu
 1200 MINA%=5:MAXA%=17:A%=MINA%:T%=8:E%=
1:*FX4,1
 1210 VDU23;10,98;0;0;0
 1220 CLS:IF K%>N% K%=N%
 1230 FORI%=1TO2:PRINTTAB(5,I%)CHR$141+C
HR$131+CHR$157+CHR$132+"PLOT PARAMETERS
 "+CHR$156:NEXT
 1240 PRINTTAB(T%,5)"Start plot"
 1250 @%=6
 1260 PRINTTAB(T%,6)"Value of A";SPC(4);
A
 1270 PRINTTAB(T%,7)"Value of B";SPC(4);
B
 1280 PRINTTAB(T%,8)"Value of C";SPC(4);
C
 1290 PRINTTAB(T%,9)"starting X";SPC(4);
X
 1300 PRINTTAB(T%,10)"starting Y";SPC(4)
;Y
 1310 PRINTTAB(T%,11)"Plot origin X "XC
 1320 PRINTTAB(T%,12)"Plot origin Y "YC
 1330 PRINTTAB(T%+3,13)"Scale";SPC(6);S
 1340 PRINTTAB(T%,14)"No. of points "N%
 1350 PRINTTAB(T%,15)"Chg col after "K%
 1360 PRINTTAB(T%,16)"Reset starting val
ues"
 1370 PRINTTAB(T%,17)"Quit "
 1380 @%=10
 1390 PRINTTAB(T%,22)"SELECT with cursor
 keys"''TAB(T%)"RETURN for action";
 1400 REPEAT:VDU31,T%-2,A%,130
 1410 I%=GET:IF I%<>13VDU31,T%-2,A%,135
 1420 IFI%=139 A%=A%-1
 1430 IFI%=138 A%=A%+1
 1440 IF A%<MINA% A%=MAXA%
 1450 IF A%>MAXA% A%=MINA%
 1460 UNTIL I%=13
 1470 IF A%=MAXA% PROCq:END
 1480  IF A%=16 PROCInit:GOTO 1220
 1490 IF A%>MINA% THEN GOTO1540
 1500 *FX4,0
 1510 E%=0:U%=0:V%=1:C%=3
 1520 ENDPROC
 1530 :
 1540 PRINT;TAB(0,A%)+CHR$130:PRINTTAB(T
%,22)"ENTER new value";BL$;:INPUT W
 1550 ON A%-MINA% GOTO 1560,1570,1580,15
90,1600,1610,1620,1630,1640,1650
 1560 A=W:XC=640-A*S*T:GOTO1220
 1570 B=W:GOTO1220
 1580 C=W:GOTO1220
 1590 X=W:GOTO1220
 1600 Y=W:GOTO1220
 1610 XC=W:GOTO1220
 1620 YC=W:GOTO1220
 1630 XC=640+(XC-640)*W/S:YC=512+(YC-512
)*W/S:S=W:GOTO1220
 1640 N%=W:GOTO1220
 1650 K%=W:GOTO1220
 1660 ENDPROC
```

WORD POWER

On the BBC micro, word processors like View, Wordwise and Interword have attracted most of the attention. But there is competition, and Howard Ward redresses the balance with a euphoric tale of Wordpower, a unique and powerful word processor from Ian Copestake Software.

Product	**Wordpower and Power Fonts**
Supplier	**Ian Copestake Software** **10 Frost Drive,** **Wirral, Merseyside L61 4XL.** **Tel. (051-648) 6287**
Price	**From £40.25 inc. VAT and p&p** **(Power Fonts extra)**

Special starter pack for Education (inc. Wordpower, Powerfont NTQ and European languages) - £75 (exc. VAT)

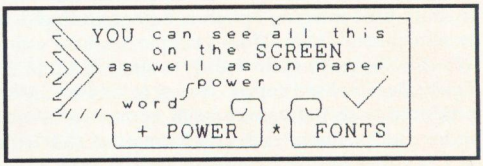
I have owned a copy of Wordpower ever since its release about two years ago when it superseded Wordsworth. Quite simply, I believe it to be the best BBC word processor on the market - and in my job I need to use all of them from time to time.

Wordpower runs on just about every kind of equipment, from an Electron to an Archimedes, and can use sideways and shadow RAM. I normally use it on a large E-Net network in Oxford. The program works fully in any screen mode, so it is just as happy on a television screen as on a high-resolution monitor. I dislike software which will not run with a second processor, so I was pleased to see that Wordpower takes full advantage of the extra memory which this offers and which is so invaluable for word processing. And there is no need to buy a separate 'hi' version.

The basic package consists of a disc, a fully-indexed manual, a function key strip and an optional ROM chip. If you have a second processor or sideways RAM you need not buy the ROM (this can make the price very competitive for educational users). You are allowed to make a security copy of the disc, which can be obtained in various DFS and ADFS sizes and formats.

Even if you buy a ROM you must still insert the Wordpower disc when you start up (it can be removed immediately afterwards so a single drive is sufficient). At first this might seem a bit of a nuisance, but there is a very valuable benefit - the 'User Variables File' or UVF. This contains settings for screen mode, colours, line length, printer type, and many other features so that the program always starts up the way you choose. The settings are easy to modify, and I keep a library of different UVFs to suit various purposes.

It is hard to imagine an easier program to use than Wordpower. After pressing Shift-Break, you can start typing immediately - very encouraging for beginners and, I am pleased to say, especially young children. The program assumes sensible default values for everything, and text is formatted automatically all the time. When you start, the screen is blank except for a small information panel at the top, and a prompt at the foot inviting you to press Ctrl-H for help.



The information panel tells you, for example, whether you are in insert (the default) or overwrite mode, and provides constantly updated figures for words typed and words free. The latter value is based on the average length of the words you have already used, and is far more meaningful to the novice than the usual 'bytes free'.

Ctrl-H produces a series of displays summarising the program's main functions: these can be used like menus or simply as on-screen reminders. Having introduced many newcomers to Wordpower I have seen time and again the remarkable amount people can do without even looking at the manual or function key strip.

Wordpower uses the function keys at all three levels. You can still create your own *KEY definitions, and f9 is left unused so you don't need to go too close to the Break key on a Model B. The cursor keys with or without Shift and Ctrl let you move around the text in a logical and flexible fashion. Ctrl combinations are generally easy to remember. Ctrl-R, for example, lets you modify a

ruler or create a new one (as many as you like). Lines of up to 250 characters are allowed; margins and tabs are set by 'dragging' a cursor to the required position.

The problem of displaying long lines is overcome by a novel device called 'slicing', whereby you can look at your text in slices of 40 or 80 columns. It's easy to check the alignment of tables or figures, but most of the time you can see all the text at once. This beats sideways scrolling and is a better solution for most users than trying to read special screen modes with 100-odd characters to the line. Wordpower also provides a sort of 'Ctrl-Lock' for people who can only press one key at a time. This 'one finger' option is invaluable to the disabled.

Wordpower has all the facilities we have come to expect in a wordprocessor, such as block copy, move, and erase, and a versatile search and replace, together with some useful extras like erase word. But in addition it offers some unusually powerful features as standard. The professional but very friendly mail-merging option is a good example. It is quite possible to build up a simple database using Wordpower itself, with records containing up to 26 fields. Personalised forms, letters or labels can be produced with individual fields appearing in any order and position, correctly formatted and with any required printer highlights. Special links allow Wordpower to mail-merge using data from ViewStore, System Delta, SupaStore and a number of other databases.

Another advantage Wordpower offers is the facility to split the screen into two windows of variable size. These windows can show two separate files, or different areas of the same file, and you can edit text in either window. You might use one window to display a block of text to be moved, while you locate the target position in the other window. For specialised tasks such as translation the windows are a blessing.

Security is one of Wordpower's strong points, and error-trapping is excellent. It is virtually impossible to delete anything by accident. Break and even Ctrl-Break seem to have no effect (primary school teachers please note); if you really wish to leave the program you have to indicate this by pressing Ctrl-L first. When you edit an existing file the program can automatically preserve the previous version for you - a real sanity saver when you realise you have just confirmed erasure of the wrong 2000 words! Incidentally, Wordpower imposes no limit on the length of your text. One file could fill a whole disc, and even discs can be linked together for printing.

Printing is an area where Wordpower scores very highly. First of all, you can carry on typing new text (or indeed amend existing text) while printing takes place in the background. You can print from the computer's memory or straight from disc, link files together, and print as many copies as you want, complete with page and copy numbers, headings and footings, draft, elite or NLQ, enlarged, condensed, double strike, emphasised, italic or superscript type if required. Line spacing, page length and various other factors can be controlled by plain English instructions in the text.

Thousands of words must have been written explaining how to achieve simple printer effects or print pound signs using other BBC word processors. The Wordpower solution is quite simple. For example, to turn underlining on just press Ctrl-P (for 'print code') followed by U. You can do this even in the middle of a word, and the U appears in 'inverse video' on the screen (although the U takes up a space on the screen Wordpower compensates for this and the printed format will still be correct). If pound signs do not print correctly on your printer, press Ctrl-P followed by P instead and the correct character can be be printed.

Each letter of the alphabet can stand for almost any sequence of codes, so a single letter could select underlined enlarged emphasised italic printing. Because of the huge range of possibilities, Wordpower makes no attempt to represent the effects directly on screen, but the inverse video makes it easy enough to tell where they start and end. The program is supplied with pre-defined sequences to suit Epson and Juki printers, but you can easily create your own via the UVF. You can even include a simple instruction in your text to change all 26 definitions during a print run: this means that the same text can be printed on two incompatible printers by changing a single word.

I recently took delivery of Wordpower Version 4 which introduces the major new feature of **Power Fonts**. I (or rather my colleagues and students) can now word-process in Russian or Greek, not to mention Faeroese, Finnish, French, German etc ... and mathematics (see examples).

Up to 219 different characters can now be seen on the screen at once, just as they will be printed. This new dimension in WYSIWYG-ness even extends to

¡ELIMINA MEJOR LOS ESCAPES! Protectores azules anti-humedad más ajustable. Asegúrese de que la cinta azul está bien adheri

ANNU BATTRE SKYDD MOT LACKAGE! Nu med bättre läckageskyd och förbättrad passform. Återförslut sedan blöjan och kontrolle fast på blöjan.

PIÙ PROTEZIONE DALLE FUORIUSCITE DI PIPÌ! Per una migliore ve di pipì, tirare il pannolino verso l'alto e distendere i bordi elast

MEHR AUSLAUFSICHERHEIT! Blaue Auslaufsperre an der Taille, m Sie die Klebebänder und -stellen vollkommen frei von Puder, Öl

MEILLEURE PROTECTION ANTI-FUITES! Pour contrôler, ouvrez seu

Το WORDPOWER περιγράφεται λεπτομερώς στο τετρασέλιδο πράσι έχετε δει. Με λίγα λόγια το WORDPOWER είναι ένα δυναμικό αλλά ονομάζουν το καλύτερο που υπάρχει για BBC ή Electron.

Ленингра́дской киносту́дии ну́жно бы́ло снять го́рные ви́ды. Ша́ня. План путеше́ствия был тако́й; в ма́е экспеди́ция вы́лет гора́х Тянь-Ша́ня и пото́м вернётся в Ленингра́д.

multi-line physics equations complete with boxes, brackets, integral signs and the like. Typing the extra characters at the keyboard is delightfully simple and easy to remember. Each Power Font is supplied with a set of notes and a specialised disc to use instead of the normal Wordpower disc; this may still be removed as soon as the program is running.

A high-quality printing option, called **Powerfont NTQ**, is also available. This was specially developed from Permanent Memory Systems' excellent Multi-Font NTQ program, which provides variable height and width and right-justified proportional spacing. Ian Copestake Software has designed language and maths/physics NTQ fonts to an impressive standard - my students and colleagues have been drooling over their print-outs ever since the new package arrived.

$$N_{xy} = -\frac{ab.D}{32}\;\frac{\sum_{m=1}^{\infty}\sum_{n=1}^{\infty} a_{mn}{}^{2}\left(\frac{m^2\pi^2}{a^2}+\frac{n^2\pi^2}{b^2}\right)^2}{\sum_{m=1}^{\infty}\sum_{n=1}^{\infty}\sum_{i=1}^{\infty}\sum_{j=1}^{\infty} a_{mn}\,a_{ij}\,\frac{m\,n\,i\,j}{(m^2-i^2)\,(j^2-n^2)}}$$

If you need special character sets of any kind you must not miss the latest version of Wordpower. The author offers to design free Power Fonts to your specification - recent customers include a word processing bureau in Greece and a bible translation institute in Sweden.

It is rare to be able to speak to the author of a program on the telephone, but Ian Copestake always seems to be available to answer any little query. This is an extremely important consideration for anyone using software in their work. We have also been regularly informed of program enhancements, updates being offered at a nominal cost.

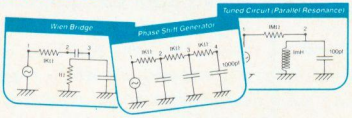
The Wordpower manual is thorough and detailed, running to 80 A5 pages, and has recently been improved by the addition of a good index and a lay-flat binding. Although easy to follow, the sheer amount of information could be daunting, although the Ctrl-H Help option pre-empts this I think. But Wordpower's friendly approach encourages you to learn about the more advanced features as and when you need them; in spite of its power the program never feels difficult.

Wordpower works with BEEBUG's Spellcheck series software and Computer Concepts' Spellmaster (except in immediate mode), and it is available in just about every disc format.

It is always easy to find fault with anything produced by someone else and I must say that there have been 'faults' with Wordpower's earlier versions. I cannot clearly remember just how many versions have been produced because Ian Copestake is always willing to listen to constructive criticism and I believe that I have given him quite a lot since Wordpower was released.

I can thoroughly recommend Wordpower. For power, ease of use, and sheer enjoyment it has to be the outright winner. Although quite widely known in the educational field, this outstanding product seems to have attracted less than its fair share of recognition amongst other users. I hope this review will do something to redress the balance. Whatever you need from a word-processor, and even if you have one already, you will not regret buying Wordpower. B

CIRCUIT ANALYSIS



Colin Attenborough presents a powerful piece of software which can analyse and display the response of a wide variety of electronic and electrical circuits.

INTRODUCTION

Circuit analysis programs are used by electronics engineers to predict the performance of electronic and electrical circuits. And although such programs have been written for the BBC micro, the one presented here is novel because the circuit description is written as a text file. Analysis programs for larger computers use this method. To write the file, a word processor is needed: Wordwise is used in the version shown here, but since the file is pure ASCII it could equally well be created with the Master Editor or with View. Since the program uses data stored in files, a disc drive is all but essential.

Because of the memory limitations of the Beeb, I have split the tasks performed by the package across two programs. One reads the circuit file, analyses the circuit, and writes the voltage at each circuit node into an output file. (A node is a point where two or more circuit elements join.) The second program, automatically chained by the first, reads the output file, and displays the results in tabular or graphical form. The sequence of operations is thus:

1. Write a circuit file.
2. Run the analysis program to generate an output file.
3. Select the display option, which chains the second program, and displays the results in tabular and graphical form.

Listing 1 contains the analysis program, and although it's quite long, it omits active devices (transistors, amplifiers, etc). These are dealt with in an extended version of the program which appears on the magazine disc, and which may be obtained as a listing for the price of an SAE (see end of article). With the version given in listing 1, you'll be able to analyse circuits containing resistors, capacitors and inductors, including lossy inductors. The results file will be written, but you'll have to wait for next month's program before you can display the results properly. In the mean-time, a minimal display program is provided in listing 2. This simply prints the real and imaginary components of the node voltages at nodes 1 and 2 of a circuit (assuming a constant current source of 1 ampere).

```
R1,1,2,1k
C1,2,E,1u
FMIN=100,FMAX=200,DF=10
IN=1
```

Figure 1. Circuit description file

THE CIRCUIT FILE

The circuit file tells the computer what components are in the circuit, how they are connected, where the input is connected and at what frequencies the circuit is to be analysed. This month I'll describe only those features used in the example circuit file. Figure 1 is a circuit file describing the circuit shown in figure 2 (a simple capacitor-resistor integrator). Figure 3 is the display obtained using the simple display program given in listing 2.

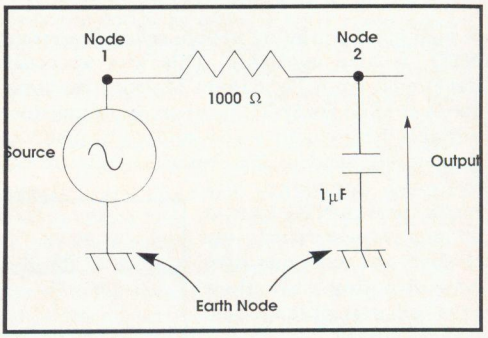


Figure 2. Circuit diagram

Resistors are given names starting with R, capacitors names beginning with C. The nodes in the circuit to which the component is connected follow: for the common earth node, the letter E is used. The component value comes last. The various fields are separated by commas. Thus in our example R1,1,2,1k means that resistor R1 is between nodes 1 and 2, and has the value 1k (or 1000 ohms). Figure 4 gives

```
Frequency     ReV(1)      ImV(1)      ReV(2)      ImV(2)
100           1000        -1591.5     0           -1591.5
110           1000        -1446.9     0           -1446.9
120           1000        -1326.3     0           -1326.3
130           1000        -1224.3     0           -1224.3
140           1000        -1136.8     0           -1136.8
150           1000        -1061       0           -1061
160           1000        -994.72     0           -994.72
170           1000        -936.21     0           -936.21
180           1000        -884.19     0           -884.19
190           1000        -837.66     0           -837.66
200           1000        -795.77     0           -795.77

                Press any key
```

Figure 3. Output from program 2.

a list of acceptable multipliers: you will see that the example file uses "u" to indicate that the capacitor's value is given in microfarads.

Frequency information is of the form FMIN=<data>, FMAX=<data>, DF=<data> where analysis is performed from FMIN to FMAX in steps of DF. The circuit is driven at a node defined by the statement IN=<node number>. The other side of the source is connected to the earth node, E.

p or P	10^-12	k or K	10^3
n or N	10^-9	M	10^6
u or U	10^-6	g or G	10^9
m	10^-3		

Figure 4 Table of multipliers

USING THE PROGRAM

Type in and save programs 1 and 2. Program 2 should be saved under the name "DISP". Use Wordwise or some other word processor to write the example circuit file given in figure 1, and save this with the filename "CR" to your current directory ($ for DFS users). ADFS users will need to create a sub-directory named "O" at this point (Type *CDIR O). You should now run program 1. The disc will be catalogued, and you should select the required circuit filename ("CR" - not "O.CR").

The file's contents will be read and displayed, and the program will then analyse the response of the circuit at each of the desired frequencies. The frequency being processed at any instant is displayed, and the results written to the new file "O.CR". The display program is then chained. It will read in the file "O.CR", and should produce a display similar to that in figure 3, giving the real and imaginary components of the voltage at nodes 1 and 2 over the specified frequency range.

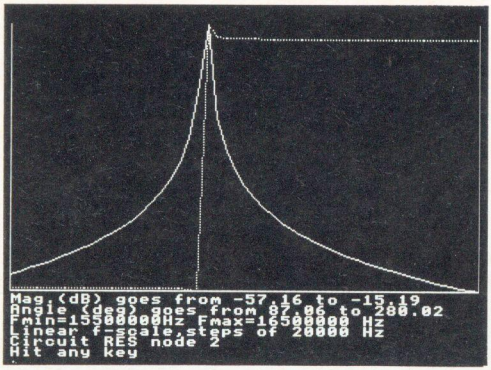
Next month the complete display program will be published, and we will discuss the syntax for all the types of component which the program can handle.

This month's magazine disc contains a version of the complete analysis program, capable of analysing active as well as passive circuits. If readers would like a listing of the full program they should send an A5 SAE to the editorial address, with the envelope marked "Circuit Analysis".

Listing 1

```
  10 REM Program Circuit Analysis
  20 REM Version B 0.4
  30 REM Author  C.Attenborough
  40 REM BEEBUG  Jan/Feb 1988
  60 REM Program subject to copyright
  70 :
 100 ON ERROR CLOSE#0:GOTO 3750
 110 MODE7:PROCinit:REPEAT
 120 CLS:PROCcleararrays
 130 *CAT
 140 INPUT'TAB(10);"Circuit file?  "CF$
 150 OF$="O."+CF$:CLS
 160 PRINTTAB(15,12)"Finding ";CF$
 170 X%=OPENIN CF$:CLS
 180 REPEAT:PROCinstruction
 190 UNTIL EOF#X%:CLOSE#0:VDU7
 200 PRINT'SPC4 "Press any key"
 210 G=GET
 220 VDU23,1,0;0;0;0;:@%=&0406:CLS
 230 PRINTTAB(6,8)"Start frequency ",fi
;" Hz"
 240 PRINTTAB(6,12)"Stop frequency  ",f
a;" Hz"
 250 PROCana:MODE 7:PROCoptions:@%=&90A
 260 IF choice%=1 @%=&90A:Z%=OPENIN OF$
:CHAIN"DISP"
 270 END
 280 :
1000 DEF PROCinit
1010 DIM R(30),C(30),L(30),Q(30)
1020 DIM rs 30,rf 30,cs 30,cf 30
1030 DIM ls 30,lf 30,qs 30,qf 30
1040 DIM G(30,30),B(30,30),RI(30)
1050 DIM II(30),RV(30),IV(30)
1060 ENDPROC
1070 :
1080 DEF PROCcleararrays
1090 FOR J%=0 TO 30
1100 rs?J%=255:rf?J%=255:cs?J%=255
1110 cf?J%=255:ls?J%=255:lf?J%=255
1120 qs?J%=255:qf?J%=255
1130 NEXT
1140 rcount%=0:ccount%=0:lcount%=0
```

```
 1150 qcount%=0:gmcount%=0
 1160 I%=0:J%=0:nodemax%=0:ENDPROC
 1170 :
 1180 DEF PROCana
 1190 F=fi:Y%=OPENOUT OF$
 1200 PRINT #Y%,CF$,fi,fa,df,in%,st%,nod
emax%,steps
 1210 REPEAT
 1220 PRINTTAB(0,16)"Calculating respons
e at ",F" Hz"SPC9
 1230 M%=nodemax%:w=2*PI*F:PROCcleargb
 1240 IF ?rs<>255 PROCloadr
 1250 IF ?cs<>255 PROCloadc
 1260 IF ?ls<>255 PROCloadl
 1270 IF ?qs<>255 PROCloadq
 1280 RI(in%)=1:PROCsolve
 1290 ON st% GOSUB 3600,3630,3660
 1300 UNTIL F>fa+0.1*df
 1310 CLOSE#Y%:ENDPROC
 1320 :
 1330 DEF PROCoptions
 1340 REPEAT
 1350 PRINTTAB(5,8)"1 Display results"
 1360 PRINTTAB(5,16)"2 Quit"
 1370 G%=GET:choice%=G%-48
 1380 UNTIL choice%>0 AND choice%<3
 1390 ENDPROC
 1400 :
 1410 DEF PROCloadr
 1420 L%=0:REPEAT
 1430 PROCcond(rs?L%,rf?L%,1/R(L%))
 1440 L%=L%+1:UNTIL rs?L%=255:ENDPROC
 1450 :
 1460 DEF PROCloadc
 1470 L%=0:REPEAT
 1480 PROCsus(cs?L%,cf?L%,w*C(L%))
 1490 L%=L%+1:UNTIL cs?L%=255:ENDPROC
 1500 :
 1510 DEF PROCloadl
 1520 L%=0:REPEAT
 1530 PROCsus(ls?L%,lf?L%,-1/(w*L(L%)))
 1540 L%=L%+1:UNTIL ls?L%=255:ENDPROC
 1550 :
 1560 DEF PROCloadq
 1570 L%=0:REPEAT
 1580 rloss=w*L(L%)*Q(L%)
 1590 PROCcond(qs?L%,qf?L%,1/rloss)
 1600 L%=L%+1:UNTIL qs?L%=255:ENDPROC
 1610 :
 1620 DEF PROCcleargb
 1630 FOR J%=1 TO M%
 1640 RV(J%)=0:RI(J%)=0:IV(J%)=0:II(J%)=
0
 1650 FOR K%=1 TO M%
 1660 G(J%,K%)=0:B(J%,K%)=0
 1670 NEXT:NEXT:ENDPROC
 1680 :
 1690 DEF PROCsus(s%,f%,bc)
 1700 B(s%,s%)=B(s%,s%)+bc
 1710 B(f%,f%)=B(f%,f%)+bc
 1720 B(s%,f%)=B(s%,f%)-bc
 1730 B(f%,s%)=B(f%,s%)-bc
 1740 ENDPROC
 1750 :
 1760 DEF PROCcond(s%,f%,gg)
 1770 G(s%,s%)=G(s%,s%)+gg
 1780 G(f%,f%)=G(f%,f%)+gg
 1790 G(s%,f%)=G(s%,f%)-gg
 1800 G(f%,s%)=G(f%,s%)-gg
 1810 ENDPROC
 1820 :
 1830 DEF PROCprod(RA,IA,RB,IB)
 1840 RP=RA*RB-IA*IB:IP=IA*RB+IB*RA
 1850 ENDPROC
 1860 :
 1870 DEF PROCdiv(RA,IA,RB,IB)
 1880 RQ=RA*RB+IA*IB:IQ=IA*RB-IB*RA
 1890 DQ=RB*RB+IB*IB:RQ=RQ/DQ:IQ=IQ/DQ
 1900 ENDPROC
 1910 :
 1920 DEF PROCsolve
 1930 FOR L%=2 TO M%:FOR J%=L% TO M%
 1940 PROCdiv(G(J%,L%-1),B(J%,L%-1),G(L%
-1,L%-1),B(L%-1,L%-1))
 1950 FOR K%=L%-1 TO M%
 1960 PROCprod(RQ,IQ,G(L%-1,K%),B(L%-1,K
%))
 1970 G(J%,K%)=G(J%,K%)-RP:B(J%,K%)=B(J%
,K%)-IP
 1980 NEXT
 1990 PROCprod(RQ,IQ,RI(L%-1),II(L%-1))
 2000 RI(J%)=RI(J%)-RP:II(J%)=II(J%)-IP
 2010 NEXT:NEXT
 2020 FOR J%=M% TO 1 STEP-1
 2030 RV(J%)=RI(J%):IV(J%)=II(J%)
 2040 IFJ%=M% THEN 2090
 2050 FOR K%=J%+1 TO M%
 2060 PROCprod(G(J%,K%),B(J%,K%),RV(K%),
IV(K%))
 2070 RV(J%)=RV(J%)-RP:IV(J%)=IV(J%)-IP
 2080 NEXT
 2090 PROCdiv(RV(J%),IV(J%),G(J%,J%),B(J
%,J%))
 2100 RV(J%)=RQ:IV(J%)=IQ:NEXT
 2110 PRINT#Y%,F
 2120 FOR J%=1 TO nodemax%
 2130 PRINT#Y%,RV(J%),IV(J%)
 2140 NEXT
 2150 ENDPROC
 2160 :
 2170 DEF PROCinstruction
 2180 PROCextractline:PROCcheck
 2190 IF instok PROCextractdata ELSE PRO
Ccomplain:ENDPROC
 2200 ENDPROC
 2210 :
```




```
 2220 DEF PROCextractline
 2230 COMP$="":REPEAT
 2240 c%=BGET#X%:COMP$=COMP$+CHR$(c%)
 2250 UNTIL c%=13 OR EOF#X%
 2260 IF NOT EOF#X% COMP$=LEFT$(COMP$,LE
N(COMP$)-1)
 2270 PRINTCOMP$
 2280 ENDPROC
 2290 :
 2300 DEF PROCcheck
 2310 fc%=ASC(COMP$):instok=FALSE:nc%=0
 2320 IF fc%=-1 OR fc%=13 instok=TRUE:EN
DPROC
 2330 FOR K%=1 TO LEN(COMP$)
 2340 IF MID$(COMP$,K%,1)="," nc%=nc%+1
 2350 NEXT
 2360 firstchar$=LEFT$(COMP$,1)
 2370 IF firstchar$="*" THEN instok=TRUE
 2380 IF firstchar$="R" AND nc%=3 instok
=TRUE
 2390 IF firstchar$="C" AND nc%=3 THEN i
nstok=TRUE
 2400 IF firstchar$="L" AND (nc%=3 OR nc
%=4) instok=TRUE
 2410 IF firstchar$="F" instok=TRUE
 2420 IF firstchar$="I" AND nc%=0 instok
=TRUE
 2430 ENDPROC
 2440 :
 2450 DEF PROCtwoterm
 2460 K%=1:sec%=0:REPEAT
 2470 NODE$="":REPEAT
 2480 N$=MID$(COMP$,K%,1)
 2490 IF N$<>"," NODE$=NODE$+N$
 2500 K%=K%+1
 2510 UNTIL N$="," OR K%-1=LEN(COMP$)
 2520 sec%=sec%+1
 2530 IF sec%=2 I%=VAL(NODE$)
 2540 IF sec%=3 J%=VAL(NODE$)
 2550 IF J%>nodemax% nodemax%=J%
 2560 IF I%>nodemax% nodemax%=I%
```

```
 2570 IF sec%=4 V=FNeval(NODE$)
 2580 IF (firstchar$="L" AND sec%=4) L(l
count%)=V:?(ls+lcount%)=I%:?(lf+lcount%)
=J%:lcount%=lcount%+1
 2590 IF (firstchar$="R" AND sec%=4) R(r
count%)=V:?(rs+rcount%)=I%:?(rf+rcount%)
=J%:rcount%=rcount%+1
 2600 IF (firstchar$="C" AND sec%=4) C(c
count%)=V:?(cs+ccount%)=I%:?(cf+ccount%)
=J%:ccount%=ccount%+1
 2610 IF sec%=5 Q(qcount%)=FNeval(NODE$)
:?(qs+qcount%)=I%:?(qf+qcount%)=J%:qcoun
t%=qcount%+1
 2620 UNTIL K%-1=LEN(COMP$)
 2630 IF I%=J% PROCcomplain
 2640 ENDPROC
 2650 :
 2660 DEF PROCcomplain
 2670 VDU7
 2680 PRINT"Error detected in ";COMP$:CL
OSE#0:PRINT
 2690 PRINT"Press any key to edit circui
t file"
 2700 G%=GET:PROCreload:ENDPROC:END
 2710 :
 2720 DEF PROCextractdata
 2730 IF fc%=-1 OR fc%=13 ENDPROC
 2740 IF firstchar$="R" PROCtwoterm
 2750 IF firstchar$="C" PROCtwoterm
 2760 IF firstchar$="L" PROCtwoterm
 2770 IF firstchar$="F" PROCfdata
 2780 IF firstchar$="I" PROCipop
 2790 ENDPROC
 2800 :
 2810 DEF FNeval(a$)
 2820 IF ASC(RIGHT$(a$,1))<58=VAL(a$)
 2830 r$=RIGHT$(a$,1)
 2840 IF r$="P" OR r$="p" f=1E-12
 2850 IF r$="N" OR r$="n" f=1E-9
 2860 IF r$="U" OR r$="u" f=1E-6
 2870 IF r$="K" OR r$="k" f=1E3
 2880 IF r$="G" OR r$="g" f=1E9
 2890 IF RIGHT$(a$,1)="m" f=1E-3
 2900 IF RIGHT$(a$,1)="M" f=1E6
 2910 =f*VAL(LEFT$(a$,LEN(a$)-1))
 2920 :
 2930 DEF PROCfdata
 2940 J%=1:F$=""
 2950 REPEAT
 2960 F$=F$+MID$(COMP$,J%,1):J%=J%+1
 2970 UNTIL MID$(COMP$,J%,1)="="
 2980 IF F$="F" PROCsinglefreq:ENDPROC
 2990 IF F$="FMIN" PROCsweep:ENDPROC
 3000 PROCcomplain:ENDPROC
 3010 :
 3020 DEF PROCsweep
 3030 REPEAT
 3040 F$=F$+MID$(COMP$,J%,1):J%=J%+1
```

```
 3050 UNTIL MID$(COMP$,J%,1)","
 3060 G$=RIGHT$(F$,(LEN(F$)-5))
 3070 fi=FNeval(G$):F$=""
 3080 FOR K%=1 TO 5
 3090 F$=F$+MID$(COMP$,J%+K%,1)
 3100 NEXT
 3110 IF F$<>"FMAX=" PROCcomplain:ENDPRO
C
 3120 F$="":J%=J%+6
 3130 REPEAT
 3140 F$=F$+MID$(COMP$,J%,1):J%=J%+1
 3150 UNTILMID$(COMP$,J%,1)","
 3160 fa=FNeval(F$)
 3170 J%=J%+1:F$=""
 3180 REPEAT
 3190 F$=F$+MID$(COMP$,J%,1):J%=J%+1
 3200 UNTIL MID$(COMP$,J%,1)"=" OR MID$
(COMP$,J%,1)","
 3210 IF F$="DF" PROClinear:ENDPROC
 3220 IF F$="LOG" PROClog:ENDPROC
 3230 PROCcomplain:ENDPROC
 3240 :
 3250 DEF PROClinear
 3260 F$=""
 3270 REPEAT:J%=J%+1
 3280 F$=F$+MID$(COMP$,J%,1)
 3290 UNTIL J%=LEN(COMP$)
 3300 df=FNeval(F$)
 3310 steps=((fa-fi)/df)+1
 3320 st%=2:ENDPROC
 3330 :
 3340 DEF PROClog
 3350 F$=""
 3360 REPEAT:J%=J%+1
 3370 F$=F$+MID$(COMP$,J%,1)
 3380 UNTIL J%=LEN(COMP$)
 3390 IF LEFT$(F$,6)<>"STEPS=" PROCcompl
ain
 3400 F$=RIGHT$(F$,LEN(F$)-6)
 3410 steps=FNeval(F$):lre=LOG(fa/fi)
 3420 ratio=10^(LRE/(steps-1))
 3430 df=ratio*fi:st%=3:ENDPROC
 3440 :
 3450 DEF PROCipop
 3460 i$=LEFT$(COMP$,3)
 3470 IF i$<>"IN=" PROCcomplain
 3480 io$=MID$(COMP$,4):i$="":J%=1
 3490 REPEAT
 3500 i$=i$+MID$(io$,J%,1)
 3510 J%=J%+1:UNTIL J%=LEN(io$)+1
 3520 in%=VAL(i$)
 3530 ENDPROC
 3540 :
 3550 DEF PROCsinglefreq
 3560 F$=RIGHT$(COMP$,LEN(COMP$)-2)
 3570 fi=FNeval(F$):fa=fi:df=fa
 3580 st%=1:steps=1:ENDPROC
 3590 :
```

```
 3600 REM Single frequency:st%=1
 3610 F=2*fa:RETURN
 3620 :
 3630 REM Linear sweep:st%=2
 3640 F=F+df:RETURN
 3650 :
 3660 REM Log sweep:st%=3
 3670 F=F*ratio:RETURN
 3680 :
 3690 DEF PROCreload
 3700 load$="KEY9 *WORDWISE|M 2"+CF$+"|M
 |["
 3710 OSCLI load$
 3720 A%=138:X%=0:Y%=137:CALL &FFF4
 3730 ENDPROC
 3740 :
 3750 ON ERROR OFF
 3760 VDU23,1,1;0;0;0;:@%=&90A:*FX21,0
 3770 IF ERR=222 CLS:PRINTTAB(12,12);CF$
;" not found";TAB(15,14);"Press any key"
:G%=GET:RUN
 3780 IF ERR=18 CLS:PRINTTAB(8,12);"You
have missed a node";TAB(15,14);"Press an
y key":G%=GET:PROCreload:END
 3790 IF ERR=26 CLS:PRINTTAB(8,12);"Data
 missing from file";TAB(15,14);"Press an
y key":G%=GET:PROCreload:END
 3800 REPORT:PRINT" at line ";ERL:END
```

* * * * *

Listing 2

```
   10 REM Minimal Display ver 0.3
   20 REM Filename .>Disp
   30 REM Author C.Attenborough
   40 :
   50 ON ERROR CLOSE#0:MODE7:REPORT:PRIN
T" at line ";ERL:END
   60 MODE 3
   70 INPUT #Z%,CF$,fi,fa,df,in%,st%,nod
emax%,steps
   80 PROCreadfile
   90 PRINT:PRINTTAB(20);"Press any key"
  100 IF GET CLOSE#0:@%=&90A:END
  110 :
  120 DEF PROCreadfile
  130 @%=&508
  140 PRINTTAB(0);"Frequency";
  150 PRINTTAB(15);"ReV(1)";TAB(30);"ImV
(1)";
  160 PRINTTAB(45);"ReV(2)";TAB(60);"ImV
(2)"
  170 REPEAT
  180 INPUT #Z%,F,re1,im1,re2,im2
  190 PRINTTAB(0);F;TAB(15);re1;TAB(30);
im1;
  200 PRINTTAB(45);re2;TAB(60);im2
  210 UNTILEOF#Z%:ENDPROC
```

PRINTING & TRANSPOSING MUSIC

This clever program by Dr Wass makes the process of transcribing music from the keyboard to the screen or printer comparatively easy, with a wide range of musical notes and other symbols. And it will automatically transpose the composition at the same time if required.

'Mustran' is a program to display and print music on normal five-line staves, with automatic transposition if selected. It will cope with all the different types of note (crotchet, quaver etc), and sharps, flats, naturals, and rests as well other features. All input is coded and entered via the standard keyboard using a simple, easy-to-learn system.

USING THE PROGRAM

Type the program in and save it away to disc or tape. When the program is run the screen first prompts for transposition up ('U') or down ('D') and for the number of steps. Transposition will effectively raise or lower a tune on the musical scale, with a corresponding change of key. If transposition is not required then either 'U'or 'D' may be pressed, followed by 0. Note that neither here, or elsewhere on input, is the use of Return needed.

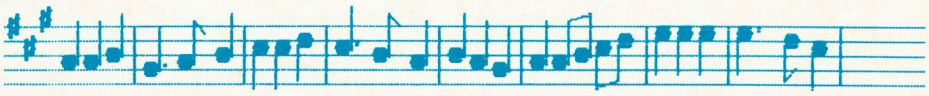
Either mode 4 or mode 0 may be selected for the screen display. Mode 4 may be used for demonstration and easier reading of prompts, but mode 0 will usually be needed to display a full line of music.

Three staves are then displayed on the screen, and the lines of the lowest stave are numbered 01 ,03, 05, 07 ,09, and the spaces 02, 04, 06, 08, for reference. For clarity, only the even numbers are displayed on the screen. Above the stave the numbers continue with 10, 11, 12, etc, and below the stave the numbers descend, with 00 ,-1, -2, etc displayed. Notes are entered (using two-letter codes) and displayed on the bottom stave, and transposed by the program on to the top stave. Once this line of music is complete further notes may be entered and transposed to the middle stave.

A flashing prompt appears at the bottom of the screen for the entry of a 'symbol' and 'position'. All the symbols are entered with one of the following codes:

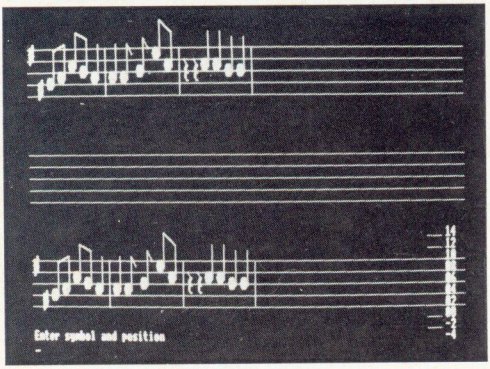
SH	--	Sharp
NA	--	Natural
FL	--	Flat
WW	--	Semibreve ('W'=White; 'S' is used for semiquaver)
MU	--	Minim with stem up
MD	--	Minim with stem down
CU	--	Crotchet with stem up
CD	--	Crotchet with stem down
QU	--	Quaver with stem up
QD	--	Quaver with stem down
SU	--	Semiquaver with stem up
SD	--	Semiquaver with stem down
DT	--	Dot
RW	--	Semibreve rest
RM	--	Minim rest
RC	--	Crotchet rest
RQ	--	Quaver rest
RS	--	Semiquaver rest
BL	--	Barline
CS	--	Change Stave
BQ	--	Single joining bar (quavers)
BS	--	Double bar (semi-quavers)
E	--	Erase
P	--	Print

Enter the two digit code as required, followed immediately by a TWO-digit position number. Once these codes have been entered they will be displayed, and if they are correct then any key except 'C' may be pressed. This will display the symbol in its correct place on the lower stave and also in its transposed position on the top stave. If 'C' is pressed after the entries have been displayed, then they will be ignored, and you will be prompted for a new entry. 'C' may



also be pressed with the same effect after the entry of just the two-letter code.

If you need to display a pair of quavers or semiquavers with a bar or double bar joining their stems, then they should first be entered and displayed as crotchets. The bar can then be added by pressing 'BQ' or 'BS' as appropriate.



If any symbol that has just been displayed on the stave needs to be changed then key 'E' may be used. This will draw a black vertical stripe on both staves, and repeated use will erase a rectangular patch. This is rather slow, but it is useful because the left-hand edge of the next symbol will appear at the position of the last black stripe. If more space is needed to the right of the last displayed music character, then entering any two-letter combination, other than those in the above list, will 'print' an invisible character, and the position for the next character will be moved along by that amount.

This position can be found by pressing key 'E' once, and the space can be reduced by repeated presses of 'E'. The 'E' key can also be used to reduce the space between symbols, for example to compact a key signature.

When the top line of music is complete, the code 'CS' (Change Stave) should be entered. The bottom stave will then clear and entering symbols can continue. They will be displayed as before on the bottom stave but now the transposed version will appear on the middle stave. The next use of the 'CS' command will clear all staves ready to re-start at the top.
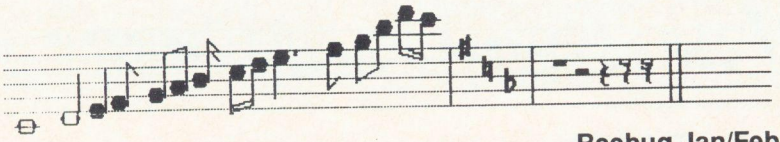
When the two upper staves are full the screen display may be sent to a printer by typing 'P'. As written the program calls Computer Concepts' Printmaster ROM to produce a screen dump using *GDUMP 0 1 2 1 0 (see line 2490). This gives full-size music, ready for playing. This line could be changed to call other printer dump routines (e.g. BEEBUG's Dumpmaster), or a routine of your own. In fact, the print routine may be called at any time, not just when both staves are complete.

There is no provision for entering time or clef signatures or marks of expression, etc., though these could be added if memory were available.

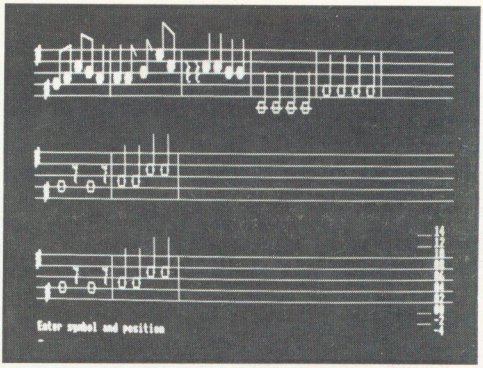
If the transposition involves a key change the key signature will not be changed automatically. However, the sharps or flats needed for the new key can be entered on the bottom stave and so positioned that they will appear in their correct positions on the upper staves after transposition. Thus, for a transposition from the key of F major, say, with one flat, to G major, with one sharp, notes will be raised by two steps. If then a sharp is entered on line 07 on the bottom stave it will appear in its correct position on line 09 on the upper staves.

The illustration shows a line of a well-known tune transposed up a whole tone so that it can be played by a B flat clarinet or trumpet. The second stave shows all the symbols that are available.

```
   10 REM Program MUSTRAN
   20 REM Version B1.1
   30 REM Author  C.A.A. Wass
   40 REM BEEBUG  Jan/Feb 1988
   50 REM Program subject to copyright
   60 :
  100 MODE 7:PROCtitle:ON ERROR GOTO 250
  110 PRINT'"Transpose music?"'"Enter 'U
' or 'D' and number of steps."
  120 T=GET:VDU T,58:NS=GET:VDU NS,13,10
  130 XST1=0:YST1=0:XST2=0:YST2=0:LS=0:L
F=0
  140 UD=NS-48+2*(NS-48)*(T=68)
```

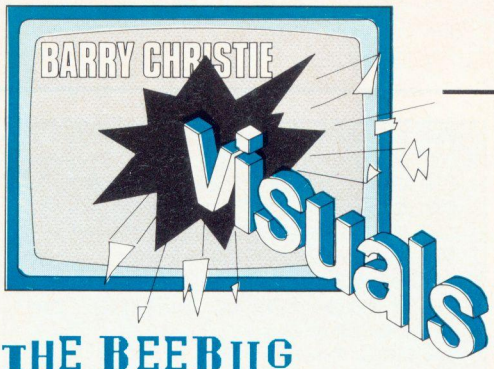


```
  150 PRINT'"MODE -- 0 or 4 ?"
  160 M=GET-48:MODE M
  170 IF M=4 SP=8 ELSE SP=4
  180 XX=4:STNo=2:STS=288:KR$=""
  190 JU=STNo*STS+16*UD
  200 PROCstave(2)
  210 PROCvdu:PROCsym
  220 REPEAT:PROCchoosesym:UNTIL FALSE
  230 END
  240 :
  250 ON ERROR OFF:MODE 7:REPORT
  260 PRINT" at line ";ERL:END
  270 :
 1000 DEF PROCtitle
 1010 FOR I=1 TO 2:PRINTTAB(5,I)CHR$141C
HR$129CHR$157CHR$131"MUSIC COMPOSITION
"CHR$156:NEXT I
 1020 ENDPROC
 1030 :
 1040 DEF PROCvdu
 1050 VDU23,224,63,127,127,255,255,127,1
27,63
 1060 VDU23,225,0,128,128,192,192,128,12
8,0
 1070 VDU23,226,63,64,64,192,192,64,64,6
3
 1080 VDU23,227,128,64,64,96,96,64,64,12
8
 1090 VDU23,228,0,128,255,0,0,255,128,0
 1100 VDU23,230,192,32,32,48,48,32,32,19
2
 1110 VDU23,231,0,6,6,103,110,126,118,23
0
 1120 VDU23,232,103,110,126,118,230,96,9
6,0
 1130 VDU23,233,0,96,96,96,110,126,118,1
02
 1140 VDU23,234,102,110,126,118,6,6,6,0
 1150 VDU23,235,96,96,96,96,96,124,126,9
9
 1160 VDU23,236,99,99,102,108,104,112,0,
0
 1170 VDU23,237,0,0,0,0,255,0,0,0
 1180 VDU23,238,64,32,16,8,14,28,56,56
 1190 VDU23,239,24,30,48,48,48,16,14,0
 1200 VDU23,240,0,130,254,30,4,4,6,8
 1210 VDU23,241,24,24,24,24,24,4,0,0
 1220 VDU23,242,0,130,254,30,132,252,24,
8
 1230 VDU23,243,255,255,255,0,0,0,0,0
 1240 VDU23,244,0,0,0,0,0,255,255,255
 1250 ENDPROC
 1260 :
 1270 DEF PROCsym
 1280 SH$=CHR$231+CHR$8+CHR$10+CHR$232
 1290 NA$=CHR$233+CHR$8+CHR$10+CHR$234
 1300 FL$=CHR$235+CHR$10+CHR$8+CHR$236
 1310 RW$=CHR$243:RM$=CHR$244
 1320 RC$=CHR$238+CHR$10+CHR$8+CHR$239
 1330 RQ$=CHR$240+CHR$10+CHR$8+CHR$241
 1340 RS$=CHR$242+CHR$10+CHR$8+CHR$241
 1350 WW$=CHR$226+CHR$230
```




```
 1360 BB$=CHR$224+CHR$225
 1370 DT$="."
 1380 ENDPROC
 1390 :
 1400 DEF PROCstave(SS)
 1410 VDU5:FOR S=0 TO SS
 1420 FOR L=0 TO 4
 1430 MOVE0,132+32*L+288*S:PLOT21,1276,1
32+32*L+288*S
 1440 NEXT L,S
 1450 FOR LiNo=-4 TO 14 STEP 2
 1460 Ypos=124+16*LiNo:Yval$=FNjust(LiNo
)
 1470 MOVE1220,Ypos:PRINT Yval$
 1480 IF Rflag MOVE1168,Ypos+8:PLOT21,12
08,Ypos+8
 1490 NEXT LiNo
 1500 ENDPROC
 1510 :
 1520 DEF FNjust(n):Rflag=FALSE
 1530 IF n>=0 AND n<10 n$="0"+STR$(n) EL
SE n$=STR$(n):IF n<13 Rflag=TRUE
 1540 =n$
 1550 :
 1560 DEF FNpos
 1570 IF J$="R" OR J$="B" OR K$="CS" YP=
0:=TRUE
 1580 Y$=GET$:IF Y$="C" =FALSE
 1590 IF INSTR("-10",Y$)=0 GOTO 1580
 1600 IF Y$=";" Y$="+"
 1610 Y1=GET-48-10*(Y$="+")
 1620 Y2$=Y$+STR$(Y1)
 1630 YP=VAL(Y2$):YY=132+16*YP
 1640 VDU4:PRINTTAB(3,30);Y2$
 1650 =TRUE
 1660 :
 1670 DEF PROCchoosesym
 1680 VDU4:PRINTTAB(0,30)"Enter symbol a
nd position"
 1690 J$=GET$:PRINTTAB(0,30)SPC26
 1700 IF J$="P" PROCprint:ENDPROC
 1710 IF J$="E" PROCerase:ENDPROC
 1720 I$=GET$:K$=J$+I$
 1730 IF K$="CS" PROCchst:ENDPROC
 1740 IF K$="CB" PROCbar:ENDPROC
```

```
 1750 IF K$="BQ" OR K$="BS" PROCbar:ENDP
ROC
 1760 VDU4:PRINTTAB(0,30)K$;";"
 1770 IF NOT FNpos ENDPROC
 1780 VDU4:PRINTTAB(6,30)"C to change: a
ny key to display";:Q=GET
 1790 PRINTTAB(0,30)SPC38;
 1800 IF Q<>67 PROCif
 1810 ENDPROC
 1820 :
 1830 DEF PROCif:VDU5:YA=228
 1840 FOR CL=1 TO 2
 1850 IF K$="BQ" OR K$="BS" PROCbar:ENDP
ROC
 1860 IF K$="SH" MOVE XX,YY+16:PRINT SH$
 1870 IF K$="NA" MOVE XX,YY+16:PRINT NA$
 1880 IF K$="FL" MOVE XX,YY+16:PRINT FL$
 1890 IF K$="RW" MOVE XX+2*SP,YA:PRINT R
W$:XX=XX+2*SP
 1900 IF K$="RM" MOVE XX,YA:PRINT RM$
 1910 IF K$="RC" MOVE XX,YA:PRINT RC$
 1920 IF K$="RQ" MOVE XX,YA:PRINT RQ$
 1930 IF K$="RS" MOVE XX,YA:PRINT RS$
 1940 IF K$="BL" MOVE XX+2*SP,YA+32:DRAW
 XX+2*SP,YA-96
 1950 IF K$="WW" MOVE XX+4*SP,YY:PRINT W
W$:LS=XX+3*SP:LF=XX+11*SP
 1960 IF (KR$="QU" OR KR$="QD" OR KR$="S
U" OR KR$="SD") XX=XX+5*SP:KR$=CHR$32
 1970 IF K$="MU" MOVE XX+2*SP,YY:PRINT W
W$:MOVE XX+7.5*SP,YY:DRAW XX+7.5*SP,YY+8
4:LS=XX+SP:LF=XX+9*SP
 1980 IF K$="DT" MOVE XX-8,YY+16:PRINT D
T$
 1990 IF K$="MD" MOVE XX+2*SP,YY:PRINTWW
$:MOVE XX+2*SP,YY:DRAW XX+2*SP,YY-104:LS
=XX+SP:LF=XX+9*SP
 2000 IF K$="CU" AND CL=1 XST2=XST1:YST2
=YST1:XST1=XX+5*SP:YST1=YY+78
 2010 IF K$="CU" MOVE XX,YY:PRINT BB$:MO
VE XX+5*SP,YY-20:DRAW XX+5*SP,YY+78:LS=X
X-SP:LF=XX+6*SP
 2020 IF K$="CD" AND CL=1 XST2=XST1:YST2
=YST1:XST1=XX:YST1=YY-102
 2030 IF K$="CD" MOVE XX,YY:PRINT BB$:MO
VE XX,YY-24:DRAW XX,YY-102:LS=XX-SP:LF=X
X+6*SP
 2040 IF K$="QU" OR K$="SU" MOVE XX,YY:P
RINT BB$:MOVE XX+5*SP,YY-20:DRAW XX+5*SP
,YY+78:DRAW XX+8.5*SP,YY+50:IF K$="SU" M
OVE XX+6*SP,YY+50:DRAW XX+8.5*SP,YY+30
 2050 IF K$="QU" OR K$="SU" LS=XX-SP:LF=
XX+6*SP
 2060 IF K$="QD" OR K$="SD" MOVE XX+3*SP
,YY:PRINT BB$:MOVE XX+3*SP,YY-24:DRAW XX
+3*SP,YY-102:DRAW XX+6.5*SP,YY-74:IF K$=
"SD" MOVE XX+3*SP,YY-82:DRAW XX+6.5*SP,Y
Y-54:LS=XX+2*SP
 2070 IF K$="QD" OR K$="SD" LS=XX+2*SP:L
F=XX+9*SP
 2080 YY=YY+STNo*STS+16*UD
 2090 YA=YA+STNo*STS
 2100 NEXT CL
 2110 IF YP<0 OR YP>10 OR (YP+UD)<0 OR (
YP+UD)>10 PROCleger
 2120 IF K$="CU" XX=XX+1.5*SP
 2130 IF K$="WW" XX=XX+9*SP
 2140 IF K$="MU" OR K$="MD" XX=XX+4*SP
 2150 IF K$="BL" XX=XX+5*SP ELSE XX=XX+7
*SP
 2160 KR$=K$:ENDPROC
 2170 :
 2180 DEF PROCleger
 2190 JL=STNo*STS
 2200 YP2=YP+UD
 2210 IF YP>10 MOVE LS,292:DRAW LF,292:I
F YP>12 MOVE LS,324:DRAW LF,324:IF YP>14
 MOVE LS,356:DRAW LF,356
 2220 IF YP<0 MOVE LS,100:DRAW LF,100:IF
 YP<-2 MOVE LS,68:DRAW LF,68
 2230 YP2=YP+UD
 2240 IF YP2>10 MOVE LS,292+JL:DRAW LF,2
92+JL:IF YP2>12 MOVE LS,324+JL:DRAW LF,3
24+JL:IF YP2>14 MOVE LS,356+JL:DRAW LF,3
56+JL
 2250 IF YP2<0 MOVE LS,100+JL:DRAW LF,10
0+JL:IF YP2<-2 MOVE LS,68+JL:DRAW LF,68+
JL
 2260 ENDPROC
 2270 :
 2280 DEF PROCchst
 2290 GCOL0,0
 2300 IF STNo=1 THEN STNo=2 ELSE STNo=1
 2310 Ypos=704*STNo-352
 2320 VDU24,0;0;1279;Ypos;:CLG:VDU26
 2330 JU=STNo*STS+16*UD
 2340 GCOL0,1:XX=0
 2350 PROCstave(STNo)
 2360 ENDPROC
 2370 :
 2380 DEF PROCbar
 2390 IF XST2=0 ENDPROC
 2400 MOVE XST1,YST1:DRAW XST2,YST2
 2410 MOVE XST1,YST1+JU:DRAW XST2,YST2+J
U
 2420 IF K$="BQ" GOTO 2450
 2430 MOVE XST1,YST1+12:DRAW XST2,YST2+1
2
 2440 MOVE XST1,YST1+12+JU:DRAW XST2,YST
2+12+JU
 2450 XST1=0:YST1=0:XST2=0:YST2=0
 2460 ENDPROC
 2470 :
 2480 DEF PROCprint
 2490 VDU2:*GDUMP 0 1 2 1
 2500 ENDPROC
 2510 :
 2520 DEF PROCerase
 2530 GCOL0,0:MOVE XX,340:DRAW XX-SP,340
:PLOT85,XX,0:PLOT85,XX-SP,0
 2540 GCOL0,0:MOVE XX,340+JU:DRAW XX-SP,
340+JU:PLOT85,XX,JU:PLOT85,XX-SP,JU
 2550 GCOL0,1:XX=XX-SP:PROCstave(2)
 2560 ENDPROC
```



THE BEEBUG SPRITE EDITOR

Barry devotes this month's Visuals to a multi-mode giant sprite editor. Next month we will feature a set of compatible sprite-moving routines.

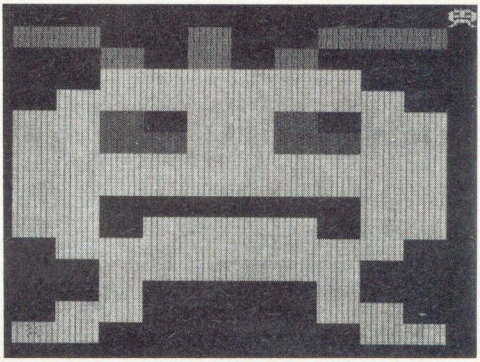
The sprite editor featured here allows the creation and editing of sprites in any of the graphics modes (i.e. modes 0, 1, 2, 4 and 5). It is totally flexible in terms of the size of sprite created, handling sprites from one pixel in size up to around half of a full screen across. Next month we will look at a set of portable machine code routines which can move such sprites around the screen. These will enable you to use sprites created with the editor in all manner of applications from games to serious software.

If you have not used sprites before, you may have thought of them as a kind of special user-defined character similar to those produced with the Beeb's VDU23 character define call. This is true to some extent, but one of the exciting properties of sprites is that they are not restricted to two colours as are the VDU23 generated characters. The number of colours in a sprite is limited only by the screen mode in which it is used. Sprites created for use in mode 2 may contain up to 16 different colours - including of course the 7 flashing pairs. These can be used to give monsters flashing eyes, or in special cases can be used to enhance animation effects by creating the illusion of moving arms, legs or wings.

THE SPRITE EDITOR

But enough of this, and on to the editor itself. First of all type in the program, and save it away before running it. As you can see from line 100, the program must be run with PAGE at or below &1400. But all this is handled automatically. When the program is run, it first asks whether you wish to begin by loading a sprite. If you answer in the negative, it will request a mode. You should enter here the screen mode in which your sprite will be used. It then requests the horizontal and vertical pixel dimensions of the sprite to be created. You may enter any number up to the limit prompted for, though you should be warned that when working on very large sprites, re-drawing operations after a save, load or flip operation are quite slow. To give some idea, an average "monster" sprite might be 10 to 20 pixels in height and width. Different screen modes however, result in differing numbers of pixels per inch, and in differing aspect ratios of the completed sprite, and some experimentation will be necessary.

Once the pixel data is entered, you move to the main editing screen. This has two areas. The bulk of the screen is occupied by the drawing area, while at the top right corner your sprite appears in its actual size. The only thing on screen at the start is a cross indicating the current pixel position. This is moved around the screen with the cursor keys, and on the Master or Compact pressing Shift will increase its rate of travel. To write a pixel, press the Tab key. The colour which is written is determined by function keys f0 to f7 in their normal or shifted form, as may be seen from the accompanying keystrip (see page 40).



THE CTRL-KEYS

Six additional functions are provided by the function keys in conjunction with the Ctrl key. Ctrl-f4 and Ctrl-f6 will flip the sprite in the X and Y direction respectively. Ctrl-f8 will clear

the screen but leave the mode and sprite size the same, while Ctrl-f9 will quit the program. In order to avoid accidents the Escape key is disabled once the Edit screen appears.

A sprite editor would not be much use without a facility to load and save sprites, and this is activated by Ctrl-f0 and Ctrl-f2 respectively. In either case the user is prompted for a filename. DFS users should note that although the program allows names of up to 10 characters in length, the DFS will only accept names of up to 7 characters (or 9 if a directory name is included). If a filing system error is generated during a load or save operation, this is trapped, and the user is informed of the error. Errors involving a filename which is not found give rise to a "Channel" error.

Finally, as you will appreciate when you come to re-load a sprite back into the editor, each sprite file contains information not only on the size of the sprite, but on the mode in which it was created. The editor uses this to reset the screen mode before a newly-loaded sprite is drawn.

Note: this program is too long to work correctly on a model B with ADFS fitted.

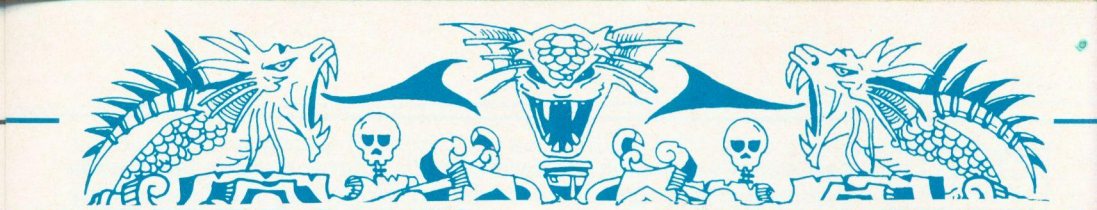
```
   10 REM Program  Sprite Editor
   20 REM Version  B 0.5B
   30 REM Author   Barry Christie
   40 REM Beebug   Jan/Feb 1988
   50 REM Program  subject to copyright
   60 :
  100 IF PAGE>&1400 THEN PAGE=&1400:CHAI
N"SprEdB"
  110 ON ERROR GOTO 3090
  120 MODE0:VDU22,7:PROCsetup1:begin=TRU
E
  130 load=(FNheading="Y")
  140 IF load THEN PROCload
  150 IF NOT load THEN PROCsetup2:VDU22,
mode%:PROCoff:PROCcursor
  160 begin=FALSE:*FX200,1
  170 :
  180 REPEAT
  190 funckey=INKEY(0)
  200 IF INKEY(-26) THEN PROCmove(-1,0)
  210 IF INKEY(-122) THEN PROCmove(1,0)
  220 IF INKEY(-42) THEN PROCmove(0,-1)
  230 IF INKEY(-58) THEN PROCmove(0,1)
  240 IF INKEY(-97) THEN PROCspritedraw
  250 IF funckey=220 THEN PROCload
  260 IF funckey=222 THEN PROCspritesave
  270 IF funckey=224 THEN PROCflipX
  280 IF funckey=226 THEN PROCflipY
  290 IF funckey=228 THEN CLS:PROCcursor
  300 IF funckey>209 AND funckey<218 THE
N curC%=funckey-202
  310 IF funckey>199 AND funckey<208 THE
N curC%=funckey-200
  320 UNTIL funckey=229
  330 PROCresetfx:MODE 7:END
  340 :
 1000 DEFPROCload
 1010 PROCspritefilename("Load")
 1020 sprite=OPENIN(filename$)
 1030 mode%=BGET#sprite:Xwidth%=BGET#spr
ite:Ywidth%=BGET#sprite
 1040 mode%=mode% AND &07
 1050 VDU22,mode%:PROCoff
 1060 PROCpixelsperbyte:PROCbytesspriteu
ses
 1070 PROCcalculatedetails:PROCloadsave(
FALSE)
 1080 PROCreplot
 1090 ENDPROC
 1100 :
 1110 DEF PROCspritesave
 1120 PROCspritefilename("Save")
 1130 sprite=OPENOUT(filename$)
 1140 BPUT#sprite,mode%:BPUT#sprite,Xwid
th%:BPUT#sprite,Ywidth%
 1150 PROCloadsave(TRUE)
 1160 PROCreplot:ENDPROC
 1170 :
 1180 DEF PROCspritefilename(name$)
 1190 PROCon:VDU28,0,31,19,30
 1200 REPEAT
 1210 CLS:*FX15,0
 1220 PRINTname$;
 1230 INPUT " name? "filename$
 1240 UNTIL LEN(filename$)>0 AND LEN(fil
ename$)<11
 1250 VDU26:PROCoff:ENDPROC
 1260 :
 1270 DEF PROCloadsave(saving)
 1280 dataaddr%=&3000+(mode% DIV 4)*&280
0+bytesline%-8*bytesonrow%
 1290 FOR Y%=0 TO maxY%
 1300 data%=dataaddr%+Y% MOD 8+(Y% DIV 8
)*bytesline%
 1310 FOR X%=0 TO bytesonrow%-1
 1320 putget%=data%+X%*8
 1330 IF saving THEN BPUT#sprite,?putget
% ELSE ?putget%=BGET#sprite
 1340 NEXT:NEXT:CLOSE#sprite
```

```
 1350 ENDPROC
 1360 :
 1370 DEF PROCreplot
 1380 FOR X%=0 TO maxX%
 1390 FOR Y%=0 TO maxY%
 1400 PROCblock(X%,Y%,POINT(chrX%+dotsiz
eX%*X%,chrY%+dotsizeY%*Y%))
 1410 NEXT:NEXT
 1420 PROCcursor:SOUND 1,-15,50,2
 1430 ENDPROC
 1440 :
 1450 DEF PROCflipX
 1460 FOR X%=0 TO maxX% DIV 2
 1470 FOR Y%=0 TO maxY%
 1480 pointX1%=chrX%+dotsizeX%*X%
 1490 pointY1%=chrY%+dotsizeY%*Y%
 1500 pointX2%=chrX%+dotsizeX%*(maxX%-X%
)
 1510 pointY2%=pointY1%
 1520 col1%=POINT(pointX1%,pointY1%)
 1530 col2%=POINT(pointX2%,pointY2%)
 1540 PROCblock(maxX%-X%,Y%,col1%)
 1550 PROCpoint(pointX2%,pointY2%,col1%)
 1560 PROCblock(X%,Y%,col2%)
 1570 PROCpoint(pointX1%,pointY1%,col2%)
 1580 NEXT:NEXT:PROCcursor:SOUND 1,-15,5
0,2
 1590 ENDPROC
 1600 :
 1610 DEF PROCflipY
 1620 FOR Y%=0 TO maxY% DIV 2
 1630 FOR X%=0 TO maxX%
 1640 pointY1%=chrY%+dotsizeY%*Y%
 1650 pointX1%=chrX%+dotsizeX%*X%
 1660 pointY2%=chrY%+dotsizeY%*(maxY%-Y%
):pointX2%=pointX1%
 1670 col1%=POINT(pointX1%,pointY1%)
 1680 col2%=POINT(pointX2%,pointY2%)
 1690 PROCblock(X%,maxY%-Y%,col1%)
 1700 PROCpoint(pointX2%,pointY2%,col1%)
 1710 PROCblock(X%,Y%,col2%)
 1720 PROCpoint(pointX1%,pointY1%,col2%)
 1730 NEXT:NEXT:PROCcursor:SOUND 1,-15,5
0,2
 1740 ENDPROC
 1750 :
 1760 DEF PROCmove(moveX%,moveY%)
 1770 PROCcursor
 1780 curX%=(curX%+moveX%+maxX%+1) MOD (
maxX%+1)
 1790 curY%=(curY%+moveY%+maxY%+1) MOD (
maxY%+1)
 1800 PROCcursor
 1810 IF M AND NOT INKEY(-1) THEN TIME=0
:REPEAT UNTIL TIME>5
```

```
 1820 ENDPROC
 1830 :
 1840 DEF PROCcursor
 1850 GCOL 3,7
 1860 cursorX%=stepX%*curX%+stepX%/2
 1870 cursorY%=stepY%*curY%+stepY%/2
 1880 PLOT 4,cursorX%-16,cursorY%-16
 1890 PLOT 5,cursorX%+16,cursorY%+18
 1900 PLOT 4,cursorX%-16,cursorY%+18
 1910 PLOT 5,cursorX%+16,cursorY%-16
 1920 ENDPROC
 1930 :
 1940 DEF PROCspritedraw
 1950 PROCcursor:PROCblock(curX%,curY%,c
urC%):PROCcursor
 1960 PROCpoint(chrX%+dotsizeX%*curX%,ch
rY%+dotsizeY%*curY%,curC%)
 1970 ENDPROC
 1980 :
 1990 DEF PROCpoint(pX%,pY%,pC%)
 2000 GCOL 0,pC%:PLOT 69,pX%,pY%
 2010 ENDPROC
 2020 :
 2030 DEF PROCblock(bX%,bY%,bC%)
 2040 blockX1%=bX%*stepX%:blockX2%=block
X1%+stepX%-dotsizeX%
 2050 blockY1%=bY%*stepY%:blockY2%=block
Y1%+stepY%-dotsizeY%
 2060 GCOL 0,128+bC%:VDU24,blockX1%;bloc
kY1%;blockX2%;blockY2%;:CLG:VDU26
 2070 ENDPROC
 2080 :
 2090 DEF PROCsetup1
 2100 Q=INKEY(-256):IF Q=253 OR Q=245 TH
EN M=TRUE ELSE M=FALSE
 2110 *FX 4,1
 2120 *FX 225,200
 2130 *FX 226,210
 2140 *FX 227,220
 2150 ENDPROC
 2160 :
 2170 DEF PROCsetup2
 2180 REPEAT
 2190 PROCenterspritemode
 2200 Xwidth%=FNenterspritewidth("X",upt
oX%,5)
 2210 Ywidth%=FNenterspritewidth("Y",upt
oY%,8)
 2220 PROCbytesspriteuses
 2230 PRINT TAB( 2,13)"Total bytes used
for sprite = ";bytesused%
 2240 PROCareyousure
 2250 UNTIL sure%
 2260 PROCcalculatedetails
 2270 ENDPROC
```

```
 2280 :
 2290 DEF PROCenterspritemode
 2300 CLS:PRINT TAB(2,1)"Please enter mo
de 01245 ... ";
 2310 REPEAT:mode$=GET$:UNTIL INSTR("012
45",mode$)
 2320 PRINT"mode ";mode$:mode%=VAL(mode$
)
 2330 PROCpixelsperbyte
 2340 PRINT TAB(4,2);bytepixel%;" pixels
 per byte in this mode"
 2350 ENDPROC
 2360 :
 2370 DEF FNenterspritewidth(which$,maxi
mum%,taby%)
 2380 PRINT TAB(4,taby%)"Enter ";which$;
" width of sprite ..."
 2390 PRINT TAB(6,taby%+1)"(max. width "
;maximum%;")"
 2400 REPEAT
 2410 PRINT TAB(31,taby%)SPC4:INPUT TAB(
31,taby%);width%
 2420 UNTIL width%<=maximum%
 2430 PRINT TAB(31,taby%);" ";width%
 2440 =width%
 2450 :
 2460 DEF PROCpixelsperbyte
 2470 dotsizeX%=2^((mode% MOD 3)+1):dots
izeY%=4
 2480 bytepixel%=2^(3-(mode% MOD 4))
 2490 uptoX%=640/dotsizeX%:uptoY%=128
 2500 ENDPROC
 2510 :
 2520 DEF PROCbytesspriteuses
 2530 bytesused%=(Xwidth% DIV bytepixel%
)*Ywidth%
 2540 IF Xwidth% MOD bytepixel% THEN byt
esused%=bytesused%+Ywidth%
 2550 ENDPROC
 2560 :
 2570 DEF PROCcalculatedetails
 2580 bytesonrow%=bytesused%/Ywidth%
 2590 bytesline%=(2-(mode% DIV 4))*&140
 2600 curX%=0:maxX%=Xwidth%-1
 2610 curY%=0:maxY%=Ywidth%-1
 2620 chrX%=1280-dotsizeX%*bytesonrow%*b
ytepixel%
 2630 chrY%=1024-dotsizeY%*Ywidth%
 2640 stepX%=((chrX% DIV Xwidth%) DIV do
tsizeX%)*dotsizeX%
 2650 stepY%=((chrY% DIV Ywidth%) DIV do
tsizeY%)*dotsizeY%
 2660 curC%=7
 2670 ENDPROC
```

```
 2680 :
 2690 DEF PROCareyousure
 2700 PRINT TAB(2,15)"Do you wish to con
tinue Y/N? ";
 2710 REPEAT:sure$=GET$:UNTIL INSTR("YNy
n",sure$)
 2720 sure%=-INSTR("YNyn",sure$) MOD 2
 2730 ENDPROC
 2740 :
 2750 DEF FNheading
 2760 title$=CHR$129+CHR$157+CHR$135+CHR
$141
 2770 title1$=title$+"Multi-Mode Sprite
Editor/Definer   "+CHR$156
 2780 title2$=title$+"By B. Christie - (
C) BEEBUG 1988   "+CHR$156
 2790 dashs$=" "+STRING$(38,"=")+" "
 2800 PRINTTAB(0,21)dashs$;title2$;title
2$;dashs$:VDU31,0,0,11,11
 2810 PRINTTAB(0,0)dashs$;title1$;title1
$;dashs$
 2820 VDU28,1,20,39,4
 2830 PRINTTAB(2,1)"Load sprite ? ";
 2840 =GET$
 2850 :
 2860 DEF PROCresetfx
 2870 *FX 225,1
 2880 *FX 226,1
 2890 *FX 227,1
 2900 *FX 4,0
 2910 *FX200
 2920 ENDPROC
 2930 :
 2940 DEF PROCoff
 2950 VDU23,1,0;0;0;0;
 2960 ENDPROC
 2970 :
 2980 DEF PROCon
 2990 VDU23,1,1;0;0;0;
 3000 ENDPROC
 3010 :
 3020 DEF PROCfileerror
 3030 PROCon:VDU28,0,31,19,30
 3040 REPORT:SOUND1,-15,200,5
 3050 dummy=INKEY(500)
 3060 PROCreplot:PROCoff
 3070 ENDPROC
 3080 :
 3090 REM <ERROR> routine
 3100 IF ERR>127 AND NOT begin THEN PROC
fileerror:GOTO 180
 3110 PROCresetfx:MODE 7
 3120 REPORT:PRINT" at line "ERL
 3130 END
```



ADVENTURE GAMES by Mitch ADVENTURE GAME

I remember the day that I received my first adventure game. Rushing home, I sat waiting for what seemed an age until the cassette slowly unwound its mystery into the depths of my machine. The fun and fascination which I found down that particular rabbit hole was responsible for the addiction that I have lived with ever since. The game was "Philosopher's Quest" written by Peter Killworth, and this game plus its successors have filled many hours of my spare time. Peter also went on to write what is my favourite adventure manual, 'Writing Adventure Games' published by Penguin. If you have not read this I can thoroughly recommend it as a blueprint for all would be adventure writers.

Never having met Peter, I was interested to see that he was a special guest speaker at the 2nd National Adventure Convention in Sutton last November. With the dragon in tow, I spent the day touring the exhibition hall, chatting with enthusiasts and being introduced to the wonders of MUD and MUGs. MUD is a Multi-User Dungeon and MUGs are of course Multi-User Games, which are generally played on some distant mini-computer via a home telephone and modem. There is even a company which will sell you a complete setup, comprising a BBC Master and the software, which will enable you to set up your own MUG which can be accessed by your friends.

Among the stands I found a new company called Topologika which, it turns out, is the new stamping ground of the old master Peter Killworth himself. In addition to marketing his former Acornsoft games the company will also be producing four new games early in the year. Among the new releases will be 'Return to Doom' which is a sequel to the science-fiction game 'Countdown to Doom'. The newly re-released versions of 'Philosopher's Quest' and 'Countdown to Doom' have been substantially revamped and are now available on disc.

PHILOSOPHER'S QUEST

This classic game involves you in a search through a cave system which contains such locations as an underground sea, a solicitor's office and an old lady's bungalow. Older hands amongst you, who have already played the original, will be surprised at the number of additional features which have been shoe-horned into this version. Peter has often used a particular problem involving a see-saw arrangement as an example of a cunning adventure puzzle (see the afore-mentioned book). It would appear that this puzzle has finally emerged into the light of day on the sea-shore of this game. The game is described as 'difficult' by the packaging and this may be true. However, it contains many puzzles which will intrigue as well as baffle. To assist the completely befuddled the game has a built-in help facility which will prompt you along the right lines towards the final solution. Remembering its antiquity, the game has a reasonable understanding of vocabulary and although its age does show in places I would recommend it to anyone.

COUNTDOWN TO DOOM

Having crash landed on the planet of Doom, you must scavenge the wrecked hulks of previous adventurer's ships for the necessary parts to repair your vessel and escape. Time and the acid atmosphere combine to form the main enemy in this game, and unless you are quick, it will destroy the remainder of your ship before you can blast off with a hold full of treasure recovered from the caves beneath the planet's surface. The required spare parts include Dilithium Crystals and shades of 'Scott me up Beamie'! The obligatory maze of tunnels beneath the planet's surface are enough to drive most adventurers mad, and this problem is compounded by a robot guardian who persistently hampers your attempts to take your ill-gotten gains along with you.

I know Peter regards this game as his favourite creation, and like its compatriots it has quite a few high spots. Personally, I hate the very existence of all mazes and this marred it for me. One point in favour of the Killworth mazes is that there is generally a logical way of solving the beasts other than by the tiresome method of 'Drop and Map'. Players of his other triumph 'Castle of Riddles' will remember the many clever techniques used to solve those particular mazes.

Both the above games retail at £9.95 from Topologika, P.O. Box 39, Stilton, Peterborough PE7 3RL, tel. (0733) 244682.

BEEBUG WORKSHOP Thanks For The Memory

Last month's workshop dealt with handling large numeric tables when memory space is tight. This month Dec McSweeney tackles two other areas where economies are possible.

STORING LITERALS

Literals, or string constants, are to be found in most Basic programs. Screen titles and page headings are typical uses in statements such as:

```
10 PRINTTAB(2,4)"Swindon Town FC"
```

or

```
120 PRINTTAB(10)"Name";TAB(30);"NI No.";
```

or

```
190 INPUT"Your favourite colour", col$
```

As they stand, the above literals occupy valuable space between PAGE and HIMEM - the area of RAM available to the user. Not a critical overhead given the size of the examples, but what of "The South-West Mercia Local Authority Sports and Social Club" or "Monthly-paid staff expenses report - January 1987"? Many tabulated print headings, too, will occupy most of the 80 characters available on a line of print, and in a poorly written program nearly identical literals might appear in several places.

Various pages of RAM beneath PAGE are set aside for functions which may not be relevant to an application. Payroll programs, for example, will almost certainly not require more than 4 SOUND envelopes; disc-based systems leave the cassette buffers untouched, unless the RS423 is being used. Briefly, the following pages could be reassigned by the user if the facilities are not required:

Page	Addresses	Function
9	(&900-&9FF)	Cassette or RS423 output buffer, or envelope 55-16 and speech buffer.
10	(&A00-&AFF)	Cassette or RS423 input buffer.
11	(&B00-&BFF)	Soft (function) key buffer.
12	(&C00-&CFF)	Character definitions, characters 224-255.

So, if you use disc, and your program employs neither function keys nor user defined characters in the range 224-255, you have 1024 bytes in which to store - well, whatever you fancy!

Two words of warning: firstly, programs which use the function key buffer should issue an *FX18 before loading them with data or defining the keys (which will destroy some or all of your data). This initialises the area and stops silly things appearing on the screen if a function key is accidentally pressed. Or you could issue *FX225 which causes the function keys to produce ASCII codes.

Secondly, you may find that machine code routines such as printer dumps load into these areas. Beware!

So how does one place literals into this area? The string indirection operator ($) placed before an address, assigns a string to be held starting at that address. For example:

```
$&900="The Economical Computer Co."
```

will store T at &900, h at &901 etc, finishing with Return character (&0D) at $&91B. You should note that a Return (character &0D) is added automatically to the end of the string, so a 40-character literal occupies 41 bytes. This means that by simply specifying the start address you can perform all the normal string

functions and manipulate the string. However you are restricted to a maximum string length of 255 bytes.

Unfortunately, if you think about it, when a literal appears in a program, it actually appears twice; once at the address &900, and again in the body of the program.

There are at least three solutions to this problem.

1. Use the fact that strings stored using indirection operators retain their contents between programs in much the same way as do the resident integer variables A%-Z% and @%. Simply write a program to store your strings in pages 9-12, and then Chain in your main program.

2. Write a program that sets up your literals and *SAVEs them to disc or tape. Now Chain your main program which will include the appropriate *LOADs when required.

3. Include a routine to initialise your variables (the routine could also include a routine to move your program down in memory) at the start of the program, then automatically delete this before the program runs. Listing 1 will give you the idea. This particular method may not work if you have loaded data into page 11 (&B00-&BFF) - think about it!

Listing 1

```
 1 REM Downloader routine - resets PAGE
 2 REM and moves the program down.
 3 REM ++ Set up your strings here, eg:
 4 $&900="Superb micros (Swindon) Ltd." :
   $&927="Another boringly long string":
   REM et cetera
 5 *K.0DELETE 1,9|MRUN|M
 6 IF PAGE<&1501 GOTO 8
 7 *K.0 FORA%=0 TO(TOP-PAGE)STEP4:
       A%!&1500=A%!PAGE:NEXT|MPAGE=&1500|M
       OLD|MDELETE1,9|MRUN|M
 8 *FX138,0,128
 9 END
10 REM Your program starts here...
```

Adapt listing 1 to suit your program. If you use more than line numbers 1-9 change the DELETE statements in lines 5 and 7. If you wish to reset PAGE to a value other than &1500 change the value in lines 6 and 7.

Subsequently, literals may be referenced by their address, e.g:

```
150 PRINT TAB(2,4)$&927
```

Or for a better standard of documentation, by assigning the address to a variable:

```
 20 heading=&927
....
2230 PRINTTAB(2,4)$heading
```

Obviously, great care must be taken not to confuse literals with numeric tables. Once you start using these techniques, Basic cannot hope to trap type mismatches and so on.

CUT YOUR SCREEN DOWN TO SIZE

You may find that the techniques described do not suit your application, or you need still more space. I'll assume you've looked critically at your choice of screen mode (try mode 5 instead of mode 2 and save 10K!) But do you need the whole screen? Type this in:

```
MODE 2    <Return>
$&7C00="Ludwig van Beethoven (1770 -
1827)"    <Return>
```

You should see a colourful splodge near the bottom of the screen. Now type:

```
$&7C0B="Wolfgang Amadeus Mozart(1756 -
1783)" <Return>
```

You should see the splodge extend to the right. Now type:

```
PRINT $&7C00    <Return>
```

and the response should be:

```
Ludwig van Wolfgang Amadeus Mozart
(1756 - 1783)
```

as the second string has partly overwritten the first.

Try this last instruction again after typing CLS. You should observe a serious drawback to the technique! In fact, screen memory is completely cleared after a change of mode or a CLS, and the information will be lost forever if you overprint the area. Besides, who needs a screen with unsightly splodges everywhere? All is not lost, however...

The screen area is processed by the 6845 CRTC chip in the Beeb which you may control using

Continued on page 32



TWO DFS UTILITIES *FREE and *MAP

Having experienced some of the delights of the ADFS system, in particular *FREE and *MAP, Bernard Hill sets about providing similar facilities on the model B for DFS users.

Having switched over recently to using the ADFS, I became aware how useful the *FREE and *MAP commands are. Being able to see the space left on a disc, and the amount of compaction required, makes using any disc system that little bit easier to use. I have therefore produced two short utilities for the DFS which will produce very similar functions, called by star commands (Master series users already have the use of both commands for the DFS and ADFS). *FREE gives an indication of used space and free space in decimal, whilst *MAP indicates the location and size of the various empty spaces on the disc.

*FREE

First we will deal with the *FREE utility. Type in the program opposite, and save it to disc before running it, using a filename other than 'FREE'. You will find it useful to keep the same line numbering, as lines 1000-1440 are also required in the second utility.

Now run the program. Provided you have typed it in correctly it will assemble the code and save it to disc under the name FREE. Test it by typing *FREE (for the current drive) or *FREE 1 (for drive 1, etc.). You can rename FREE as L.FREE, and then by setting the library to drive 0 with *LIB :0.L you can still execute *FREE when your current drive is other than drive zero. In the case of utilities which access the disc directly - such as this - you are advised to test its operation first on a disc whose contents could be sacrificed if necessary (such as an unwanted backup disc).

On this month's magazine disc you will also find the source code for a version of FREE which handles Watford's DFS and DDFS.

```
 10 REM Program FREE
 20 REM Version B1.0
 30 REM Author  Bernard Hill
 40 REM BEEBUG  Jan/Feb 1988
 50 REM Program subject to copyright
 60 :
100 FOR opt=0 TO 3 STEP 3
110 P%=&900 : Q%=P%
120 zpg=&B0:buffer=&E00
130 sec0=&E00:sec1=&F00
140 [ OPT opt
150 .go
160 JSR getdrive
170 LDX #0:JSR rdsec:LDA #2:STA val+1
180 LDX #0 : STX val : STX val+2
190 SEC:LDA sec1+7 : SBC #2 : STA fsec
200 LDA sec1+6 : AND #3 : STA fsec+1
210 LDX sec1+5:BEQ finfileloop
220 .fileloop
230 JSR getnsec:CLC
240 LDA nsec : ADC val+1 : STA val+1
250 LDA nsec+1 : ADC val+2 : STA val+2
260 SEC:LDA fsec : SBC nsec : STA fsec
270 LDA fsec+1: SBC nsec+1: STA fsec+1
280 .nextfile
290 DEX:DEX:DEX:DEX:DEX:DEX:DEX:DEX
300 BNE fileloop
310 .finfileloop LDX#0
320 .lp LDA msg1,X : BEQ end1
330 JSR &FFEE : INX : BNE lp
340 .end1 JSR outdec:LDY #0
350 .lp LDA msg2,Y : BEQ end2
360 JSR &FFEE : INY : BNE lp
370 .end2 LDA #0 : STA val
380 .end2 LDA #0:STA val:LDA fsec
390 STA val+1:LDA fsec+1 : STA val+2
400 :
410 .outdec
420 LDA #0 : STA started :LDX #20
430 .ploop LDY #0
440 .aloop SEC
450 LDA val  :SBC tens  ,X:STA val
460 LDA val+1:SBC tens+1,X:STA val+1
470 LDA val+2:SBC tens+2,X:STA val+2
480 BMI noloop : INY : JMP aloop
490 .noloop CLC
500 LDA val  :ADC tens  ,X:STA val
510 LDA val+1:ADC tens+1,X:STA val+1
520 LDA val+2:ADC tens+2,X:STA val+2
530 CPY #0:BNE notz:CPX #0:BEQ notz
540 BIT started : BMI notz
550 LDA #32 : JSR &FFEE : JMP endout
560 .notz TYA : CLC : ADC #48
570 .outnow JSR &FFEE
580 LDA #&80 : STA started
590 .endout CPX #0 : BEQ enddecout
600 DEX:DEX:DEX:DEX:CPX #8:BNE ploop
610 BIT started:BMI comma
```

```
 620 LDA #32 : BNE over
 630 .comma LDA #ASC","
 640 .over JSR &FFEE : JMP ploop
 650 .enddecout  JMP &FFE7
 660 .tens
 670 EQUD 1
 680 EQUD 10
 690 EQUD 100
 700 EQUD 1000
 710 EQUD 10000
 720 EQUD 100000
 730 .msg1 EQUS "Bytes used      "+CHR$0
 740 .msg2 EQUS "Bytes free      "+CHR$0
 750 .start EQUW 0
 760 .nsec EQUW 0
 770 .fsec EQUW 0
 780 .val EQUD 0
 790 .started EQUB 0
 800 :
1000 .getdrive LDX #zpg:LDA #1:LDY #0
1010 JSR &FFDA
1020 .lp1 LDA (zpg),Y :CMP #32 :BNE got
1030 INY : BNE lp1
1040 .got CMP #13 : BNE anotherdrive
1050 LDX #pbk MOD 256:LDY #pbk DIV 256
1060 LDA #6 : JSR &FFD1
1070 LDA data+1
1080 .anotherdrive
1090 CMP #48 : BCC baddrive
1100 CMP #52 : BCS baddrive
1110 AND #3 : STA drive : RTS
1120 :
1130 .pbk
1140 EQUB 0:EQUD data:EQUD 1:EQUD 0
1150 .data EQUD 0
1160 :
1170 .baddrive
1180 EQUW 0 : EQUS "Bad drive"+CHR$0
1190 :
1200 .rdsec
1210 STX sector : LDX #pblock MOD 256
1220 LDY #pblock DIV 256
1230 LDY #pblock DIV 256 : LDA #&7F
1240 JMP &FFF1
1250 :
1260 .getnsec LDA sec1+5,X :BNE not0
1270 LDA sec1+4,X : BEQ zerolen
1280 .not0 LDA #1:STA nsec:LDA sec1+4,X
1290 LDA sec1+4,X : BNE not00: DEC nsec
1300 .not00
1310 LDA sec1+5,X:CLC:ADC nsec:STA nsec
1320 LDA sec1+6,X : AND #&30 : CLC
1330 ROR A : ROR A : ROR A : ROR A
1340 STA nsec+1 : RTS
1350 .zerolen
1360 LDA #0 : STA nsec : STA nsec+1:RTS
1370 :
1380 .pblock
1390 .drive EQUB 0 : EQUD buffer
1400 EQUW &5303
1410 .track EQUB 0
1420 .sector EQUB 0
1430 .nsecs EQUB &22
1440 .return EQUB 0
2000 ]
2010 :
2020 NEXT
2030 c$="SAVE free "+STR$~Q%+" "+STR$~P
%
2040 PRINT"*";c$
2050 OSCLI c$
2060 END
```

*MAP

The second utility implements a *MAP command which gives a list of empty spaces on the disc in terms of their location and size. It is comparable to the Watford DFS's *HELP SPACE command, but is called with a *MAP command (for the current disc) or *MAP 1 for drive 1 etc. The program listed below needs supplementing with lines 1000-1440 from the first utility, but once this is done you should save the resulting program, (with a filename other than MAP) before running it. If you have typed in the program correctly, then it will assemble the code and save it on disc under the name MAP.

Again, test your new disc utility first on a disc whose contents do not matter (or on a write-protected disc) as the program reads the disc directory, and a single typing error in the listing could cause it to WRITE OVER the disc directory! If you rename the utility as L.MAP then you can leave it on drive 0 (as with FREE) even when using other drives, provided you set the library with *LIB :0.L.

With these two short utilities, model B DFS users have some of the same facilities as their more sophisticated ADFS (and Master series) brethren. You could also program two function keys with these commands for even greater ease of use.

```
  10 REM Program MAP
  20 REM Version B1.0
  30 REM Author  Bernard Hill
  40 REM BEEBUG  Jan/Feb 1988
  50 REM Program subject to copyright
  60 :
 100 HIMEM=&7000
```

```
110 FOR opt=0 TO 3 STEP 3
120 P%=&900 : Q%=P%
130 zpg=&A8:buffer=&E00
140 start=&E00:end=&E80:sec1=&F00
150 [ OPT opt
160 .go
170 JSR getdrive
180 LDX #0 : JSR rdsec
190 LDX #0 : JSR rdsec:LDX sec1+5
200 BEQ finfileloop
210 .fileloop
220 CLC:TXA:ROR A:ROR A:TAY:DEY:DEY
230 LDA sec1+7,X : STA start,Y
240 LDA sec1+6,X:AND #3:STA start+1,Y
250 JSR getnsec:LDA start,Y:CLC
260 ADC nsec:STA end,Y:LDA start+1,Y
270 ADC nsec+1:STA end+1,Y
280 DEX:DEX:DEX:DEX:DEX:DEX:DEX:DEX
290 BNE fileloop
300 .finfileloop LDX #0
310 .lp LDA msg,X:BEQ endmsg:JSR &FFE3
320 INX : BNE lp
330 .endmsg LDA #2 : STA last
340 LDA #0 : STA last+1
350 LDA sec1+5 : CLC : ROR A : ROR A
360 TAX : DEX : DEX : BMI nd
370 .loop
380 LDA start,X : CMP last : BNE space
390 LDA start+1,X:CMP last+1:BEQ nosp
400 .space JSR spaces:LDA last+1
410 JSR outhex :LDA last : JSR outhex
420 JSR spaces :SEC : LDA start,X
430 SBC last : PHA : LDA start+1,X
440 SBC last+1 : JSR outhex
450 PLA : JSR outhex : JSR &FFE7
460 .nosp LDA end,X : STA last
470 LDA end+1,X : STA last+1
```

```
480 DEX : DEX : BPL loop
490 .nd SEC : LDA sec1+7 : SBC last
500 STA end:LDA sec1+6 : AND #3
510 SBC last+1 : STA end+1:BNE espace
520 LDA end : BEQ noespace
530 .espace
540 JSR spaces:LDA last+1 : JSR outhex
550 LDA last : JSR outhex
560 JSR spaces:LDA end+1 : JSR outhex
570 LDA end : JSR outhex:JSR &FFE7
580 .noespace RTS
590 :
600 .spaces
610 LDY #3 : LDA #32
620 .lp JSR &FFE3 : DEY : BNE lp : RTS
630 .outhex
640 PHA
650 .outhex PHA : ROR A : ROR A:ROR A
660 ROR A : JSR outhx : PLA
670 .outhx AND #&F
680 TAY : LDA hex,Y : JMP &FFE3
690 :
700 .msg EQUS "Address : Length"+CHR$1
3+CHR$0
710 .hex EQUS "0123456789ABCDEF"
720 .nsec EQUW 0
730 .last EQUW 0
2000 ]
2010 :
2020 NEXT
2030 c$="SAVE map "+STR$~Q%+" "+STR$~P%
2040 :
2050 PRINT"*";c$
2060 OSCLI c$
2070
2080 END
```

BEEBUG WORKSHOP-*(continued from page 29)*

the command:

```
VDU23,0,R,V,0,0,0,0,0,0
```

This places the value V in register R of the 6845. One register is of interest here; register 6, the "vertical displayed" register. This contains the number of lines displayed on the screen, and may be reduced, for example, to 30 in mode 2, releasing 1280 bytes!

A little experimentation will show you that this is not the whole story. To stop the released memory from being overwritten by CLS or PRINT commands, you must also redefine your text and graphics windows to exclude the area. Your newly freed space is still vulnerable to changes of mode, so beware! Try experimenting with listing 2, which provides a demonstration of using the screen memory for saving text.

Listing 2

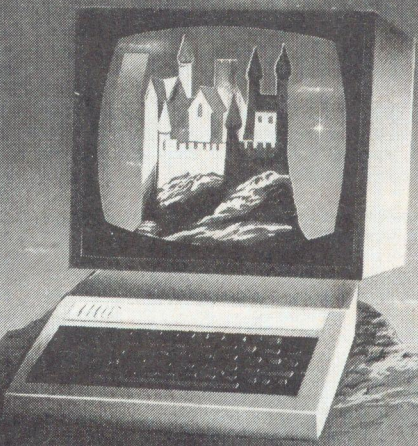
```
 10 REM Reduced screen size demo
 20 REM by Dec McSweeney
 30 :
100 MODE 1
110 VDU23,0,6,30,0,0,0,0,0,0
120 REM cut the screen by two lines
130 VDU24,0;64;1279;1023;
140 REM graphics window excludes this
    area
150 VDU28,0,29,39,0
160 REM so does the text window
170 $&7B00=STRING$(20,"Spare memory")
180 $&7C00=STRING$(13,"More spare
    memory!       ")
190 $&7D00=STRING$(25,"loads and ")
200 $&7E00=STRING$(20,"loads of it ")
210 $&7F00=STRING$(11,"How much do you
    want? "
220 CLS:CLG:REM Clears the screen
230 FOR N%=&7B00 TO &7F00 STEP &100
240 PRINT $N%
250 NEXT
```

Experience the thrills of real-life adventuring

ELECTRON & BBC MICRO USER SHOW

...at the Renold Building, UMIST, Sackville Street, Manchester

10am-6pm Friday March 18
10am-6pm Saturday March 19
10am-4pm Sunday March 20



Of course you'll find the very latest software and peripherals for the complete Acorn range at the Electron & BBC Micro User Show. But there'll also be an exciting bonus:

Your personal passport to the fascinating world of adventuring!

- ★ Take part in real-life adventures and fight sequences under the expert guidance of Britain's top role-playing specialists.
- ★ Join in a tantalising treasure hunt for the young and young at heart.
- ★ Find solutions to all those vexing puzzles at our Adventure Advice Centre.
- ★ Take home prizes of the most popular adventure games for the BBC Micro and Electron.

It all adds up to a fantastic day-out for all the family!

IT'S SO EASY TO GET THERE!

BY CAR... AA signposting and ample car parking space nearby
BY COACH... Chorlton Street Station only a short walk away
BY RAIL... 300 yards from Piccadilly railway station
...and every advance ticket comes complete with a detailed map.

Avoid the queues! Get your ticket in advance – and SAVE £1 A HEAD!

YOUR ADVANCE TICKET ORDER

Admission at door:
£3 (adults),
£2 (under 16s)

Advance ticket orders must be received by Wednesday, March 9, 1988.

Please supply:

____ Adult tickets at £2 (save £1) £____
(Order four adult tickets, get the fifth FREE!)

____ Under 16s tickets at £1 (save £1) £____
(Order four under-16s tickets, get the fifth FREE!)

☐ I enclose a cheque made payable to Database Exhibitions — Total £____

☐ Please debit my Access/Visa card no: — Expiry date: / /

☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐

Signed: _____

ELECTRON & BBC MICRO USER SHOW

Renold Building
Sackville Street
Manchester
March 18-20, 1988

Post to: Electron & BBC Micro User Show Tickets, Europa House, Adlington Park, Adlington, Macclesfield, Cheshire SK10 5NP.

Name _____

Address _____

_____ Postcode _____

PHONE ORDERS: Ring Show Hotline: 0625 879920
PRESTEL ORDERS: Key *89 then 614568383
MICROLINK ORDERS: Mailbox 72:MAG001

Please quote credit card number and full address MU2 88

A301

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 30p per word (inclusive of VAT) and these will be featured separately. Please send us all ads (personal and business) to MEMBERS' ADS, BEEBUG, Dolphin Place, Holywell Hill, St. Albans, Herts AL1 1EX. The normal copy date for receipt of all ads will be the 15th of each month.

BBC B with DFS, 40/80T disc drive and software, £350. Almost new CUB colour monitor 1431, £175. Dual 80T 800k Cumana disc drive, £165. HCR ROM box never used, £70. ADI, Megarom, Toolkit Plus, £20 each. Tel. 01-363-7114.

BBC B, dual Mitsubishi 80T drives, steel CPU and keyboard cases (plus original), hi-res green monitor, ADFS/DFS, View 3.0, Exmon 2, Masterfile II, Books, £400. Every issue of Acorn User to date, offers. Tel. (0730) 61901.

Voyager 7 modem with Commsoft ROM, including leads, as new, £70. Tel. Alton (0420) 63554.

BBC B with speech processor, DFS and 32k sideways RAM, £175. Cumana twin DS drives, own power supply, £175. Acorn Z80 2nd processor + software and manuals, £175. 65C02 2nd processor £75. Prism 1000 modem + software £45. HCR ROM/RAM board for up to 24 ROMs £60. Open logo £30. Books and software. Tel. Hornchurch (04024) 74633 eves.

DFS upgrade kit (1770) for BBC micro £30. Tandy CGP-115 printer/plotter - spare paper £35. Quinkey keyboard + adaptor and software £35. Acornsoft Forth cassette + 2 books, £10. Tel. Horley (0293) 773524 eves.

Epson printer MX80 with manual £60, working order. Acorn disc drive 40T SS with discs, little used, £20. Tel. 01-868 5166.

WANTED Teletext adaptor with ATS ROM and Viglen console for BBC B. FOR SALE: 40T 100k disc drive £50 ono. Acorn disc interface 8271 complete including DNFS £35 ono or the two for £80. Tel. Derby (0332) 556381.

Master Compact 128, boxed as new, £100+ of software discs, books, games, joysticks, £350. Tel. 01-961-2925 before 1 p.m.
BBC Master Turbo board, only £79 ono. Tel. Bedford (0234) 67067 eves.

Torch ZEP100 Z80 second processor board. Complete with useful business software. £110. Tel. 01-451-0520.

HCR external ROM box B. Includes extra card giving space for up to 24 ROMs. Also 64k battery backed RAM. Only 5 months old. £60 inclusive p&p. J.Follett, 26 Arbor Lane, Winnersh, Berks, RG11 5JD.

APTL sideways ROM board with RAM chips and write protect/read inhibit switches £20. Solidisk DFDC board with DFS/ADFS chips series 1 £20. Solidisk 128k board, barely used, with software £45. BEEBUG's Sleuth £10. Tel. (0642) 311848.

View 2.1 chip + manuals and keystrip, no box. Also View Printer Driver Generator £25. Spellcheck II chip, disc and manual, boxed, £10. C.Blore, 163 Billing Rd, Northampton, NN1 5RS.

Serial 8056 thermal printer as new £30. Tel. Northwood Middx (09274) 24377 eves.

Printer Mannesmann Tally MT85, dot matrix, Epson compatible, tractor and friction feed, £220 ono. Tel 01-952-7244.

Software, all original and hardly used. Disc:- Master Compact and Electron: Crazee Rider £9, Palace of Magic £9, Codename Droid £9; Master only: Grand Prix Construction £9, 40 Screens £6. More on tape for the B, B+ and the Master. Postage included. Tel. (0222) 865248 ask for Tony.

Acorn Archimedes A310 as new, NEC Multi-Sync monitor, £1200 or any sensible offers. Tel. 01-607- 6072.

ROMs with manuals etc: Wordwise+, Printermaster, Extended Graphics, Spellcheck 2, Spy 2 + Acorn DNFS, £15 each. InterWord unused £35. Quest mouse + Paint £40. Solidisc 2MEG board with ADFS + DFS set, £50. Tel. Upminster (04022) 23811.

Spellcheck II, boxed with dictionary disc and manual, £10. Tel. Gerald French, Stalbridge (0963) 63497.

Double sided 200K 40T disc drive, £50. Elite disk & manuals £6. Toolkit Plus ROM + manual £10. Penfriend ROM + manual £5. Tel. (091536) 2066.

Solidisk 2.12 DFS interface plus 2.12 ADFS, unused with original packing, manual and fitting instructions, £20. Tel. Eastergate 2683 eves.

Solidisc RTC £15. Watford S.W.ROM board with 16k S.W.RAM £20. Opus DD disc interface with DFS £30. Nightingale modem + Comstar II £70. Acorn 6502 2nd processor + HI WW+ & HI Basic £70. BEEBUG Spellcheck III £15. Timewarp £10. All with manuals. Tel. (02814) 5332.

BBC B issue 7, 16k sideways RAM, original games e.g. 'Elite', User and Advanced guides, excellent condition, boxed £240. Tel. (0934) 820952.

Replay 1770 and ADFS version £15, Slave utility ROM £8, Oxford Pascal ROM £8, almost new. Tel. Kings Lynn, Norfolk 771581.

NEW TECHNICAL REFERENCE GUIDE NOW AVAILABLE!
£12
Please contact us for further details.

At last a machine code development system that *really* does outperform ADE

SYSTEM APPLIED TECHNOLOGY LTD.

DEPT. G.
Fifth Floor Sheaf House
Sheaf Street
Sheffield S1 2BP
Tel. (0742) 768682

Access and Barclaycard welcome
Further information available

ADE+

Features

- Runs on BBC B, B+, Master, Compact, ADFS, DFS, ANFS, NFS
- Full use of all available RAM
- Intelligent memory management unit (MMU)
- User switches on all main features
- Automatic search for macros on disc
- Assemble absolute or relocatable code
- Source program tokenised or full text
- REPEAT..UNTIL, WHILE..WEND etc. high level language constructs
- Macros nestable to any depth
- Excellent error diagnostics
- Linker
- Editor and symbolic disassembler provided

A comparison of three 65C12 assemblers

Feature	ADE +	BBC Basic	MACROM
Number of pseudo-ops	64	4	36
Use of all available RAM	Yes	No	No
Macros	Yes	No	Yes
Relocatable output	Yes	No	No
Linker with libraries	Yes	No	No
High level constructs	Yes	Yes	No
Number of error reports	39+warnings	3+BASIC errs	20
65C00 extended opcodes	All	65C 12 only	65C 12 only
Switch of 65 C 12 opcodes	Yes	No	No
ROM size	32K	16K	16K
Disassembler	Yes	No	Yes
Label restrictions	No	Yes	Yes

ADE+

Versions and prices

Recommended for Master Compact...
ADE+ MMU and 65C00 series assembler on disc (3.5" ADFS). 32K sideways RAM required.
£42.00 + vat

Recommended for BBC B, B+ ...
ADE+ MMU and 65C00 series assembler on 2 16K EPROMS with DFS 5.25" utilities disc.
£46.00 + vat

Recommended for Master 128, turbo..
ADE+ MMU and 65C00 series assembler on EPROM cartridge with 5.25" DFS utility disc.
£49.00 + vat

Upgrade
ADE to ADE+ (upgrade to either disc, EPROM or cartridge, you must send in ADE ROM chip)
£27.00 + vat

Upgrade
ASM to ADE + (upgrade to either disc, EPROM or cartridge, you must send in ASM ROM chip)
£35.00 + vat

Please add £1.25 P&P per unit.

ADE+

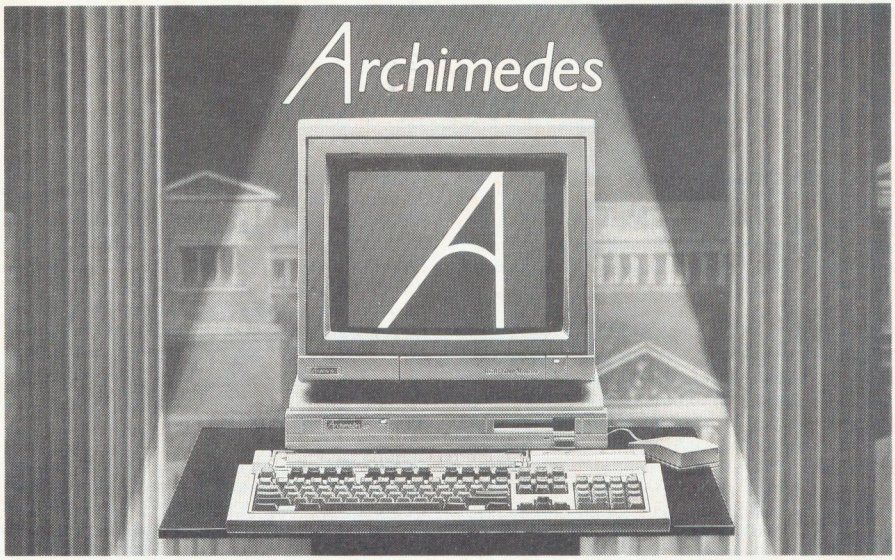
The Ultimate Assembly Language Development Tool

ADE+ is a 65C00 series assembler system supporting all the mnemonics of the 65C12 used in the latest BBC microcomputers plus the additional 'Rockwell' instructions. ADE+ is fast, faster in fact than the in-built BASIC assembler and all rival products that we have tested. The assembler produces absolute code that can be *RUN or linker modules that can be merged with the output from other programs using the ADE+ linker. ADE+ supports a powerful linker which drastically cuts assembly time; a feature normally only found on minis and mainframes. The linker will even link the output from compilers with your assembly language programs. Full library support for both the linker and the assembler is provided - fast searching for unknown instructions in a random access macro library. ADE+ is a modular system with many modules to add later; i.e. a mouse based editor & a Z80 cross assembler! A print spooling system uses sideways RAM as a print buffer to eliminate waiting time; your listing runs off as a background job! Use the print spooler from BASIC or your own programs. ADE+ uses ALL available memory. With a second processor attached the IO processor spare memory is used as a buffer to reduce the amount of disc access. All available memory is handled by ADE+'s intelligent memory management module. Use your own favourite editor or the one provided. Assemble from disc or memory. Full utilities including librarians, converter for BBC BASIC etc. ADE+ must be *the* bargain of 1988!

It operates at 4 million instructions per second.

It's the world's fastest microcomputer.

It has been voted the 1987 microcomputer of the year.†



†Home/Small Business Category of the British Microcomputing Awards. Model shown here is the Archimedes 310 with colour monitor and includes Mouse.

- At an operating speed of 4 million instructions per second just about everything you do happens instantly. With this kind of power at your fingertips the advantages are enormous.
- Already Archimedes* has won the British Microcomputer of the Year award, a clear winner against the other two finalists, the Amstrad 1512* and the Amiga 500*
- In addition to BBC Basic V, high level languages such as 'C' and 'Pascal' can be used for specialist applications fully exploiting the computer's outstanding power, graphics and speed.
- The huge potential of Archimedes doesn't stop there. The 310M with its built-in PC emulator can run Lotus 1.2.3,* D BASE III,* MS-WORD,* and other popular business programs available under MS-DOS.*
- With prices starting at £749 exc. VAT for the 305 with mono monitor and rising to £1035 exc. VAT for the 310M with colour monitor, the Archimedes 300 series represents unparalleled value for a computer system of such capability.

PRIORITY REQUEST

Please send me an information pack on Archimedes together with a list of dealers.

Name:_____

Address:_____

_____

Postcode:_____ Tel:_____

☐ Tick here for written details of the 0% finance deal available through participating dealers.

Acorn Computers Ltd., FREEPOST 4335, BS1 3YX

Acorn The choice of experience.

17

— OR PHONE 0800 100 100 —

0% (APR 0%) FINANCE FOR LIMITED PERIOD

Interest free credit is available on all Archimedes systems during the period 1.10.87 to 31.12.87. Maximum period of credit 12 months. Subject to status. Written quotation available on request from participating dealers who are licenced credit brokers.

*MS-DOS and MS-WORD are trade marks of Microsoft Corporation.
*Amstrad 1512 is a trade mark of Amstrad Consumer Electronics plc.
*Amiga 500 is a trade mark of Commodore Business Machines.
*Lotus 1.2.3 is a trade mark of Lotus Corporation.
*D BASE III is a trade mark of Ashton-Tate.
*Archimedes is a trade mark of Acorn Computers Ltd.

Business Ads

MARKS AND STATISTICS for handling students marks. Widely used in schools, colleges. Spreadsheet format, calculates totals, sorts, analyses, normalises etc. Prints lists, histograms. BBC B, Master. 40-80 track. £17. *In-form, 73 Woodfield Park, Colinton, Edinburgh EH13 0RA.*

CHESS EXPERT SYSTEM (Master/B/B+). A complete disc-based game storage and analysis system with extensive features and data links to Colossus 4 and White Knight 12. Supplied with 2000-move openings database, or optional 37000-move set. Send £1 for demonstration disc to: *Bernard Hill, Hawthorn Bank, Scott's Place, SELKIRK TD7 4DP.*

SHAREMATIC. The best software package for the keen investor, (runs on BBC Master 128). Free - automatic updating from Teletext. Free - 500 trading days of data for 170 shares. Plots, Graphs, Point & Figure, & Oscillator and more. Only £35. For brochure write to: *Citymagic Associates, Dept BB3, 40 Manor Road, Goldington, Bedford MK41 9LQ. Tel. (0234) 856050/67067.*

QDUMP ROM Loads into SWRAM. Produces large dumps of all screen modes, also fast text dumps. Indenting, Windowing and Triggering. B/ Master/ Tube. Amazing value £5 (specify tracks). SAE for wonderful examples. *David James, 29 Hollybush Road, CARDIFF CF2 6SY.*

JOURNAL INDEXING SYSTEM (JISys) for BBC (state drives and model) £35.00. JOURNAL INDEXING SYSTEM for Master 512 (written in 'C') to index up to 10,000 articles £65.00. **VIDEO INDEXING SYSTEM (VISys)** for Master 512 (written in 'C') for professional VIDEO LIBRARIES, up to 10,000 titles, with powerful print of a cross-referenced catalogue £65.00. (Sorry, not for straight BBCs!) **JOB COSTING SYSTEM (CostSys)** for MASTER 128/COMPACT (not 'B', sorry!), £65.00. All prices are exclusive to BEEBUG members. FROM: *K.A.SPENCER (Software), 74 Dovers Park, BATHFORD, Nr Bath BA1 7UE.*

OMNIROM - the ultimate Utility ROM for the MASTER and COMPACT. Adds 3 new CONFIGURATION Commands - FOREGROUND and BACKGROUND COLOURS and FONT (which integrates 3 new Fonts), ADFS/DFS Menu System, plus many other utilities, such as 65C12 Disassembler, Memory and Disc Editors, BASIC STATUS and much, much more. Supplied on 16K ROM. Price £12.95 (p&p incl.). **THE SCRAMBLER** for BBC B, MASTER & COMPACT PROTECT your files for as little as £19.95 (p&p incl.). The system to help you comply with the DATA PROTECTION ACT. ENCODES any file held on disc (ADFS/DFS). Menu driven. Unique PASSWORD. Supplied on ROM. Ideal for schools, colleges. *Number 3 Software, 3 Dairy Farm Court, Attleborough, Norfolk NR17 2BT.*

Personal Ads *(continued from page 34)*

Disc drive DS 40/80T £70. Acorn full DFS kit £35, DNFS £10, WE DDFS £30. View £25, Spellcheck III £18, Acorn printer driver generator £7, Dumpmaster £15, Discaid £10, Buffer and Backup £10, Beebfont £10, Hershey characters £10, Printmaster £15, AMX mouse (latest) + Art + Paint Pot + Desk + utilities £50, Graphpad I £25, Graphpad III £35. Modem £15. Tel (093484) 2410 eves.

WYSIWYG Plus ROM and manual in original pack, unwanted gift, £15 + p&p. Tel. (0274) 571406.

Viglen ROM Cartridge system for the Master with 4 cartridges, £10. Master Reference Manuals parts 1 & 2, £9 each. Tel. Southampton (0703) 845104.

Teac twin 40T DS drives in a housing with connecting cables, £145. Tel. Luton (0582) 570459 eves. and weekends.

EXMON II (model B) £10, Advanced User Guide £9 (plus postage). Tel. (051722) 9868 eves.

BBC model B, Cumana DD 800k disc drive, CVx16/2 ROM/RAM board, 1770 DFS/ADFS, Wordwise Plus, Spellmaster, InterSheet, InterChart, battery backed sideways RAM, 'OAK Universal' PC case, Wordease, Facit 4511 printer. Total cost £1553. Will sell for £950 ono. Tel. (061928) 6923.

HELP!. Books required for Acorn Z80 second processor (ones usually supplied with the unit). Tel. (0245) 267456.

BBC Model B, Solidisk SWR256 RAM board. ADFS/DDFS/DFDC & Real-time Clock, Viglen remote keyboard, Aries B20 RAM board, Dual 40/80 disc drives, Pace Nightingale modem & Commstar ROM, AMX Mouse & Super Art ROM, Speech!, Toolkit ROM, Printmaster ROM, White Knight 12, many games, joysticks, all leads, manuals and boxes in VGC £550. Tel. 01-743 7523.

Microline 82A dot matrix printer with parallel & serial interface, friction & tractor feed, fast (120cps/bi-directional), very reliable. Includes parallel connector and manual £140. Tel. (09363) 3300.

Master 128 (as new), Cumana Dual 80T/DS drives, monochrome monitor, 6502 2nd processor, lots of software: utilities, languages, ROMs etc, £550. Tel. Bookham (0372) 58655.

BBC B with Torch Z80 disc pack (dual 400K disc drive) and Perfect software (with manuals), good condition, £400 for quick sale. Tel. 01-845 6302.

Master Turbo 128 £390, Microvitec medium resolution monitor & swivel stand £190, Akther Dual MD400A 40/80 drives £160, Spellmaster (new) £40, pristine condition. Tel. (0295) 65262.

NEW

WORDWISE PLUS II

"The feel of InterWord yet the style of WORDWISE, making a great combination".
BEEBUG Dec 87

Preview in 80 columns
With a long document in memory, there is normally insufficient room to preview it in 80 column mode. WORDWIDE PLUS II allows you to preview your text in 80 columns, irrespective of how full memory is. A menu-driven facility permits several files to be printed out as if they were one long document.

Easy file selection
WORDWISE PLUS II makes memorising filenames, drives and directories a thing of the past. A single key press shows all the files in the current directory. Select one with the cursor key or enter it by name. Works with all properly written DFS's. E.g. Acorn DFS ADFS, Watford 62-file systems. Also Solidisk's "unlimited catalogue entry" DFS.

Multiple documents
WORDWISE PLUS II provides a text area and ten 'segments'. This feature allows up to eleven separate documents to be in memory at once. Just a single key press now enables you to select the text or any segment, from both the menu and edit mode.

Improved search and replace
Search and Replace operations are now even easier to use. Both the search and replace strings are stored for instant editing or re-use. Works directly from edit mode.

Check saved file
Disc drives are not 100% reliable. For peace of mind, WORDWISE PLUS II can check any part of your text against any file. This way you can be sure that your text has been correctly saved.

Easy to use
WORDWISE and WORDWISE PLUS received consistently excellent reviews for convenience and ease of use. WORDWISE PLUS II still retains the familiar WORDWISE PLUS environment, but offers even greater flexibility.

Programming Language
Like WORDWISE PLUS, WORDWISE PLUS II contains a powerful programming language which is very easy to use. Over 30 new keywords are provided, which greatly simplify many programming tasks.

WORDWISE PLUS II runs all programs produced for WORDWISE PLUS, but is much quicker. WORDWISE PLUS II permits any program to be "tokenised, which saves valuable memory. A tokenised program can run several times faster than the original version.

Extra CTRL keys
CTRL keys are the short-cuts in WORDWISE word processing. WORDWISE PLUS II provides over a dozen extra CTRL keys to solve common requirements. E.g. mark word at cursor, mark paragraph, delete current line, GOTO any string. Operations on marked sections of text can now be chosen from a menu.

Supplied on one chip
WORDWISE PLUS II is a complete word processor in its own right. It is 32K long (twice the length of WORDWISE PLUS) but behaves like an ordinary ROM. WORDWISE PLUS II can replace your existing word processing chip.

Compatibility
WORDWISE PLUS II is 100% compatible with existing WORDWISE and WORDWISE PLUS files. It runs WORDWISE PLUS segment programs and works with **Spell Master**. Works with all filing systems (Tape, DFS, ADFS, Network etc.)

Additional embedded command
A new embedded command is present in WORDWISE PLUS II. **PD** (Print Date) means you never have to remember the date again! (This requires a Master or Model B with Solidisk Real Time Clock to operate).

Free utilities
WORDWISE PLUS II is supplied with a disc containing several useful programs. The mail-merge routine lets you prepare a circular letter. A multiple column utility prints your text in up to 5 columns. A label printer is also included.

Printer buffer
Why wait around while your printer churns out a long document? With the printer buffer supplied your computer becomes available for use again in a fraction of the time. (NB This requires sideways RAM to work. We can supply a suitable module — see below).

Floating point maths
The WORDWISE language only allows arithmetic involving whole numbers, but virtually any calculation is possible from within WORDWISE PLUS II. For example, a WORDWISE PLUS II segment program could easily produce invoices showing totals, discounts, VAT and so on.

GUARANTEE

This is our offer with WORDWISE PLUS II. Buy it. Try it. If you don't want to keep it, return it to us for a refund!
(Available only when purchased from IFEL).

40%
Off normal price to existing WORDWISE PLUS owners
£39.95



IFEL (Interface Electronic & Computing)
36 UPLAND DRIVE, PLYMOUTH, DEVON PL6 6BD

For same day despatch, telephone your requirements to us on:

(07555) 7286

Please allow £1 for postage.

WORDWISE PLUS II	£65.95
Upgrade from **WORDWISE PLUS**	£39.95
Spell Master (£47.00 when purchased with WORDWISE PLUS II)	£47.50
16K RAM Module (No soldering to install)	£14.95

ADVERTISING IN BEEBUG

For advertising details, please contact
Yolanda Turuelo
on
(0727) 40303
or write to
Dolphin Place, Holywell Hill,
St Albans,
Herts.AL1 1EX

SPRITE EDITOR

This keystrip is designed to be used with the Sprite Editor on pages 23 to 26 of this issue. Either cut out the keystrip or photo-copy this page and place under the plastic strip on your micro when using this program.

Key f0	Key f1	Key f2	Key f3	Key f4	Key f5	Key f6	Key f7	Key f8	Key f9
LOAD		SAVE		FLIP X		FLIP Y		CLEAR SCREEN	STAR
Black/ White	Red/ Cyan	Green/ Magenta	Yellow/ Blue	Blue/ Yellow	Magenta/ Green	Cyan/ Red	White/ Black		
Black Black Black	White Red Red	Yellow Green	White Yellow	Blue	Magenta	Cyan	White		

THE MASTER PAGES

Devoted to the Master Series Computers

In this issue we have a varied selection of articles, reviews and other information for Master and Compact users.

Our main article shows how the ADFS may be used from within Basic, and provides a set of routines that may be used to build utilities such as disc menus and the like.

We have two reviews, on the new Smart Cartridge from Care Electronics, and a Real-Time Clock for the Compact.

Finally, we conclude with some more Hints and Tips for Master and Compact Users.

MASTER SERIES

Talking to the ADFS

Lee Calcraft presents some routines for accessing the ADFS, from which a variety of interesting utilities may be constructed.

If you have looked at the Advanced Reference Manual for the Master or Compact you may have come across the various operating system calls for accessing the DFS and ADFS from assembly language. They are somewhat involved, and require the use of complex parameter blocks for transferring data. In this pair of articles, I will be developing a set of Basic procedures which allow the user to obtain information from the ADFS in a relatively painless manner.

The user may then put the various pieces of information to use in whichever way he chooses. For example, it is possible to create many new ADFS commands in this way. And to illustrate the power of the procedures we will put together routines to catalogue a complete disc, to give the disc space used by the current directory, and to display a disc catalogue in a user-defined format. It is also a simple matter to create an ADFS auto-menu customised to the user's individual requirements. But first of all the procedures themselves.

DRIVE NO, DIRECTORY NAME & TITLE

The block of code to perform the task of reading the current drive number, directory name and title is given in listing 1. It derives its information from two calls to OSGBPB (Get Block Put Block). As with all of the routines presented, the procedure uses for workspace the so-called transient program area (&DD00-DEFF). This should be free at all times on both Master and Compact. Model B users with ADFS fitted will however need to change all references to this area, both direct and implied. The two pages at &900 should make a suitable alternative. Also common to all routines is a degree of DFS compatibility. All the routines presented give some kind of result on the DFS, but the way in which the directory structure is treated by the native OS calls makes DFS use cumbersome, and no attempt at full DFS compatibility has been sought in the writing of these procedures.

To test out the first procedure, type in listing 1 exactly as it appears, and add the following:

```
10 REM Procedure Tester
20 MODE 0
30 PROCtitle
40 PRINT"Drive No. ";drive
50 PRINT"Title     "title$
60 PRINT"Name      "name$
70 END
```

If you run the program, you should see the currently selected drive number, directory title and name displayed on the screen.

As you will gather, the main procedure returns these three parameters in the variables **drive, title$** and **name$.** To fully test the routine, you can move around the directory structure of your disc, and re-run the program. This particular routine should give completely accurate information when run on the DFS, though the title displayed will always be the disc title, since titles cannot be assigned to individual directories as they may be on the ADFS.

Listing 1

```
 1000 REM------------------------------
 1010 REM     LISTING 1   ref .>ADFS1ut3
 1020 REM------------------------------
 1030 REM Get drive no, dir title & name
 1040 :
 1050 DEFPROCtitle
 1060 X%=0:Y%=&DD
 1070 !&DD01=&DD10
 1080 A%=5:CALL &FFD1
 1090 title$=FNread(?&DD10)
 1100 drive=?(&DD10+len+2)
 1110 !&DD01=&DD0E
 1120 A%=6:CALL &FFD1
 1130 name$=FNread(?&DD10)
 1140 ENDPROC
 1150 :
 1160 DEFFNread(len)
 1170 text$=""
 1180 FOR A=1 TO len
 1190 text$=text$+CHR$(?(&DD10+A))
 1200 NEXT
 1210 =text$
 1220 REM------------------------------
```

READING FILE AND DIRECTORY NAMES

The second block of code, given in listing 2, reads the names of all objects in the currently selected directory, together with the total number of such items. These parameters are returned in the array **file$(n)** and the variable **fileno** respectively. The procedure again uses OSGBPB, but because the parameter block for a full directory can exceed the space available at &DD00, the call to OSGBPB is made twice for directories containing more than 24 objects. The first call reading the first 24, the second reading the remainder. The maximum number of objects in any given directory is 47 on the ADFS.

To test this procedure, (assuming that your machine already contains all the above code) type in listing 2 and add the following:

```
  20 MODE 0:DIM file$(50)
  70 PROCfile
  80 PRINT"Number of objects ";fileno
  90 FOR A=1 TO fileno STEP 4
 100 FOR B=0 TO 3
 110 PRINTTAB(5+14*B)file$(A+B);
 120 NEXT:PRINT:NEXT
 130 END
```

Note the dimensioning of the array file$ in line 20. This must always be present when PROCfile is used. When you run the program you should now see a display of the names of all objects in the current directory in a four column format.

Listing 2

```
 2000 REM------------------------------
 2010 REM     LISTING 2   ref .>ADFS2ut3
 2020 REM------------------------------
 2030 REM Get array of object names
 2040 REM in current directory
 2050 :
 2060 DEFPROCfile
 2070 N%=1
 2080 X%=0:Y%=&DD
 2090 !&DD01=&DD10
 2100 !&DD05=24
 2110 !&DD09=0
 2120 PROCreadfile
 2130 !&DD05=48
 2140 !&DD09=24:N%=N%-1:PROCreadfile
 2150 fileno=N%-2
 2160 ENDPROC
 2170 :
 2180 DEFPROCreadfile
 2190 FORA=&DD10 TO &DEF0 STEP4
 2200 !A=0
 2210 NEXT
 2220 A%=8:CALL &FFD1
 2230 Z%=0
 2240 REPEAT
 2250 len=?(Z%+&DD10)
 2260 file$(N%)=""
 2270 Z%=Z%+1
 2280 FOR A=1 TO len
 2290 file$(N%)=file$(N%)+CHR$(?(Z%+&DD1
0))
 2300 Z%=Z%+1
 2310 NEXT
 2320 N%=N%+1
 2330 UNTIL len=0
 2340 ENDPROC
 2350 REM------------------------------
```

To turn this display into an automatic disc menu requires only that a letter or number be displayed alongside each name, and that the keyboard be checked for a corresponding entry.

At present however, we have no way of distinguishing a file name from a directory name. So if the user selects a directory name using such a menu, the routine would be unaware that it should not be chained in as a Basic program, and errors would of course occur. To deduce whether each object in the current directory is a file or sub-directory, another call must be made in which each name is offered in turn for identification. This is accomplished in the third procedure block.

READ INFO ON EACH OBJECT

The third block of code, given in listing 3, builds on the information obtained by PROCfile in listing 2, and extracts information on each object in the current directory. Before this third block of procedures is called, the user must dimension a new array **file%(n,m)** of size 50x4 to accommodate the information read by this procedure. In fact the procedure PROCcatinfo uses the parameter full to determine the quantity of information retrieved. If **full** is set to FALSE, then only the status of the object is stored. If the object is a file, then file%(n,0) will be set to 1, whereas if it is a directory it will be set to 2. If the procedure is called with **full** set to TRUE, then information on each object's load address, execution address, length and file attributes will be stored in **file%(n,1)** to **file%(n,4)** respectively. The variable n used here takes any value between 1 and 47, and simply indicates the position of the object in the total list of objects in the current directory. It corresponds to the use of n in the assignment **file$(n)** used by PROCfile.

There are many uses for the file information read by this procedure, such as calculating the total disc space used in a given directory. But to get it working, we can use a simpler application. First of all, with listings 1 and 2 and all their associated calling code already in memory, type in listing 3. Then add the following lines:

```
  20 MODE 0:DIM file$(50),file%(50,4)
  75 PROCcatinfo(FALSE)
 105 PRINTTAB(5+14*B);
 110 IF file%(A+B,0)=2THEN COLOUR0:COL
OUR129
 114 PRINTfile$(A+B);
 116 COLOUR7:COLOUR128
```

This will cause all directories in the display of current objects to be listed in reversed out text. It does this by calling PROCcatinfo, and simply reading the result in **file%(n,0)** to determine the colour of the display.

Listing 3

```
 3000 REM-------------------------------
 3010 REM     LISTING 3    ref .>ADFS3ut3
 3020 REM-------------------------------
 3030 REM Get Catalogue info for all obj
ects
 3040 REM in current directory
 3050 :
 3060 REM  Note PROCfile must have been
called
 3070 DEFPROCcatinfo(full)
 3080 PROCsetup
 3090 FOR A=1 TO fileno
 3100 file%(A,0)=FNinfo(file$(A),full)
 3110 IF full THEN file%(A,1)=load:file%
(A,2)=exec:file%(A,3)=length:file%(A,4)=
attrib
 3120 NEXT
 3130 ENDPROC
 3140 :
 3150 DEFPROCsetup
 3160 X%=0:Y%=&DD
 3170 A%=5
 3180 !&DD00=&DD20
 3190 ENDPROC
 3200 :
 3210 REM Get catalogue info on single o
bject
 3220 REM Must call PROCsetup first
 3230 DEFFNinfo(fname$,full)
 3240 $&DD20=fname$
 3250 U%=USR(&FFDD)
 3260 IF full THEN load=!&DD02:exec=!&DD
06:length=!&DD0A:attrib=?&DD0E
 3270 U%=U% AND &FF
 3280 IF U%=0 VDU7
 3290 =U%
 3300 REM-------------------------------
```

As an additional example of the use of the object-type information held in the array **file%(n,m)**, try adding the following lines to the program developed so far:

```
 130:
 140 PRINT'"Directories :"
 150 PROCcatinfo(FALSE)
 160 FOR A=1 TO fileno
 170 IF file%(A,0)=2 PRINTTAB(5)file$(A
)
 180 NEXT
 190 END
```

Continued on page 46



Peter Rochford takes a look at the latest ROM cartridge for the Master, one which seems a good deal smarter than most.

Product	Master Smart Cartridge
Supplier	Care Electronics 800 St Albans Road, Garston, Watford WD2 6NL.
Price	£39.95 + p & p

How often have you been playing a computer game and had to abandon it due to a pressing engagement? The opportunity to 'freeze' a program at any instant would provide an ideal solution to such frustrations. Or maybe some other interruption takes you away from the machine longer than a few minutes, and just as you had cranked up your best score yet, and a host of extra lives too! Wouldn't it be nice to have the facility to save the game away on disc or tape, and then reload it later and continue exactly where you left off?

Well, Master owners can now do just that with Care Electronics' Smart Cartridge, consisting of a holder ready fitted with Care Electronics' EPROM. This 'black box' plugs into one of the Master's cartridge ports and enables you to perform this handy trick and quite a few others too.

On top of the cartridge is a small red button, and when pressed a number of different tasks can be initiated. These tasks are chosen from an on-screen menu and relevant details of the tasks chosen are then stored in the cartridge's own RAM.

You can choose to save the contents of the Master's normal screen RAM, the Shadow RAM or the entire RAM, either straight to disc or to banks 4 & 5 of sideways RAM for subsequent saving to disc. Alternatively, you could opt to temporarily freeze the current program, send a screen dump to an Epson-compatible printer, call a Help file, poke a byte into any memory address, call a machine code program resident in memory or execute an operating system star command.

The ability to issue an OS command from within a program is very exciting. For example, you could exit from a program, examine memory by calling a machine code monitor such as BEEBUG's Exmon and then alter the contents of an address and return to the program.

This feature of the cartridge is extremely powerful, but there are also potential dangers and abuses. Not least of these is that it makes the pirating of software much easier. Care Electronics acknowledge this with a statement in their literature condemning this practice, which is illegal.

THE SMART CARTRIDGE IN USE

In the main, using the cartridge should be straightforward. I say should be, as the documentation supplied with it is really awful and leaves you much to work out for yourself. In particular, two features of the cartridge that appear on the menu screen are not documented at all (and calls to both suppliers and manufacturers have so far failed to determine what these extra functions do). Once you have got to grips with the various features of the cartridge though, all works well. The saving and reloading of programs mostly worked fine in practice, but there were a few games I tried that would not work properly on reloading.

CONCLUSION

This is a most intriguing and powerful piece of kit. It has so many possibilities that many Master owners I am sure would find it a great asset in one way or another.

I would suggest, however, that Care Electronics consider lowering the price of the Smart Cartridge. It is, I feel, rather expensive at present. Also, they should most certainly supply decent documentation. The four sheets of computer printout supplied at present are badly written and quite inadequate. B



One of the more obvious omissions on the Compact is a real-time clock. PMS have tackled this to produce a really smart solution. Lance Allison reports.

One of the features supported by the Master but missing from the Compact, is a resident real-time clock and calendar. Permanent Memory Systems, the company which produces the desktop utility GENIE, has recently released its answer in the form of a fully Master-compatible Compact Clock.

Product:	**Compact Clock**
Supplier:	**Permanent Memory Systems, 38 Mount Cameron Drive, East Kilbride G74 2ES. Tel. (03552) 32796**
Price:	**£34.90 inc. VAT**

The device is supplied in a plain box with a five page instruction leaflet. To my surprise I found that the Real Time Clock, complete with resident software and battery backup, was no larger than a normal ROM mounted on a carrier board. Previous Real Time Clocks have required both the connection of the necessary hardware and the installation of a sideways ROM to operate the clock. PMS have combined these two devices into the same package. The first paragraph of the instruction leaflet makes reference to the fact that the carrier board into which the ROM is plugged is known as a "Smart Socket" and contains both the RTC chip and its battery. The battery is not rechargeable but has a life expectancy of about ten years.

The fitting instructions are brief but to the point. Installation is no more complicated than inserting a normal ROM into one of the spare sideways ROM sockets. This should take no longer than about ten minutes and requires no technical skill or soldering. Once the device is fitted I would recommend that it is never removed for any reason as the legs are extremely flimsy and easily broken. It is surprising that a more durable base was not used.

Once in place, the Compact will accept exactly the same commands as would a Master for accessing the in-built Real Time Clock. Entering TIME$="Tue,19 Jan 1988.15:50:00" will set the date to the 19th January 1988 and the time to three fifty in the afternoon. The time is easily displayed with the *TIME command. As with the Master, the format of the date cannot be altered. The Compact Clock accommodates all legal methods of reading the Real Time Clock on the Master; even the Operating System calls OSWORD 14 and 15 are fully supported. This means that useful utilities such as the Date Stamping segment program for Wordwise Plus (published in BEEBUG Vol.6 No.6), will work without modification on a Compact.

Three further features increase the power of the PMS Real Time Clock considerably. The first allows the time to be displayed in the top right hand corner of the screen in either your own programs or in an application such as a word processor or spreadsheet. Unfortunately, although I have found this to work well with Wordwise and View, it did not work with Interword and I would anticipate other problems with commercial programs.

There is also an alarm facility and a timer. Up to four alarms may be set at any one time and, providing that the machine is on when the time elapses, an audible alarm will be sounded for about ten seconds. The timer also allows an alarm to be set relative to present time. For instance, if your program includes a line *TIMER 00:01:30 the alarm will sound exactly one and a half minutes after that line is executed.

Although the PMS Real Time Clock supports all Acorn approved methods of reading the Master Clock, this does not mean that all software will be able to use it. For example, any program which directly peeks into the CMOS RAM will fail in this way. Some programs, like BEEBUG's Master ROM, specifically ignore time and date facilities if the software determines it is running on a Compact. In addition, PMS have included a *NOALARM command which disables the clock display, alarms and timer, but does not clear the stored times.

One feature which I found disappointing was that the Real Time Clock Display enabled itself after a Ctrl-Break despite my having issued the *NOALARM command previously. I contacted PMS who assured me that this fault has been rectified and that the RTC will remain disabled after a hard reset.

SUMMARY

The Compact Clock is a very clever device which is easily installed and operated. It provides a very useful Real Time Clock and is well documented. The price may seem high for the facility provided, though the technology is very much state-of-the-art. If you really do need a Real Time Clock on your Compact, then this device from PMS will fit the bill admirably. B

Continued from page 43

This will display a list of directories located in the currently selected directory.

TOTAL SPACE USED

The following sequence, which should be typed in when all the routines so far discussed are already resident (i.e. lines 10-190 and lines 1000 upwards), displays the total number of bytes used by the objects in the current directory. This can be extremely useful when trying to find out which directories on a disc contain the files which use up the most space, and which should therefore be pruned to make space for other items. The routine makes use of both PROCfile (which it assumes has already been called), and PROCcatinfo. But this time PROCcatinfo is called with the parameter TRUE. The result is that the array **file%(n,m)** is filled with extended file information, containing file lengths, execution addresses, and so on. The former is then used to calculate the total space used by the files in the current directory.

```
  190:
  200 PROCcatinfo(TRUE)
  210 tot=0
  220 FOR A=1 TO fileno
  230 tot=tot+file%(A,3)
  240 NEXT
  250 PRINT"Total space used : "tot" byt
es"
```

In the next issue we will give further examples of the use of these three procedures. This will include their recursive calling to implement a complete disc cataloguer, as well as a flexible disc menu system. B

Hints Hints Hints

CONDITIONAL PRINTER BUFFER

David Graham

If you use an EXEC file to engage the Master ROM's printer buffer prior to listing a program or to engage a word processor, you may have noticed that the command:

```
*BUFFON
```

also clears the buffer of any resident data. This can be very annoying if you are halfway through a printout at the time.

The solution is simple: switch your buffer on only if it is already off. To do this, use:

```
IF ADVAL(-4)=63 THEN *BUFFON
```

The ADVAL function here returns the current size of the printer buffer. The default size for the Beeb's internal buffer is 63 bytes in length, and the command *BUFFON is only made if the default buffer is in place. I should add that if there happens to be just 63 bytes free in your extended buffer at the instant that the ADVAL is issued, your buffer will still be cleared; but the chances of this happening are extremely small.

APPENDING BASIC REVISITED

Dennis Weaver

The following update vastly improves the hint we gave last month. The accompanying EXEC file will append a Basic program to the one currently in memory. When you save this file (preferably in the Library directory of your disc), use a name other than APPEND (this is a reserved word) - APP would work well. Then to append a program at any time, just type *APP and supply a filename in response to the prompt.

```
*| FILE APPEND
VDU21
*KEY0 VDU21|MVDU6:INPUT"Prog Name? "F
  ILE$:A%=TOP-2:OS.("LOAD "+FILE$+" "
  +STR$~A%)|MOLD|M
*FX15,1
*FX138,0,128
```

B

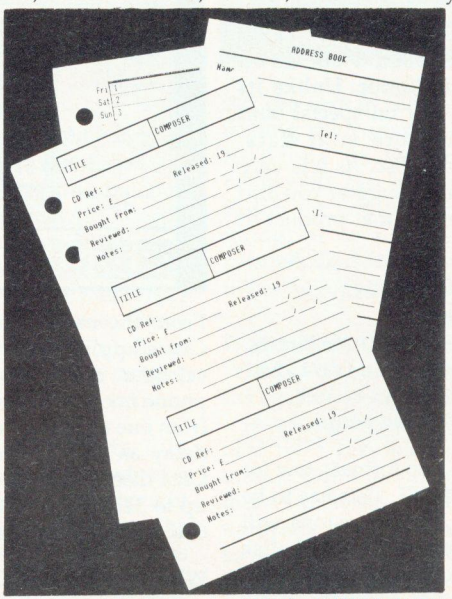
MEWSOFT'S PERSONAL ORGANISER

Dave Somers enhances his Filofax personal organiser with MEWsoft's Fax*File package to print custom forms and diaries.

Product	Fax*File Organiser
Supplier	MEWsoft, 11 Cressy Road, London NW3 2NB. Telephone 01-267 2642.
Price:	£12.95 inc. p&p.

Described by some as a 'Yuppie Pack', MEWsoft's Fax*File Organiser is a suite of programs enabling calendars, address books, and user-defined forms to be printed out ready for insertion into any personal organiser (e.g. Filofax, Leefax, etc.). To use the software you will require an Epson compatible printer capable of double density graphics.

For your money you get a 40 or 80 track DFS format disc containing the software, a 6 page A5 instruction manual, and a sample pack of computer paper which comes pre-punched for insertion into your organiser. An ADFS compatible version is also available.



THE ADDRESS BOOK

This allows you to create, edit, and print an address book of up to 48 entries. The format consists of 8 pages with 6 entries on each one. When editing the entries you are presented with a display of one of the pages as it will be printed. You can move a cursor over the fields and edit them as necessary. Once all the entries have been completed, they can be sorted alphabetically by surname, and saved to disc for later retrieval.

THE DIARY PRINTER

This allows you to print a diary for any year in either month-per-page or week-per-page formats. One very nice touch is the ability to have anniversaries or birthdays automatically printed alongside the appropriate date. A display is shown for each month and you can enter such messages if required.

THE FORMS DESIGNER

Probably the most useful facility of all is the forms designer which enables you to design and print your own style of forms.

You are presented with a display of a form over which you can place both horizontal or vertical lines, and text in any location. Such forms can be saved to disc for later retrieval. Some useful sample forms are provided with the software.

THE EXTENSION DISC

Four extra utilities are available on an extension disc which costs £9.95 inc. p&p. They are a label printer, and a short form address program which can print on one page a list of names and telephone numbers, both of which use the address book data files, a mini-database with up to 4 fields and 60 records, and a program to print an annual calendar on a single page.

FINAL COMMENTS

In use Fax*File is extremely friendly, with on-screen prompts and instructions. For example, when loading a data file, selecting a non-existent one will cause a list of available files to be displayed for you to select. As a utility program for the personal organiser enthusiast I cannot fault it.

Genie Junior & SideWriter

With 'Pop-Up' software becoming more and more popular, Dave Somers looks at the latest offerings for 1988.

INTRODUCTION

Last year saw the introduction of the first "pop-up" software for the Beeb in the form of Genie from PMS. Since then, Genie has been re-written to produce a cheaper version, and other companies have produced their own "pop-up" software. This review takes a look at what's currently available.

GENIE JUNIOR

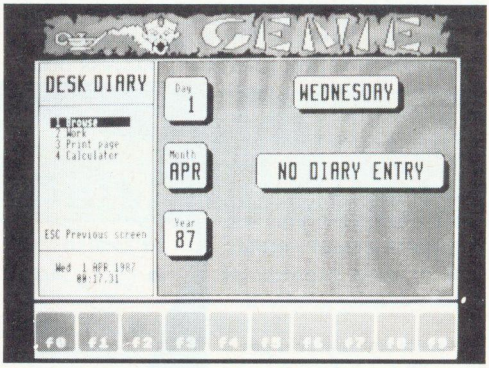
Following the success of Genie, Permanent Memory Systems has released a cut-down version aptly called Genie Junior. Unlike its elder brother, it does not rely on special (and expensive) hardware to store your data. Instead, it uses a ROM and a disc. As it is very similar in operation to Genie, I suggest you also refer to the previous review of that product in BEEBUG Vol.6 No.4.

Product	**Genie Junior Supplier**
Permanent	**Memory Systems**
	38 Mount Cameron Drive North,
	East Kilbride, Scotland G74 2ES.
	Tel. (03552) 32796 (24 hours)
Price	**£26.00 inc. VAT and p&p.**

Genie Junior is available in two formats, one for DFS use, and another for ADFS. This has to be specified when ordering. These versions will also work with most compatible RAM disc systems and Winchester hard disc systems. A network-compatible version is currently under development and should be available by the second quarter of 1988.

The ROM is fitted inside the Beeb in the usual manner, or it could be loaded as a ROM image into sideways RAM if you desire. It can be called upon in one of two ways - either pressing Shift-Ctrl-G or typing *GENIE. In either case, the screen changes to mode 7, and you are then prompted to enter the Genie Data disc. Before you can use the system, you have to make a copy of the Data disc supplied and use the copy - it will not work with the supplied master.

Once the Data disc has been placed in drive 0, there is a pause as data is transferred between the BBC micro and the disc. When that has been completed, the usual Genie screen is called up. If the system has been 'locked', you have to enter a password before you can access the data. Another security feature ensures that the ROM and disc supplied will only work with each other - a code is encrypted in the data disc and if it doesn't match that in the ROM it won't work.



From the main menu you can access the six Genie applications, namely the address book, calendar, calculator, desk diary, note pad, and phone book. When moving around the system, from one application to another, there is a slight delay as data is transferred to and from the Data disc. Unlike Genie which could only store up to 32K of data, Genie Junior can store up to 64K (if there is enough space on your disc).

When leaving the Genie Junior environment, you are prompted to insert the Data disc in the drive. After another pause, the computer is returned to its previous state, exactly as it was immediately before Genie Junior was invoked.

In operation, Genie Junior has the same general look and feel as Genie, although operation is slower as the data has to be accessed from disc. Indeed, the instruction manual supplied is

more or less identical to the original Genie manual, but with some extra supplements for the new working environment.

SIDEWRITER

For those who do not require all the bells and whistles provided by Genie Junior, DABS Press has an alternative, SideWriter, which is described by the originators as an "electronic notepad".

Product	**SideWriter**
Supplier	**DABS Press**
	76 Gardner Road, Prestwich,
	Manchester M25 7UH.
	Tel. 061-773 2413
Price	**£7.95 5.25" disc, £9.95 3.5" disc.**
	Prices inc VAT and p&p.

SideWriter is, in essence, a pop-up notepad facility. The package consists of a disc containing the software, and a four page A5 instruction manual, all packaged in a disc wallet.

The software is in the form of a ROM image on the disc for loading into any 16K Sideways RAM bank, a utility to perform this function which works with most popular RAM boards, and software to export data from SideWriter. Sideways RAM is thus an essential pre-requisite for the use of this software. SideWriter is entered in a very similar fashion to Genie Junior, except that the key combination is Shift-Ctrl-Tab. The screen then changes to a mode 7 display, and you're in.

SideWriter functions as a notepad facility (hence the later references to pad), based on a 128 by 76 character page which can be thought of as a 4 by 4 matrix of normal screens. The mode 7 display shows just one of these screens at a time. You can type in text anywhere, with the cursor keys to move the cursor around as expected. Unfortunately, the only way to edit the display is by 'overtyping', as there is no 'insert' mode for text entry. Colour can be introduced into your text by using the function keys to insert Teletext colours codes.

By using Shift in combination with the cursor control keys you can scroll to the next screen. If, however, you press Ctrl in combination with the cursor keys, SideWriter will 'jump' direct to the next screen. After entering your notes, you leave SideWriter by pressing the Tab key. This, just as with Genie Junior, causes the original screen display to be restored, and the computer to be returned to the state it was in before SideWriter was called.



All the text in SideWriter can be cleared by issuing the command *PADCLEAR. To obtain hardcopy of the contents of SideWriter *PADPRINT can be used to print it out to an Epson (or compatible) printer in condensed mode, or to a wide-carriage daisy-wheel printer. To save a pad to disc, the PADSAVE utility as supplied on the disc can be used.

FINAL WORDS

When it comes to choosing between the two, the choice is very difficult. SideWriter is in itself a very simple product, yet one that achieves its objectives very nicely and for a very modest outlay. Genie Junior on the other hand offers more functions, should you need them, and naturally at a commensurate price. I would have no hesitation in recommending either. The final decision depends on your own particular requirements. B

NOTE: We had hoped to include the long awaited Sidestep from Maze Technology in this review, but a copy was still not available at the time of writing. When Sidestep does appear we will endeavour to publish a review of it in BEEBUG magazine.

MACHINE CODE CROSS REFERENCER

Disassembling machine code programs sounds exciting but can be unrewarding unless you know what to look for. Jan Stuurman's utility will help sort out all the additional information you need to make sense of any machine code listing.

Anyone who has ever disassembled a machine code program knows how difficult it is to understand its workings. Even such simple operations as changing keys or parameters (lives, enemies, etc) in games programs require long hours of careful study if they are to be deciphered. The program listed below will help in various ways by finding subroutine and conditional branch entry-points, and by keeping track of all variables (addresses) used by a machine code program, to provide a comprehensive cross-reference listing.

The program mainly uses indirection operators, but also assembles a short machine code sort routine. Keyboard errors here may completely corrupt your new program so make sure you save it to disc before running it.

When the program is run you will first need to decide whether or not hard copy is required, as ALL information is then printed for reference, and then choose whether the target machine code (to be analysed) is a disc file (F), or in ROM or RAM (R). All input to the program is prompted, and most of it is checked before being accepted.

Next, specify the start address and length of the target machine code. The user may also specify any parts of the code to be skipped (up to 150 such sections) by giving the start and finish address for each section. Tracing through the target program halts when the address specified after the 'From' is about to be looked at, and resumes at the address specified after the 'To'. Blocks of data and text that are embedded in the code may thus be ignored. Failure to skip any data usually shows up in the output as entry-points outside the code, or addresses that do not make sense (e.g. in ROM). This feature is sometimes useful in finding all the data blocks in the code. The amount of output may be reduced by omitting the conditional branches. A 16K ROM, for example, is likely to contain as many as 1000 branches, and omitting these will save both time and memory. Output may further be restricted to operating system and zero page subroutines and addresses only (addresses below &400, or above >&BFFF).

```
BEEBUG M/C CROSS-REFERENCER

Zero page base                  b
Zero page variable              v
Zero page indirect              i

&0020   &8003i  &800Bv  &801Cv  &8022v
        &8052v  &8073i  &807Bi  &80D2v
        &810Ai  &8129v  &815Ci  &816Dv
        &8196v  &81ADv  &81C1v  &81DBb
        &820Ei  &8211v  &8231v  &823Fi
        &8256i  &8260v  &8272v  &828Fv
        &82A1v  &828Ci  &82C1i  &8305i
        &8316i  &8344v  &8376v  &8393i
        &83A5v  &83B6i  &83C4i  &83CCv
        &83E1i  &83F3v  &8401v  &8473v
        &8491i  &8496v  &84A3i  &84DFv
        &84ECv
&002F   &8441v
&0030    &80B2v
```

Output is in paged mode (press Shift to scroll) unless it is directed to a printer and consists of a summary table followed by the cross-reference listing as determined by the initial dialogue. The Beeb's own operating system can also be cross-referenced since the ROM number is ignored for addresses below &8000 or above &BFFF.

PROGRAM NOTES

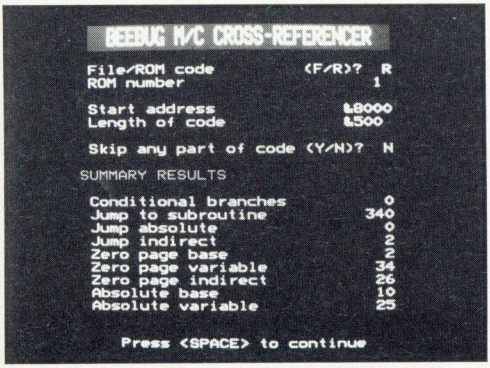
The program should be run with PAGE set to &1300, though on a Master, Compact or when using the OPUS DDOS, PAGE may be left at its default of &E00.

PROCinit sets up two byte arrays, B% and C%. The array B% stores the number of bytes each 6502 opcode occupies minus one (the NEXT in line 1530 adds one). Array C% is used to classify each opcode, with 0 for implied, accumulator or immediate addressing, and 1 to 9 as given by the array code$().

PROCtrace inspects each byte of the target code and, depending on its class, either ignores it or

stores the reference. When a BRK instruction is found, the variable byte is set to 0, and PROCbrk skips all code up to and including the next zero byte. Embedded error messages that follow Acorn's standard will thus be skipped automatically. For conditional branches, the actual destination address is calculated. Each reference is stored above HIMEM and below the mode 7 screen in a five byte format:

2 byte operand (high byte first)
2 byte source address (id)
1 byte reference class



A 'trace overflow' error will be displayed when there is insufficient memory to store the next reference. The position at which tracing halts is displayed, and the program continues by proceeding directly to the ordering and output of the results so far obtained. The same error message will be given when an attempt is made to read past the end of a disc file. This error occasionally occurs when tracing embedded data.

The array code%() keeps running totals of stored references. PROCstat shows these totals for each class when tracing is complete. References are then sorted by a small machine code insertion sort routine and listed under four headings:

Conditional branches,
Jumps,
Zero page addresses,
Absolute addresses.

With the latter three groups, a lower case letter classifies each particular reference.

```
   10 REM Program MCCROSS
   20 REM Version B1.1
   30 REM Author  Jan Stuurman
   40 REM BEEBUG  Jan/Feb 1988
   50 REM Program subject to copyright
   60 :
  100 MODE7:HIMEM=PAGE+&2000
  110 ON ERROR GOTO200
  120 PROCtitle
  130 PROCinit
  140 PROCinput
  150 PROCtrace
  160 PROCstat
  170 PROCresults
  180 VDU3,15:MODE7:END
  190 :
  200 ON ERROR OFF:CLOSE#0
  210 IF ERR=193 overflow=T%:GOTO160
  220 VDU3,15:VDU23,1,1;0;0;0;
  230 MODE 7:REPORT:PRINT" at line ";ERL
  240 END
  250 :
 1000 DEFPROCtitle
 1010 FOR I=0 TO 1
 1020 PRINTTAB(2,I)CHR$141CHR$129CHR$157
CHR$131"BEEBUG M/C CROSS-REFERENCER  "CH
R$156
 1030 NEXT:PRINT
 1040 PR%=FNchrinp("YN",3,"Output to pri
nter",26)
 1050 PRINTTAB(11)CHR$131"Please wait ..
."
 1060 ENDPROC
 1070 :
 1080 DEFPROCinit:LOCALI%,J%
 1090 B%=&B00:C%=&C00:sort%=&900
 1100 Z%=HIMEM-5:K%=150:overflow=FALSE
 1110 DIMskip%(K%,1),code%(9),code$(9)
 1120 skip%(0,0)=&FFFF:skip%(0,1)=&FFFF
 1130 pcode$=" saibvibv"
 1140 FORI%=0TO255STEP32:FORJ%=0TO31
 1150 B%?(I%+J%)=VALMID$("11011111010122
221101111102022222",J%+1,1)
 1160 C%?(I%+J%)=VALMID$("07076666000099
991707555508088888",J%+1,1)
 1170 NEXT:NEXT
 1180 ?B%=0:B%?32=2:B%?64=0:B%?96=0:B%?1
62=1
 1190 C%?32=2:C%?76=3:C%?108=4
 1200 FORI%=1TO9:READ code$(I%):NEXT
 1210 DATA Conditional branches
 1220 DATA Jump to subroutine,Jump absol
ute,Jump indirect
 1230 DATA Zero page base,Zero page vari
able,Zero page indirect
 1240 DATA Absolute base,Absolute variab
le
 1250 ENDPROC
 1260 :
 1270 DEFPROCinput:VDU28,0,24,39,3,12
 1280 IF PR% VDU2,1,13,10,1,27,1,64
```

```
 1290 type%=FNchrinp("FR",3,"File/ROM co
de",26)
 1300 IF type% PRINTTAB(3)CHR$134"Enter
filename"TAB(21)CHR$131;:INPUT""file$' E
LSE PRINTTAB(3)CHR$134"ROM number"TAB(32
)CHR$131;:INPUT""rom%'
 1310 strt%=FNhexinp(3,"Start address",3
0)
 1320 leng%=FNhexinp(3,"Length of code",
30)
 1330 PRINT':fini%=strt%+leng%-1
 1340 IF FNchrinp("YN",3,"Skip any part
of code",26) PROCskip
 1350 VDU28,0,24,39,17,1,13
 1360 CB%=FNchrinp("YN",3,"Conditional b
ranches",26)
 1370 OS%=FNchrinp("YN",3,"OS subroutine
s only",26)
 1380 OA%=FNchrinp("YN",3,"ZP/OS address
es only",26)
 1390 ENDPROC
 1400 :
 1410 DEFPROCtrace:LOCALI%,R%,S%,T%
 1420 VDU26,3,23,1,0;0;0;0;
 1430 IF type% PROCdisc
 1440 PRINTTAB(13,24)CHR$133CHR$136"TRAC
ING";
 1450 I%=0:fini%=strt%+leng%-1
 1460 FORT%=strt%TOfini%
 1470 IF T%>=skip%(I%,0) T%=skip%(I%,1):
I%=I%+1:IF T%>=fini% GOTO1530
 1480 S%=FNgetbyte(T%):IF S%=0 PROCbrk:G
OTO1530
 1490 IF C%?S%=0 GOTO1520
 1500 ON C%?S% GOSUB1560,1610,1610,1610,
1650,1650,1650,1680,1680
 1510 IF Z%>&7BF5 overflow=T%:T%=fini%
 1520 T%=T%+B%?S%
 1530 NEXT
 1540 ENDPROC
 1550 :
 1560 IF NOTCB% RETURN
 1570 R%=FNgetbyte(T%+1)
 1580 IF R%<128 R%=T%+R%+2 ELSE R%=T%+R%
-254
 1590 PROCput:RETURN
 1600 :
 1610 R%=FNgetbyte(T%+1)+256*FNgetbyte(T
%+2)
 1620 IF OS%AND(R%<&200 ORR%>&235 ANDR%<
&C000) RETURN
 1630 PROCput:RETURN
 1640 :
 1650 R%=FNgetbyte(T%+1)
 1660 PROCput:RETURN
 1670 :
 1680 R%=FNgetbyte(T%+1)+256*FNgetbyte(T
%+2)
 1690 IF OA%ANDR%>&3FF ANDR%<&C000 RETUR
N
 1700 PROCput:RETURN
```

```
 1710 :
 1720 DEFPROCstat:LOCALI%
 1730 IF PR% VDU2:PRINT'':VDU1,27,1,69
 1740 IF overflow VDU7:PRINTTAB(3,9)CHR$
129"TRACING OVERFLOW ERROR AT &";~overfl
ow:VDU1,13,1,13
 1750 VDU28,0,24,39,11:IF NOT PR% VDU12
 1760 PRINTTAB(11)CHR$133"SUMMARY RESULT
S"'
 1770 VDU1,27,1,70:FORI%=1TO9
 1780 PRINTTAB(3)CHR$134;code$(I%)TAB(30
)CHR$131;RIGHT$("    "+STR$code%(I%),4)
 1790 NEXT:PROCspace
 1800 ENDPROC
 1810 :
 1820 DEFPROCresults
 1830 PRINTTAB(4,24)CHR$133CHR$136"SORTI
NG"CHR$137CHR$131"Please wait ...";
 1840 PROCassem:CALL sort%
 1850 PRINTTAB(4,24)SPC28;
 1860 IFCB% PROClist(1,1)
 1870 PROClist(2,4)
 1880 PROClist(5,7)
 1890 PROClist(8,9)
 1900 ENDPROC
 1910 :
 1920 DEFPROCskip:LOCALI%
 1930 VDU28,0,15,32,11,1,13,1,13
 1940 I%=-1:REPEAT I%=I%+1
 1950 skip%(I%,0)=FNhexinp(3,"From ",11)
 1960 IF skip%(I%,0)=&FFFF GOTO1980
 1970 skip%(I%,1)=FNhexinp(17,"To ",23)
 1980 UNTIL skip%(I%,0)>=fini% OR skip%(
I%,1)>=fini% OR I%=K%
 1990 ENDPROC
 2000 :
 2010 DEFPROCdisc:LOCALA%,X%,Y%,I
 2020 REPEAT PRINTTAB(0,24)SPC39;
 2030 PRINTTAB(2,24)CHR$133CHR$136"Inser
t disc and press <SPACE>";
 2040 REPEAT UNTIL GET=32:PRINTTAB(0,24)
SPC39;
 2050 A%=5:X%=&70:Y%=0
 2060 !&70=&100:$&100=file$:CALL&FFDD
 2070 load%=!&72 AND&FFFF:size%=!&7A
 2080 IF strt%<load% strt%=load%
 2090 IF leng%>=size% leng%=size%-1
 2100 F%=OPENIN(file$)
 2110 IF F%=0 VDU7:PRINTTAB(3,24)CHR$129
"File ";file$;" not found.";:I=INKEY400
 2120 UNTIL F%<>0:ENDPROC
 2130 :
 2140 DEFPROClist(C1,C2):LOCALF%,G%,H%,I
%,T%,M%,S%
 2150 VDU28,0,22,39,3,12
 2160 IF PR% VDU2,1,12,1,27,1,69:M%=10:S
%=2 ELSE M%=5:S%=1
 2170 FORI%=C1 TO C2
 2180 PRINTTAB(3)CHR$134;code$(I%);TAB(3
0)CHR$131;MID$(pcode$,I%,1)
 2190 NEXT
```

```
 2200 F%=&10000:VDU28,0,22,39,8:IFPR% VD
U1,27,1,70 ELSEVDU14
 2210 FORI%=HIMEM TO Z% STEP5
 2220 IF I%?4<C1 ORI%?4>C2 GOTO2270
 2230 G%=256*I%?0+I%?1:H%=256*I%?2+I%?3
 2240 IF G%<>F% PRINT'CHR$133;FNhexout(G
%)" ";:F%=G%:T%=0
 2250 T%=(T%+1)MODM%:IF T%=0 T%=1:PRINT'
SPC8; ELSE PRINTSPCS%;
 2260 PRINTCHR$134;FNhexout(H%);MID$(pco
de$,I%?4,1);
 2270 NEXT:PROCspace
 2280 ENDPROC
 2290 :
 2300 DEFFNchrinp(A$,X%,text$,A%):LOCALG
%,G$
 2310 PRINTTAB(X%)CHR$134;text$;TAB(A%)"
(";LEFT$(A$,1);"/";RIGHT$(A$,1);")? "CHR
$131;
 2320 REPEAT:*FX15 1
 2330 G$=CHR$(GETAND&5F):G%=INSTR(A$,G$)
 2340 UNTIL G%
 2350 PRINTG$:=G%-2
 2360 :
 2370 DEFFNhexinp(X%,text$,A%):LOCALG%,G
$
 2380 PRINTTAB(X%)CHR$134;text$;CHR$131;
TAB(A%)"&";
 2390 REPEAT:*FX15 1
 2400 G%=GET
 2410 IF (G%>47ANDG%<58)OR(G%>64ANDG%<71
) G$=G$+CHR$G%:VDUG%
 2420 IF G%=127ANDLENG$>0 G$=LEFT$(G$,LE
NG$-1):VDUG%
 2430 UNTIL G%=13
 2440 IF G$="" G$="FFFF":PRINTG$;
 2450 =EVAL("&"+G$)
 2460 :
 2470 DEFFNhexout(X%)
 2480 ="&"+RIGHT$("000"+STR$~X%,4)
 2490 :
 2500 DEFPROCbrk
 2510 REPEAT T%=T%+1
 2520 UNTIL FNgetbyte(T%)=0ORT%=skip%(I%
,0)ORT%=fini%
 2530 ENDPROC
 2540 :
 2550 DEFFNgetbyte(T%)
 2560 IF type% THEN PTR#F%=T%-load%:=BGE
T#F%
 2570 !&F6=T%:Y%=rom%:=USR(&FFB9)AND&FF
 2580 :
 2590 DEFPROCput Z%=Z%+5
 2600 Z%?0=R%DIV256:Z%?1=R%MOD256
 2610 Z%?2=T%DIV256:Z%?3=T%MOD256
 2620 Z%?4=C%?S%
 2630 code%(C%?S%)=code%(C%?S%)+1
 2640 ENDPROC
 2650 :
 2660 DEFPROCspace
```

```
 2670 VDU1,13,10,3,26,15:*FX15,1
 2680 PRINTTAB(6,24)CHR$131"Press <SPACE
> to continue";
 2690 REPEAT UNTIL GET=32:PRINTTAB(5,24)
SPC28;
 2700 ENDPROC
 2710 :
 2720 DEFPROCassem
 2730 base=&70:end=&72:kfiv=&80
 2740 iptr=&74:jptr=&76:kptr=&78
 2750 !base=HIMEM:!end=Z%
 2760 FORpass=0TO2STEP2:P%=sort%
 2770 [OPT pass
 2780 CLC
 2790 LDA base:ADC #5:STA jptr
 2800 LDA base+1:ADC #0:STA jptr+1
 2810 .jloop LDY #4
 2820 .trans1
 2830 LDA (jptr),Y:STA kfiv,Y
 2840 DEY:BPL trans1
 2850 SEC
 2860 LDA jptr:SBC #5:STA iptr
 2870 LDA jptr+1:SBC #0:STA iptr+1
 2880 .iloop LDY #0
 2890 .compare
 2900 LDA kfiv,Y:CMP (iptr),Y
 2910 BNE endcmp
 2920 INY:CPY #5:BCC compare
 2930 .endcmp BCS greq
 2940 .less CLC
 2950 LDA iptr:ADC #5:STA kptr
 2960 LDA iptr+1:ADC #0:STA kptr+1
 2970 LDY #4
 2980 .shift
 2990 LDA (iptr),Y:STA (kptr),Y
 3000 DEY:BPL shift
 3010 SEC
 3020 LDA iptr:SBC #5:STA iptr
 3030 BCS cont1:DEC iptr+1
 3040 .cont1
 3050 CMP base:LDA iptr+1:SBC base+1
 3060 BCS iloop
 3070 .greq CLC
 3080 LDA iptr:ADC #5:STA iptr
 3090 BCC cont2:INC iptr+1
 3100 .cont2 LDY #4
 3110 .trans2
 3120 LDA kfiv,Y:STA (iptr),Y
 3130 DEY:BPL trans2
 3140 CLC
 3150 LDA jptr:ADC #5:STA jptr
 3160 BCC cont3:INC jptr+1
 3170 .cont3
 3180 LDA end:CMP jptr
 3190 LDA end+1:SBC jptr+1
 3200 BCS jloop
 3210 RTS
 3220 ]:NEXT:ENDPROC
```

1st COURSE

Getting Animated

Mike Williams investigates some of the techniques needed for achieving smooth, flicker-free animation on the screen.

In the previous First Course article, when dealing with colour and its uses, I used an example program in which a 'car' was made to move across the screen from left to right. Now the movement was hardly smooth, and I said at the time I would deal with this subject of animation in more detail in a future article. Well here it is.

First of all let's be quite clear as to the form of animation we are talking about, as apparent screen movement can be achieved in various ways. What I am referring to is any means by which an object displayed on the screen can literally be made to change position. There are other techniques, using colour manipulation, by which objects appear to exhibit movement, but the observed motion is achieved by pre-drawing the object concerned in a sequence of different positions, and then revealing each one in turn.

As I stated last time, the basic principle is to display an object on the screen, then remove it and redisplay it in a slightly different position. By repeating this process an object appears to move. The technique can be applied to 'drawn' graphics (that is objects produced using MOVE, DRAW and PLOT) or to character graphics, usually with user-defined characters. Any drawn object will take some time to emerge, particularly if it is at all complex, and thus animation of wire-frame objects is not as successful, particularly if undertaken in Basic. However, once designed, any character can be quickly placed on the screen 'ready made' as it were.

Even so, the simpler the object the quicker and smoother the technique will appear. An object consisting of one character will work better than an object consisting of two or more. In order to examine this technique further consider the following program (the line numbering leaves room for later additions).

```
   10 REM Program Alien
  100 MODE 4:ON ERROR GOTO 230
  120 PROCdefine
  140 REPEAT
  160 PROCmove
  180 UNTIL FALSE
  200 END
  220 :
  230 MODE 7:REPORT:PRINT" at line ";ERL:E
ND
  240:
 1000 DEF PROCdefine
 1020 VDU23,224,66,66,126,90,126,60,36,102
 1040 alien$=CHR$224
 1060 ENDPROC
 1080 :
 1100 DEF PROCmove
 1120 VDU5:GCOL3,1
 1140 FOR P=0 TO 1280 STEP 4
 1160 MOVE P,300:PRINT alien$;
 1180 MOVE P,300:PRINT alien$;
 1200 NEXT P
 1220 VDU4
 1240 ENDPROC
```

The program uses a procedure to define any characters we need, in this case just one called alien$, and a second procedure called PROCmove to animate that object. Although the object is a character, VDU5 at line 1120 combines graphics and text cursors so that the future position of any character on the screen is determined by the position of the graphics cursor, hence the subsequent use of MOVE.

Once VDU5 has been executed, GCOL rather than COLOUR is used to select both the colour to plot, and the mode of (graphics) plotting. GCOL3,1 specifies Exclusive-OR plotting. This mode, as explained in previous articles, has the effect that plotting an object twice in succession in exactly the same position causes the object to appear and then disappear, exactly what is needed for animation.

Lines 1140 to 1200 consist of a loop in which our 'alien' is repeatedly displayed on the screen and then removed. Each time the loop is repeated the position of the alien is moved to the right as specified by the step size in the loop. Increasing or decreasing the step size will, as might be expected, speed up or slow down the apparent rate of movement. You decide what you need.

However, remember that the choice of screen mode determines horizontal graphics resolution. In modes 2 and 5 the minimum step size is eight, in modes 1 and 4 the minimum is four, and in mode 0 the minimum is two. In any mode, selecting a step size less that the minimum for that mode will not result in movement of the object every time the loop is repeated, but movement will still appear to be slowed down as more loops are needed to achieve the minimum step size necessary for movement. Try out the program, using different step sizes and different modes to see the effect.

You will probably have found that most variations you try result in obvious flicker in the moving alien. The problem is that the two statements to place and remove the alien are adjacent, and consequently the alien is no sooner visible than it is removed. Thus within each loop, more time is spent without the alien on the screen than is spent with. That's why flicker results. To reduce flicker we must seek ways of maximising the time the alien is visible, and minimising the time it is invisible. One way of achieving this is to insert an INKEY statement between the two PRINT statements. Try adding:

```
1170 Z=INKEY(5)
```

You should find that this does reduce flicker, but of course it also slows down the speed of movement too.

A better solution is to change the loop so that the time between the two PRINT statements becomes the time the object is removed rather than the time for which it is visible. To achieve this we need an extra statement before the loop is entered to place the alien on the screen, and then within the loop remove it and straight away put it back. But it must be put back one step further on. So add line 1130 and amend line 1180 as follows:

```
1130 MOVE 0,300:PRINT alien$;
1180 MOVE P+4,300:PRINT alien$;
```

If you are going to experiment, you will find it more convenient to assign the step size (currently 4) to a variable (say S%) additionally in line 1120 and replace the '4' above and in line 1140 with this. You should find that you now have a more smoothly moving alien. However, flicker is still likely to be obvious, and if you look closely you will see an occasional ripple run vertically through the moving alien.

This results from the way in which the monitor screen is scanned electronically to maintain a picture. The picture scans from top to bottom, and the ripples are caused by the screen refresh coinciding with removal and replacement of the alien on the screen. There is, however, an FX call which will solve most of our problems, and this is *FX19. If you check this out in your user guide it will be defined as 'wait for vertical synchronisation'. Without going into the technical detail, this gives us the maximum amount of time to 'move' the alien on the screen between one screen refresh and the next, thus hiding as much as possible of the change to the position of the alien itself (or whatever graphics you are using). Add a new line 1150 to the program as follows:

```
1150 *FX19
```

The result should be that your alien has been turned into a real smoothie, gliding effortlessly across the screen. Again try experimenting by putting the *FX19 call at different points within the loop and observe the effect. In our example it is positioned immediately before the two statements which remove and then replace the alien on the screen.

Well there we have a variety of methods and ideas that should help to animate your screen displays. Remember two things in particular. All computer generated animation is an illusion, just like all those movies with their endless succession of 'still' pictures. Secondly, the more complex the image you are trying to animate (e.g. multi-character objects) the more difficult you will find it to achieve smooth results. The next time you see a computerised arcade game in action, just note how little movement there often is, and how small the moving objects are. There are always limits to what can be done, even if those limits are moving forward.

TWO "C" COMPILERS REVIEWED

C is suddenly the 'in' language, and two more C compilers have appeared for the BBC micro. Ray Hughes, who reviewed Acornsoft's C for BEEBUG, looks at the competition.

Product	**Brasscourt C**
Supplier	**Brasscourt Ltd** **Oakford Farm, Marshfield** **Chippenham, Wilts SN14 8AN.** **Tel. (0249) 655980**
Price	**£95.75 inc VAT**
Product	**Mijas Small C Supplier** **Mijas software** **Mijas, Winchester road** **Michedever, Hants SO21 3DG.**
Price	**£50.00 inc VAT**

This is a second look at some of the recent versions of the C programming language that have been made available for the BBC micro and Master series. In Vol.6 No.5 I reviewed the Acornsoft version (with some reference to BEEBUG's own C Compiler); this month I shall take a look at two more versions and assess their potential.

One of the most surprising things about the four C compilers now available for the Beeb is the fact that for a so-called standard language these packages really are quite different in how they work, what they are capable of, and what systems they can be run on.

MIJAS "SMALL" C

This implementation is a combination of three different software packages, the Small C compiler, the 65C02 development package and the debugger. Mijas will supply the software on a number of different disc formats (DFS, ADFS etc) and all the ROM images on the disc can be supplied as proper ROMs if requested. 40 track DFS format drives are not recommended for use with this software.

This suite of programs is very useful for the development of assembly language programs, with the additional benefit of of a C based structure.

Using this software suite, it is possible to automate virtually the whole process of program creation. This is done using the MAKE directive which enables a pre-created ASCII file (MEMFILE) to drive all the individual program stages directly. It is somewhat similar to using a standard BBC EXEC file, but the MAKE version is much more flexible in operation. The linker, debugger and assembler can all be run directly with the MAKE facility. There are also a number of other ways in which this utility really can automate and speed up the generation of your software including code relocation. It allows for the use of labels within the MAKE file which are then replaced by the required file names at execution time.

```
TCPP $1_C :2.TCPP_I
BASIC
CH."TCC"
:2.TCPP_I
$1
:2.ASM_A
*SHELL
ASMB :2.ASM_A :2.ASM_O AX
LINK
CODE 1902 $1
RE :2.ASM_O
SE :2.CRT0_O
SE :2.LIB_O
SE :2.SYS_O
SE :2.CRT1_O
SE :2.OSHDR_O
EN
PU :2.TCPP_I
PU LITFILE
PU :2.ASM_O
```

In the above example (MEMFILE), the file is run from within the 'shell' environment. The first line then calls the C pre-processor (TCPP) using the given filename (i.e. MAKE SIEVE). During execution of this MAKE file, the $1_C string is converted to SIEVE_C. The pre-processor then writes its output to an intermediate file on drive 2 called TCCP_I. The MAKE file then leaves the shell environment, drops back to Basic and chains to the C Compiler (TCC), supplying as a source file :2.TCPP_I. This file is then assembled to machine code, linked with all the relevant

library files etc, and is given a RUN address of &1902. The PU commands at the end are simply removing the intermediate files that were created during the above process.

Another point to note is that MOST of the software, but not ALL, is Tube compatible. It is also supposed to run under the ADFS, but my experiences proved it was easier to work under DFS with the Tube turned off. This is on my ancient old BBC and may not be the same as on the Master, particularly as PAGE on the Master is always set at &E00 whatever filing system is currently selected.

BRASSCOURT C

This is the first software I have ever used on the Beeb which has been dongled. That is to say, a small hardware device has to be fitted to the user port to enable the software to run. However, once the package has been used to produce stand alone code, the dongle is no longer required to run the resulting compiled programs.

The software consists of several machine code files that are simply *RUN to operate. However, program development does not follow the usual path of:

```
EDIT SOURCE
COMPILE SOURCE
LINK with LIBRARY files
RUN
DEBUG
RE-EDIT etc etc.
```

With Brasscourt C, one simply creates a source text file. A reasonable editor is supplied for this. After that:

```
*CC sourcefile TO codefile
*ASM codefile otherfiles TO runfile
```

No linking is required; the compiler produces a compact form of p-code. The Assembler then converts this, with any other additional programs to a final stand-alone machine code file. Brasscourt C generates compact code which is quite fast in operation. The main fault in the software, as supplied, is the lack of anything like a reasonable collection of library routines, but more of this later.

FINALLY

The alternative Acornsoft and BEEBUG versions of C are both very near to the accepted standard, and both offer added enhancements to make fuller use of the Beeb. The BEEBUG version generally produces more compact code. The speeds of both versions are quite similar, but at the moment only the Beebugsoft version is capable of producing stand alone code (albeit with the addition of an extra program). Both have excellent manuals, and can be heartily recommended.

Brasscourt and Mijas fall into a different camp. While neither of these packages seems to have any real faults, two of the C language's better known plus points are the standardisation of operation and the portability of source code. Brasscourt and Mijas do not attempt to conform to the norm as far as operation goes, nor are they particularly portable. They do both produce stand-alone code and they both have reasonable manuals. In summing up I believe that the Brasscourt version is very easy to use (if the user port is not required for a mouse), though both Brasscourt and Mijas versions are limited by their very simple libraries.

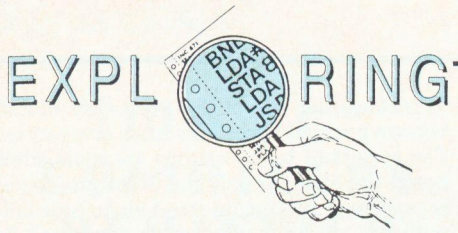
The Mijas version is much more complicated to use, and takes a while to get the best out of it; it is really an Assembler package that allows the use of a limited subset of the C structure. As an assembler development package this is fine, but as a development tool for C programs it is quite poor. The prices of the Brasscourt & Mijas versions are expensive considering what they offer. The Acornsoft and BEEBUG versions both offer good value for money, and are excellent systems to learn the C language with.

BENCHMARKS

As a rough comparison of speed and compiled code size the PCW INTMATH benchmark was run using all four versions of the C language for the BBC micro (model B) with the following results.

INTMATH	RUNtime (secs)	Codesize (bytes)
Acornsoft	1.3	8876
Beebugsoft	2.5	230
Brasscourt	1.8	1430
Mijas	2.0	4656

Full details of the INTMATH benchmark program used can be obtained from BEEBUG Vol.5 No.6.



ASSEMBLER

Part 7

A series for complete beginners to machine code by Lee Calcraft

This month: Decision making using the CMP, CPX and CPY instructions, including the implementation of a bubble sort routine.

In part 3 of this series we introduced the CMP instruction, using it for simple checks to see if two bytes were equal in value. In reality CMP is a powerful and frequently used instruction, though in many respects its power derives from the various branch instructions with which it is used. As you can see from the table, CMP may be used with a wide range of addressing modes. The table also gives details of the two associated instructions CPX and CPY which perform a similar function with the X and Y registers as CMP performs with the accumulator. You will note that a much smaller range of addressing modes is available with these latter instructions.

All three instructions can take as operand an address or immediate data, and the effect of each instruction is to compare register with operand. Thus:

```
LDA &70
CMP #8
```

will load the accumulator with the contents of address &70, and then compare it with the immediate value 8. The direct equivalent using CPY and the Y register would be:

```
LDY &70
CPY #8
```

and similarly for CPX and the X register. In all three cases the result of the compare instruction appears in three processor flags: Z (the zero flag), C (the carry flag), and N (the negative flag). The accumulator and X and Y registers are left *unchanged* by their respective compare instructions. What the processor does is to perform a subtraction of the operand from the appropriate register, and set the flags according to the result, but then it discards the result, leaving the register containing the same number that it contained before the comparison took place.

ADDRESSING MODES SUPPORTED

CMP	CPX CPY
Immediate	Immediate
Absolute	Absolute
Zero page	Zero page
Zero page indirect*	
Absolute indexed	
Zero page indexed	
Pre-indexed indirect	
Indirect indexed	

*** 65C12 only**

HOW THE FLAGS ARE SET

To see how the flags are set, consider the following:

```
LDA #8
CMP #5
```

Here the accumulator is loaded with the number 8. Then the CMP #5 instruction causes 5 to be subtracted from it. The three flags are then set as follows:

Z=0 (since the result is not zero)
C=1 (since the subtraction does not involve a borrow)
N=0 (since bit seven of the result is not set)

Once the flags are altered, the accumulator is reloaded with its original contents (8).

Of the three flags affected by the instruction, the zero flag performs exactly as we would expect. The negative flag also performs consistently, but you should remember that because the negative flag simply and only reflects the state of the top bit of the result (see part five of this series), it *cannot* be used to indicate that the result of the comparison is negative. For example, if we loaded the accumulator with the value 200, and compared it with 5, the N flag would be *set* on the result, since 200-5=195, and all numbers greater than 127 have the top bit set, thus causing the negative flag itself to be set.

In many respects the most important of the three flags affected by the compare instructions is the carry flag, and understanding the way in which it is set requires a little thought. The best way to see what is happening may be to think of the SBC instruction (again, see part five of this series), which alters this flag in a similar way. Before the compare instruction is executed, the processor automatically sets the carry flag, just as the user must do before performing a single-byte subtraction. It then *unsets* this flag if a borrow was, or indeed would have been, necessary. If no borrow takes place, the operation will leave the carry flag *set*. The accompanying table gives a number of examples, and it is worth taking a look at these before proceeding.

EXAMPLE RESULTS OF USING CMP

Accumulator	8	8	8	200
Operand	5	8	9	5
Simulated Result	3	0	255	195
Zero flag	0	1	0	0
Carry flag	1	1	0	1
Neg flag	0	0	1	1

TESTING FOR A=OPERAND

From this table, we can easily deduce the sequence of instructions required when testing for a variety of conditions. To check if two numbers are equal, we need only use BEQ (Branch if EQual) or BNE (Branch if Not Equal). Thus:

```
LDA &70
CMP #8
BEQ match
```

will cause a branch to the label **match** if address &70 contains the value 8. As you may remember, both BEQ and BNE test the state of the zero flag.

TESTING FOR A<OPERAND

To test if &70 contains a number less than 8, we can test the carry flag using BCC (Branch if Carry Clear). Thus:

```
LDA &70
CMP #8
BCC lessthan
```

This will cause a branch to the label **lessthan** if the contents of &70 are less than 8. This occurs because in the simulated subtraction, there will be a borrow if the number in the accumulator is less than the operand (8). In such a case the carry flag will be cleared, and we can test for this with BCC.

TESTING FOR A>=OPERAND

If we replace the BCC instruction with BCS (Branch if Carry Set), a branch will be performed not only when &70 contains a number *greater* than 8, but also when the two numbers are *equal*. Thus using BCS tests for a *greater than or equal* condition (>=). This occurs because in the simulated subtraction, no borrow is required as long as &70 contains a number greater than or equal to 8. To test for a simple *greater than* condition, we must first eliminate the possibility of equality. Thus:

```
LDA &70
CMP #8
BEQ equal
BCS greaterthan
```

The routine will now only branch to the label **greaterthan** if the contents of the accumulator are indeed greater than the operand.

TESTING FOR A<=OPERAND

To branch if &70 contains a number less than or equal to the operand, we must also use two branch instructions:

```
LDA &70
BEQ lessthanorequal
BCC lessthanorequal
```

The first branch instruction (BEQ) picks up the equality condition, while the second acts when the accumulator contains less than the operand.

The foregoing applies equally well to the three compare instructions, and from the examples given we should now be able to detect any required condition. We are thus in a good position to look at some real applications of these instructions.

ORDERING TWO NUMBERS

The object here is to place two single-byte numbers in order of magnitude. We will assume that they are held in locations &70 and &71, and that we require the first location to hold the larger number. We can obviously start our code by loading the first or the second

number into the accumulator, and then comparing it with the other. If we load the location which is to hold the highest number on exit, then we can make use of the fact that the BCS instruction tests for a *greater than or equal* condition. Thus if we start with the following sequence:

```
LDA &70
CMP &71
BCS exit
```

we will branch to the label **exit** if &70 contains a number greater or equal to that in &71, and under these circumstances, no change in the numbers is required. We now need only to swap the two bytes if the branch is not made. Here is the complete routine:

```
LDA &70
CMP &71
BCS exit
LDX &71
STA &71
STX &70
.exit
```

MULTI-BYTE COMPARISONS

In the case of multi-byte comparisons the procedure is a little more complex. We must compare the two numbers one byte at a time, starting at the highest byte. If the two bytes are in the right order, then we can end the routine, while if the top byte of the first number is less than the top byte of the second, we must swap the two numbers. But there is a third possible condition: the two top bytes might be equal. In this case we must compare the next highest bytes, and so on.

Suppose that we have two four-byte numbers stored at &70-&73 and &74-&77, and that they are stored low byte first, as is usual with the 6502. The code below will do the trick:

```
LDY #4
.loop
LDA &6F,Y
CMP &73,Y
BCC swap
BNE noswap
DEY
BNE loop
JMP noswap
.swap
\ Swapping routine here
.noswap
```

Here we start by loading Y with 4, since we will be counting backwards in order to check the high bytes first, and there are four pairs of bytes to check at maximum. The next two instructions load and compare the first two bytes. We use **BCC swap** to branch to the swapping routine if the first number is less than (but not equal to) the second. The two remaining possibilities are either equality (in which case we must check the next pair of bytes), or that the first number is greater than the second (in which case no swap is necessary). We decide which by using **BNE noswap** to filter out the condition where no exchange is required. Y is then decremented, and checked against zero. Note that no CPY instruction is required, since if Y=0 the zero flag is automatically set. This is why it is more efficient to arrange index loops to count down to zero rather than up to some arbitrary number. If Y is still greater than zero, the loop is repeated. Otherwise both numbers must be exactly equal, and we use **JMP noswap** to skip the swapping routine. The swapping routine itself, which we have not included here would simply swap over the four pairs of bytes, one pair at a time.

A BUBBLE SORT

We will conclude this month's episode with a sorting routine which makes use of the CMP instruction with indexed addressing. Programmers usually resort to machine code when writing sort routines, since in spite of its speed, interpreted Basic is rarely fast enough for the multitude of operations required in even the simplest of sorts. The example which we shall look at here uses the so-called **bubble sort** technique. It gets its name from the way in which the larger elements of the data array seem to drift to the top of the list as the sort proceeds. It works by repeatedly sweeping through the data, comparing adjacent items, and swapping them over if they are in the wrong order. The array is continually scanned in this way, with the scan size reducing by one item with each pass, until a complete scan is made with no swaps, indicating a perfectly sorted list. It should be said that the bubble sort is one of the slowest sort algorithms, but it is easy to implement, and acquits itself reasonably

well providing that the number of items involved is not too great.

The program given in the accompanying listing is a complete bubble sort with demonstration program. It will sort an array of up to 256 single byte integers in around 0.5 seconds on a Master 128. The base address of the array is set in line 50 to &A00. When the program is run, the code is first of all assembled, then a set of 256 random numbers is generated, and both displayed and stored in the table at &A00. The sort routine is then called, and the array is displayed again showing the result of the sort, and the time taken. For comparison purposes I tested an equivalent routine written in Basic, and it took around *200 times longer* to complete the task. In just 0.5 seconds the machine code version performs some 250 scans of the data, and around 16000 data swaps, though the precise number depends on how disorderly the data is at the start of the sort.

HOW IT WORKS

Just as with the earlier examples, the bubble sort has two main routines: one to check whether two adjacent items should be swapped or not, the other to perform the swapping where necessary. Lines 170 to 190 cope with the comparison, using BCS to branch around the swapping routine in cases where the first item is greater than or equal to the next. As you can see, we have used the Y register as an index to the table of data held from **table** to **table + tot**.

There is a small complication when performing the swapping operation: we cannot load one of the data items into the X register as we have done earlier because only LDA (and not LDX or LDY) can be used in the indexed addressing mode. This is no real problem, though. We simply use X as a temporary store for the data held in the accumulator (TAX on line 210 is a fast instruction which transfers the contents of the accumulator to the X register). The data is then reinstated in line 240 using TXA, which has the reverse effect.

Every time that a swap is made, the value 1 is stored in the location **flag** (set up in line 50 as zero page address &70). Once the array of data has been scanned (tested on line 300), the flag is checked to see whether it still contains zero. If not it means that one or more swaps occurred during the last pass, and that a new pass must be made. When finally this flag holds the value zero at the end of a pass, the sort is complete.

Next month we will take a look at further examples using the compare and branch instructions, including search routines, and the use of look-up tables.

```
 10 REM Bubble Sort
 20 REM By Lee Calcraft
 30 REM Version B 0.6
 40 :
 50 flag=&70:tot=&71:table=&A00
 60 MODE0
 70 FOR pass=0 TO 1
 80 P%=&900
 90 [
100 OPT pass*3
110 \      **BUBBLE SORT**
120 .main
130 LDY #0          \ Point to start
140 STY flag        \ Zero flag
150 :
160 .nextloc        \ Main loop
170 LDA table,Y
180 CMP table +1,Y \ Compare adj data
190 BCS noswap
200 .swap           \ Swap them
210 TAX
220 LDA table+1,Y
230 STA table,Y
240 TXA
250 STA table+1,Y
260 LDA #1
270 STA flag        \ Set flag
280 .noswap
290 INY             \ Next pair
300 CPY tot
310 BNE nextloc
320 DEC tot         \ Decrement scan
330 LDA flag        \ Check flag
340 BNE main
350 RTS
360 ]
370 NEXT
380 :
390 @%=&0305
400 FOR A=0 TO 255
410 Z=RND(255)
420 table?A=Z
430 PRINT,Z;:NEXT
440 :
450 ?tot=255
460 TIME=0:CALL &900
470 PRINT''TIME/100;" Seconds"'
480 FOR A=0 TO 255
490 PRINT,table?A;
500 NEXT
510 PRINT:@%=&90A
```

TERRAS .5



By O.R.Thomas

The planet Terras 5 is about to explode and with it the ancient mines will be lost forever. Your mission is to evacuate as many of the mining bases as possible before time runs out. Each mine covers an area of twenty-five times the size of the screen, so only a small part of it is visible at any one time.

Starting at the top left-hand corner of the mine, you must use your ship's scanner to locate each of the bases in turn, correctly orientate your ship for landing, and evacuate all personnel from the base in as little time as possible. Contact with the walls will result in death. When you have evacuated all of the bases you should then return to the entrance.

Scattered throughout the mines are a number of so-called "jump-gates", distinguished by their arrow-head entrances. These will transport your ship to a different part of the mine, but remember, it's a one-way trip only and attempting to enter through an exit gate will result in death.

Each new level will present you with further difficulties, such as increased thrust, double gravity and even triple gravity.

You control your ship using Z and X to rotate left and right respectively and Shift to thrust. Copy and Delete allow you to freeze and unfreeze the game and Q will return you to the title screen.

Points are awarded for each base visited and also for the time remaining at the completion of a level. An extra life is awarded every 50 points.

Having typed in the game, disc users should save the program before running it, as it will automatically relocate itself at &E00. The game can be made a little more challenging by altering Z% in line 160 so that it starts at a higher level of play. Alternatively you may find that the game requires a little more practice. Altering L% in line 170 will give you as many lives at you want!

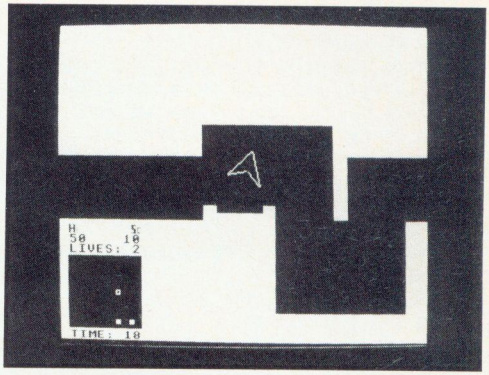
Terras 5 also works on an Archimedes with very little modification. Remove line 100 and insert a line that reads:

```
215 TEMP=TIME: REPEAT UNTIL TIME=TEMP+8
```

This will slow the game down to a playable level!

```
   10 REM Program TERRAS 5
   20 REM Version B1.0
   30 REM Author  O.R.Thomas
   40 REM BEEBUG  Jan/Feb 1988
   50 REM Program subject to copyright
   60 :
  100 IF PAGE>&E00 GOTO3080
  110 ON ERROR GOTO360
  120 DIM C%(17,1),F%(3,1,1),vf%(3,1)
  130 PROCinit
  140 REPEAT:MODE7:PROCoff:PROCtitle
  150 quit=FALSE:end=FALSE
  160 Z%=1:PROClevel:PROCmaze
  170 C%=1:X%=0:Y%=0:S%=0:T%=100:L%=4:sx
%=525:sy%=650:N%=0
  180 MODE1:PROCoff
  190 REPEAT:PROCprepare:TIME=0
  200 completed=FALSE:crash=FALSE
  210 REPEAT:PROCkeys:PROCcalc:PROCship1
:PROCoff
  220 IF T%=10 OR T%<3 SOUND 1,-15,200,1
  230 IF K% PROCcollision
  240 IF INKEY(-17) quit = TRUE
  250 IF X%=0 AND Y%=-1 completed = TRUE
  260 IF D% PROCcrash:crash=TRUE
  270 IF L%=0 end=TRUE
  280 IF TIME>420 AND T%>0 T%=T%-1:TIME=
0:PRINT TAB(7,31);T%;CHR$32;
  290 UNTIL quit OR completed OR end OR
crash
  300 IF completed MODE7:PROCoff:Z%=Z%+1
:PROClevel:PROCcomplete
  310 UNTIL quit OR end
  320 IF quit S%=0:L%=0
  330 IF end MODE 7:PROCoff:PROCend
```

```
  340 UNTIL FALSE
  350 :
  360 ON ERROR OFF:MODE7:IF ERR=17 END
  370 REPORT:PRINT" at line ";ERL:END
  380 :
 1000 DEF PROCoff
 1010 VDU 23,1,0;0;0;0;
 1020 ENDPROC
 1030 :
 1040 DEF PROCprepare
 1050 K%=FALSE:D%=FALSE:vx%=0:vy%=8:w1%=
0:R%=0:x1%=sx%:y1%=sy%
 1060 PROCscreen
 1070 VDU29,x1%;y1%;
 1080 PROCship2(0,C%)
 1090 ENDPROC
 1100 :
 1110 DEF PROCkeys
 1120 IF INKEY(-106) A%=TIME:REPEAT UNTI
L INKEY(-90):TIME=A%
 1130 IF INKEY(-98) R%=-1
 1140 IF INKEY(-67) R%=1
 1150 IF INKEY(-1) AND T%>0 vx%=vx%+(C%(
w1%,0)/V%):vy%=vy%+(C%(w1%,1)/V%):SOUND0
,-15,100,4
 1160 ENDPROC
 1170 :
 1180 DEF PROCship1
 1190 w2%=w1%:w1%=w1%+R%:R%=0
 1200 IF w1%=18 w1%=0
 1210 IF w1%=-1 w1%=17
 1220 VDU29,x2%;y2%;
 1230 IF POINT(C%((w1%+6)MOD18,0),C%((w1
%+6)MOD18,1))=3 OR POINT(C%(w1%,0),C%(w1
%,1))=3 OR POINT(C%((w1%+12)MOD18,0),C%(
(w1%+12)MOD18,1))=3 K%=TRUE
 1240 PROCship2(w1%,(C%+1)MOD2)
 1250 VDU19,((C%+1)MOD2)+1,3;0;19,C%+1,0
;0;29,x1%;y1%;
 1260 PROCship2(w2%,C%)
 1270 C%=(C%+1)MOD2:x1%=x2%:y1%=y2%
 1280 VDU29,0;0;
 1290 ENDPROC
 1300 :
 1310 DEF PROCship2(A%,E%)
 1320 GCOL3,E%+1:MOVE0,0
 1330 DRAWC%((A%+6)MOD18,0),C%((A%+6)MOD
18,1):DRAWC%(A%,0),C%(A%,1)
 1340 DRAWC%((A%+12)MOD18,0),C%((A%+12)M
OD18,1):DRAW0,0
 1350 ENDPROC
 1360 :
 1370 DEF PROCcalc
 1380 vy%=vy%-G%:x2%=x1%+vx%:y2%=y1%+vy%
 1390 IF x2%>1280 x2%=0:x1%=-100:sx%=80:
sy%=475:X%=X%+1:PROCscreen
 1400 IF x2%<0 x2%=1280:x1%=-100:sx%=120
0:sy%=475:X%=X%-1:PROCscreen
```




```
 1410 IF y2%>M% AND M%=1024 y2%=0:y1%=-1
00:sx%=525:sy%=50:Y%=Y%-1:PROCscreen:ELS
E IF y2%>M% PROCgate
 1420 IF y2%<0 y2%=1024:y1%=-100:sx%=525
:sy%=924:Y%=Y%+1:PROCscreen
 1430 ENDPROC
 1440 :
 1450 DEF PROCscreen
 1460 IF X%=0 AND Y%=-1 ENDPROC
 1470 VDU26,19,3,0;0;
 1480 GCOL0,0:COLOUR0:COLOUR131:CLS
 1490 RESTORE3040
 1500 FOR A%=1 TO 4:READ E%:FOR I%=1 TO
E%
 1510 READ J%,O%,P%,Q%
 1520 IF MID$(M$,Y%*25+X%*5+A%,1)="1" PL
OT69,J%,O%:PLOT69,J%,Q%:PLOT85,P%,O%:PLO
T85,P%,Q%
 1530 NEXT:NEXT
 1540 PLOT69,31,40:PLOT69,289,40:PLOT85,
31,280:PLOT85,289,280
 1550 PRINTTAB(1,20)CHR$227TAB(7)CHR$228
'TAB(1);H%;TAB(9-LEN(STR$(S%)));S%'TAB(1
)"LIVES: ";L%TAB(1,31)"TIME: ";T%;
 1560 IF X%=0 AND Y%=0 PLOT69,525,800:PL
OT69,100,1024:PLOT85,950,1024:IF N%<B% G
COL0,3:PLOT69,400,865:PLOT69,525,810:PLO
T85,525,920:PLOT85,650,865
 1570 GCOL0,3:VDU5
 1580 FOR A%=0 TO 4:FOR E%=0 TO 4
 1590 MOVE A%*50+50,268-(E%*50)
 1600 IF MID$(M$,E%*25+A%*5+5,1)="1" VDU
224,8,225
 1610 NEXT:NEXT
 1620 MOVE X%*50+50,268-(Y%*50)
 1630 GCOL0,1:VDU224
 1640 GCOL0,2:VDU8,225
 1650 IF MID$(M$,Y%*25+X%*5+5,1)="1" MOV
E550,446:PRINTSTRING$(5,CHR$226):MOVE550
,442:GCOL0,1:PRINTSTRING$(5,CHR$226)
 1660 IF MID$(M$,Y%*25+X%*5+5,1)="2" MOV
E710,450:PRINTSTRING$(5,CHR$127)
```

```
 1670 VDU4:GCOL0,0
 1680 M%=1024
 1690 IF MID$(M$,Y%*25+X%*5+5,1)="3" PLO
T69,550,450:PLOT69,750,450:PLOT85,550,95
0:PLOT85,750,950:GCOL0,1:DRAW650,1000:DR
AW550,950:M%=950:VDU24,0;0;1279;946;
 1700 IF MID$(M$,Y%*25+X%*5+5,1)="4" PLO
T69,450,100:PLOT69,450,600:PLOT85,550,15
0:PLOT85,650,600:PLOT85,650,100:GCOL0,1:
MOVE650,96:DRAW450,96
 1710 IF Y%<3 VDU19,3,Y%+1;0; ELSE VDU19
,3,Y%+2;0;
 1720 ENDPROC
 1730 :
 1740 DEF PROCcollision
 1750 K%=FALSE
 1760 IF X%=0 AND Y%=-1 ENDPROC
 1770 IF T%=0 L%=1:D%=TRUE:ENDPROC
 1780 IF w1%>0 OR x2%<610 OR x2%>646 OR
y2%>460 OR MID$(M$,Y%*25+X%*5+5,1)<>"1"
D%=TRUE:ENDPROC
 1790 SOUND1,1,100,30
 1800 y2%=456:PROCship1:K%=FALSE
 1810 VDU5:MOVE550,446:GCOL3,2:PRINTSTRI
NG$(5,CHR$226):MOVE550,442:GCOL3,1:PRINT
STRING$(5,CHR$226):VDU4
 1820 M$=LEFT$(M$,Y%*25+X%*5+4)+"2"+RIGH
T$(M$,(4-Y%)*25+5*(4-X%)):N%=N%+1
 1830 S%=S%+10:PRINTTAB(9-LEN(STR$(S%)),
21);S%:IF S%/50=S%DIV50 L%=L%+1:PRINTTAB
(8,22);L%
 1840 vx%=0:vy%=4
 1850 A%=TIME:REPEAT UNTIL INKEY(-1):TIM
E=A%
 1860 ENDPROC
 1870 :
 1880 DEF PROCcrash
 1890 L%=L%-1
 1900 *FX21,4
 1910 VDU19,0,7;0;29,x2%;y2%;
 1920 RESTORE3060
 1930 FOR A%=0 TO 3:FOR E%=0 TO 1:READ J
%,O%:FOR I%=0 TO 1
 1940 F%(A%,E%,I%)=C%((w1%+J%)MOD18,I%)*
O%
 1950 NEXT:NEXT:NEXT
 1960 FOR A%=0 TO 3
 1970 vf%(A%,0)=RND(ABS(vx%)+8)*-SGN(vx%
):vf%(A%,1)=RND(ABS(vy%)+8)*-SGN(vy%)
 1980 NEXT:VDU19,0,0;0;
 1990 FOR A%=-15 TO -3
 2000 GCOL3,(C%+1)MOD2+1:SOUND0,A%,246,6
 2010 FOR E%=0 TO 3
 2020 vf%(E%,1)=vf%(E%,1)-3
 2030 F%(E%,0,0)=F%(E%,0,0)+vf%(E%,0):F%
(E%,0,1)=F%(E%,0,1)+vf%(E%,1):F%(E%,1,0)
=F%(E%,1,0)+vf%(E%,0):F%(E%,1,1)=F%(E%,1
,1)+vf%(E%,1)
 2040 MOVEF%(E%,0,0),F%(E%,0,1):DRAWF%(E
%,1,0),F%(E%,1,1)
 2050 NEXT:VDU19,(C%+1)MOD2+1,3;0;19,C%+
1,0;0;
 2060 GCOL3,C%+1
 2070 FOR E%=0 TO 3
 2080 MOVEF%(E%,0,0)-vf%(E%,0),F%(E%,0,1
)-vf%(E%,1):DRAWF%(E%,1,0)-vf%(E%,0),F%(
E%,1,1)-vf%(E%,1)
 2090 NEXT:C%=(C%+1)MOD2:NEXT
 2100 ENDPROC
 2110 :
 2120 DEF PROCcomplete
 2130 S%=S%+T%DIV10*10
 2140 IF T%DIV10=0 GOTO2180
 2150 FOR A%=S%+10-T%DIV10*10 TO S% STEP
10
 2160 IF A%/50=A%DIV50 L%=L%+1
 2170 NEXT
 2180 PROCmaze
 2190 X%=0:Y%=0:sx%=525:sy%=650:N%=0:T%=
100
 2200 ENDPROC
 2210 :
 2220 DEF PROCmaze
 2230 X%=0:Y%=0:M$=STRING$(125,"0")
 2240 FOR A%=1 TO 24
 2250 E%=RND(4)
 2260 IF E%=1 AND Y%=0 OR E%=2 AND X%=4
OR E%=3 AND Y%=4 OR E%=4 AND X%=0 GOTO22
50
 2270 I%=X%:J%=Y%
 2280 IF E%=1 J%=Y%-1
 2290 IF E%=2 I%=X%+1
 2300 IF E%=3 J%=Y%+1
 2310 IF E%=4 I%=X%-1
 2320 IF VAL(MID$(M$,J%*25+I%*5+1,5))>0
REPEAT:X%=RND(5)-1:Y%=RND(5)-1:UNTIL VAL
(MID$(M$,Y%*25+X%*5+1,5))>0:GOTO2250
 2330 M$=LEFT$(M$,Y%*25+X%*5+E%-1)+"1"+R
IGHT$(M$,(4-Y%)*25+5*(4-X%)+5-E%)
 2340 X%=I%:Y%=J%:M$=LEFT$(M$,Y%*25+X%*5
+(E%+1)MOD4)+"1"+RIGHT$(M$,(4-Y%)*25+5*(
4-X%)+4-(E%+1)MOD4)
 2350 NEXT
 2360 M$="1"+RIGHT$(M$,124)
 2370 FOR A%=1 TO B%
 2380 X%=RND(5)-1:Y%=RND(5)-1
 2390 IF X%=0 AND Y%=0 OR MID$(M$,Y%*25+
X%*5+5,1)<>"0" OR MID$(M$,Y%*25+X%*5+3,1
)="1" GOTO2380
 2400 M$=LEFT$(M$,Y%*25+X%*5+4)+"1"+RIGH
T$(M$,(4-Y%)*25+5*(4-X%))
 2410 IF A%>3 GOTO2480
 2420 X%=RND(5)-1:Y%=RND(5)-1
 2430 IF MID$(M$,Y%*25+X%*5+5,1)<>"0" OR
 MID$(M$,Y%*25+X%*5+1,1)="1" GOTO2420
 2440 M$=LEFT$(M$,Y%*25+X%*5+4)+"3"+RIGH
```

```
T$(M$,(4-Y%)*25+5*(4-X%))
 2450 X%=RND(5)-1:Y%=RND(5)-1
 2460 IF MID$(M$,Y%*25+X%*5+5,1)<>"0" OR
 MID$(M$,Y%*25+X%*5+3,1)="1" GOTO2450
 2470 M$=LEFT$(M$,Y%*25+X%*5+4)+"4"+RIGH
T$(M$,(4-Y%)*25+5*(4-X%))
 2480 NEXT
 2490 ENDPROC
 2500 :
 2510 DEF PROCtitle
 2520 PRINTTAB(0,5)CHR$129;CHR$157;CHR$1
31;CHR$141;TAB(16)"TERRAS 5"'CHR$129;CHR
$157;CHR$131;CHR$141;TAB(16);"TERRAS 5"'
TAB(13)CHR$134;"by O.R.Thomas"'
 2530 PRINTTAB(4)CHR$133"The controls ar
e as follows :"''TAB(13)CHR$131;"Z - rot
ate left"'TAB(13)CHR$131;"X - rotate rig
ht"'TAB(9)CHR$131;"SHIFT - thrust"''TAB(
3)CHR$131;"COPY/DELETE - freeze/unfreeze
"'TAB(13)CHR$131;"Q - quit"
 2540 PRINT'CHR$129;CHR$157;CHR$136;CHR$
131;TAB(11)"PRESS SPACE TO PLAY"
 2550 *FX15,1
 2560 REPEAT UNTIL GET=32:CLS
 2570 ENDPROC
 2580 :
 2590 DEF PROClevel
 2600 PRINTTAB(14,9)CHR$141;CHR$131;"LEV
EL ";Z%'TAB(14)CHR$141;CHR$131;"LEVEL ";
Z%
 2610 IF Z%<4 B%=Z%+2
 2620 PRINT''TAB(15)CHR$129;B%;" Bases"
 2630 IF Z%=1 OR Z%=3 G%=1:PRINT'TAB(12)
CHR$129;"Normal Gravity"
 2640 IF Z%=2 OR Z%=4 G%=2:PRINT'TAB(12)
CHR$129;"Double Gravity"
 2650 IF Z%>4 G%=3:PRINT'TAB(12)CHR$129;
"Triple Gravity"
 2660 IF Z%<3 V%=12:PRINT'TAB(12)CHR$129
;"Normal Thrust" ELSE V%=7:PRINT'TAB(11)
CHR$129;"Increased Thrust"
 2670 ENDPROC
 2680 :
 2690 DEF PROCend
 2700 IF S%>H% PRINTTAB(11,5); ELSE PRIN
TTAB(11,10);
 2710 PRINTCHR$129;CHR$157;CHR$131;CHR$1
41;"GAME OVER";SPC(3);CHR$156'TAB(11)CHR
$129;CHR$157;CHR$131;CHR$141;"GAME OVER"
;SPC(3);CHR$156
 2720 PRINT''TAB(2)CHR$129"HIGH SCORE"TA
B(26)"YOUR SCORE"'TAB(2)CHR$131;H%;TAB(3
6-LEN(STR$(S%)));S%
 2730 IF S%>H% H%=S% ELSE GOTO2760
 2740 PRINTTAB(8,19)CHR$136;CHR$131;CHR$
141;"CONGRATULATIONS!!!"'TAB(8)CHR$136;C
HR$131;CHR$141;"CONGRATULATIONS!!!"
 2750 PRINT'TAB(2)CHR$136;CHR$131;CHR$14
1;"YOU HAVE BEATEN THE HIGH SCORE"'TAB(2
)CHR$136;CHR$131;CHR$141;"YOU HAVE BEATE
N THE HIGH SCORE"
 2760 *FX15,1
 2770 TIME=0:REPEAT UNTIL TIME>700 OR IN
KEY(-99):CLS
 2780 ENDPROC
 2790 :
 2800 DEF PROCinit
 2810 ENVELOPE1,0,0,49,0,9,1,3,63,-1,0,-
1,126,0
 2820 VDU23,224,240,144,144,240,0,0,0,0,
23,225,0,96,96,0,0,0,0,0,23,226,255,0,25
5,0,255,0,0,0,23,227,204,204,204,253,205
,205,205,0,23,228,240,192,199,244,52,52,
247,0
 2830 H%=50
 2840 FOR A%=0 TO 340 STEP 20
 2850 C%(A%/20,0)=70*SIN(RAD(A%)):C%(A%/
20,1)=70*COS(RAD(A%))
 2860 NEXT
 2870 ENDPROC
 2880 :
 2890 DEF PROCgate
 2900 COLOUR128:CLS
 2910 I%=0
 2920 FOR A%=0 TO 4:FOR E%=0 TO 4
 2930 IF MID$(M$,E%*25+A%*5+5,1)="3" AND
 (Y%>E% OR Y%=E% AND X%>=A%) I%=I%+1
 2940 NEXT:NEXT
 2950 X%=5:Y%=4:J%=0
 2960 REPEAT
 2970 X%=X%-1:IF X%=-1 X%=4:Y%=Y%-1
 2980 IF MID$(M$,Y%*25+X%*5+5,1)="4" J%=
J%+1
 2990 UNTIL J%=I%
 3000 sx%=550:sy%=224:x2%=sx%:y2%=sy%:y1
%=-100
 3010 PROCscreen
 3020 ENDPROC
 3030 :
 3040 DATA 2,400,1024,650,800,400,800,75
0,450,4,550,700,950,450,750,450,950,100,
950,100,1200,400,1000,400,1279,600,2,400
,0,650,300,550,150,800,650,2,0,400,500,6
00,500,450,750,700
 3050 :
 3060 DATA 0,1,12,1,0,1,6,1,0,0,6,1,0,0,
12,1
 3070 :
 3080 *KEY0 *TAPE|MFOR A%=0 TO TOP-PAGE
STEP4:A%!&E00=A%!PAGE:NEXT|MPAGE=&E00|MO
.|MRUN|M
 3090 *FX138,0,128
```



The Comms Spot

In response to members' enquiries, Peter Rochford devotes this month's look at the Comms scene to the vexed question of user-to-user file transfer.

We have received a good many letters and mailboxes from BEEBUG members who experience difficulties with 'user-to-user' file transfer. In simple terms, for those who are unfamiliar with this practice, it involves transferring data to another computer via the telephone line using your own computer, a modem and suitable terminal software.

It would be simple to give a step-by-step guide to user-to-user file transfer with the Beeb if all of us used the same modem and terminal software. Sadly we don't, so I will provide you with some general guidelines, and leave you to interpret what I say with reference to your own set-up.

To perform user-to-user file transfer you will naturally need your Beeb, a modem and some competent terminal software such as Command, Commsoft or Commstar. Along with this you will need a willing accomplice at the other end of the telephone line who is suitably equipped to send and receive. Both parties will need a certain amount of patience and tolerance as initially things do not always go as planned. Having said that, file transfer is really a doddle, I can assure you. Rule one is - don't panic!

The safest way to send files down the telephone is by using an error-checked method. For most people, this will mean the popular Xmodem protocol (an accepted standard for electronic data transfer) which is incorporated into most of the better Beeb terminal software, and which will allow you to transfer all types of files including View or Wordwise, Basic programs and machine code. There are other error-checked protocols such as that used by Kermit, but they are less popular and a discussion of these is best left until I can examine them in greater detail in another Comms Spot.

To begin your file transfer session, make sure you are organised first, and that everything is set up and ready. Next establish your call verbally with the other party and make sure everything is ready at that end too. If you are sending the file, set your modem and terminal software baud rate to be 1200 answer (I know this sounds wrong, but it is correct). This may also be designated as V23A or possibly 75/1200, or even 1200 Transmit. The potential receiver of the file should set his/her set-up to 1200 originate. This may be designated as V23O, or possibly 1200/75 or 1200 Receive.



Now go into the 'text terminal' mode of your terminal software. This normally allows you to make file transfers as well. Both users should then agree to go on-line (using the Command ROM for example, this would mean pressing f3). You should now hear the carrier tone (a high pitched whistle) on the telephone line. Replace your receiver and type something at the keyboard. This should appear on the screen of the other terminal, and on your screen too if ECHO ON is enabled via your terminal software.

It is worth determining in advance whether you are going to send your file directly from disc or from the terminal software buffer. If it will fit, load it into the buffer and transfer it from there as this is much faster. The other user should also elect to download the file to their buffer if possible. When all is ready press the appropriate key to upload. For Command this would be Shift-f5, and the recipient would need to press f5 to download. With this done, you

should now see messages on the screen indicating that transfer is taking place. The blocks of data will be counted off and any error messages displayed. Should there be errors, the software should take care of them by re-transmitting the corrupted block of data.

When all is complete, the recipient should save the file to disc straight away. You should now still be able to 'chat' to each other via the keyboard. Signal that you both want to return to voice and pickup the telephone receiver. Now instruct your system to go off-line and the carrier tone should cease and enable you to talk to the other user.

To receive files is just as straightforward as sending them. Just set your system up, this time for receiving as previously described, and go through the same routine but as a receiver.

You will have to use the above instructions in conjunction with the manual for your own terminal software. There should be no real problems as long as you make yourself familiar with your own system and make sure you know exactly what you are going to do before you go on-line.

Basic files transferred and then saved to disc should load and run without any problems, although the addresses (as shown by *EX) will be wrong. You can load the files from disc back into memory and then re-save them in order to correct this. Word-processor files, such as those for View or Wordwise, should again load straight in after transfer. Machine code files that are to be *RUN will need to have their load and execution addresses set correctly before use. All files may appear to be longer than they originally were, as the Xmodem protocol sends data in 128 byte blocks and pads out a block if the file is shorter than the space left in the last block.

Many problems occur with file transfer as a result of nerves. From my experience, a good many otherwise usually calm and capable people seem to go to pieces when trying to conduct a user-to-user session. This, I think, stems from a fear of appearing incompetent when the inevitable mistake occurs such as losing the line. This will happen occasionally and it is no problem as long as you have agreed beforehand who phones who back when it happens.

During any transfer of files do keep an eye on both the screen messages and the modem status lights. If you notice that the terminal software has to keep on re-transmitting the same block over and over again, then it is likely that noise on the line is corrupting the data. You will have to use your own judgement at the time, but if it lasts a while it may be worth picking up the receiver, going off-line and trying again having re-dialled to get a different exchange line. This is why I suggested that you should look at the modem lights regularly, as if the carrier light goes off, then the other party has probably picked up their phone and put their computer off-line.

File transfer is a simple process but does not always run smoothly. Like all other things in life it is made much easier through practice and experience. I could tell you some hilarious tales of my own experiences with user-to-user transfers in the early days, and if Dave Longley, the long-suffering Adrienne, and Ray Hughes are reading this, they will know just what I mean! B

POINTS ARISING..POINTS ARISING..POINTS ARISING..

TRIVIA QUIZ (BEEBUG Vol.6 No.7)

Unlike the magazine disc/tape version of the Quiz program, the listing in the magazine does not distinguish between DFS and ADFS systems, and this can result in problems if filenames of the wrong length are specified. The following amendments will prevent any filename longer than that allowed by the current filing system from being entered. Delete line 1810 and add the following lines:

```
70 A%=0:Y%=0:fs=USR(&FFDA) AND &FF
80 DFS=(fs=4):ADFS=(fs=8):CFS=(fs=1)
1810 IF ADFS L=10 ELSE IF CFS L=8 ELSE L=7
1815 F$=FNinput(L)
```

B

HINTS and tips HINTS and tips HINTS and tips HINTS and tips HINTS and tips

FAST DRIVES
Max Christian
Many disc-drive owners never bother to insert the links into the Beeb's keyboard PCB to set the drive speeds. However, the inserting of a link can be simulated using FX255. Most modern drives can function reliably on the highest speed but, if this should prove unreliable on your drive, reduce the speed according to the following table. The values following the *FX 255 set the relevant bits of the power-up configurating byte. The remaining bits affect the mode in which the computer powers up, and the action of Shift-Break. For this reason you should only use those values printed here:

*FX 255,255 - slowest speed (default).
*FX 255,239 - high speed.
*FX 255,207 - highest speed.

This is particularly relevant now that the ADFS is used widely. The speed at which languages can compile and link, or the speed at which databases can sort, can be increased substantially in this way.

To check reliability, using the chosen setting, it is suggested that you use the disc test routine published in BEEBUG Vol.3 No.5.

TWO INTO ONE
Stuart Squires
One limiting factor on the Master series computers and the BBC B+ is the small number of ROM sockets. However, the ROM sockets are in fact designed to hold up to 32K ROMs. If you have an EPROM programmer capable of blowing 32K EPROMs you can blow two 16K byte images onto one ROM. In theory this allows up to ten ROM images to be resident in a BBC B+ at the same time.

CHALLENGER FIX
Ray Charles
People who use the Opus Challenger 3 will find that quite a large number of commercial games and utilities generate an error when the disc in question is booted. This is easily rectified by issuing a *OPT 8,0 command before booting the disc.

ALLOCATING DISC SPACE
Dudley Long
If you are using the *SAVE command to allocate disc space for a file, the file size is not limited to the 64K of actual RAM. Thus to initialise a large file, say 200K in size, on an 80 track disc, just use:

```
*SAVE BIG 0 31E00
```

TIMING OUT
Paul Cuthbertson
When waiting for data to arrive at the RS423 port, it is often important to provide for the link failing. While *FX2,1 transfers GET's attention to the serial link, if the link fails, GET will hang until the next byte arrives. *FX2,1 makes INKEY look at the serial port as well, though, so we can use INKEY(100), for example, to give us a maximum delay of one second between bytes. If INKEY returns -1 then the link has failed, and the program can warn the operator or take other appropriate action.

Another interesting point is that INKEY with negative numbers still looks at the keyboard, so rudimentary control from the keys is still possible by this means when monitoring the RS423 port.

COLOUR COMPOSITE VIDEO
Way back in Vol.1 No.2 (now out of print) we published the hint below, but we believe the information is well worth repeating.

The composite video output connector provides only black and white output. However, it is possible to obtain a colour (PAL) signal by fitting a 470pf capacitor between the emitter of Q9 and the base of Q7. It may also be necessary to make link S39. This applies to the BBC model B, the BBC B+, and the Master series and will allow a colour video monitor to be connected to all machines.

MINIATURE DISC LABELS
David Lee
It is sometimes helpful to be able to know exactly which files are present on a disc without having to actually catalogue the disc on the computer.

A useful program which will fit all 31 files on an 89mm by 49mm sticky label, which can be stuck onto the disc envelope is as follows:

```
   *KEY0 |B|A|O|A|[|AS|A|@|A|[
|A3|A|R|C*CAT|B|M|C
```

This will program the function key f1 to produce the label. Put the printer 'on line' and position the label ready for printing, then press f1 with the appropriate disc in the drive. This utility should work on any Epson compatible printer. B

POSTBAG

POSTBAG

CARTRIDGE FOR THE COMPACT

For some time I have been searching for an easy way of changing ROMs in the Compact without having to open up the computer. The main requirement is for an external ZIF socket but none is available.

The best solution is the Viglen Cartridge System, which allows any ROM to be fitted externally in seconds. The cost is reasonable (£14.95 for the complete kit). The only socket you must leave empty in the micro is IC38 which is equivalent to the extension port.

T.W.Turner

WORLD BY NIGHT AND DAY

I recently downloaded this program from the BEEBUG Micronet pages (see also BEEBUG Vol.6 No.2) and would like to make some comments.

Firstly, it would be a good idea if users were to be warned about the time that the password routine takes. Several times I stopped the program run thinking that nothing was happening. It would also help if the password routine were to remind us where the password may be found in the magazines.

Once the password routine had run its course I found that I had an excellent program. However, when the program is working in automatic mode it sometimes misses a minute. This seems to be because line 1330 lets the program go on for 6000 'ticks' before it checks the clock again, and 6000 'ticks' plus the time to execute the code is just over a minute. Murphy's law ensured that the effect of this was evident when I was explaining to my wife that at the next ten minute point the map would be redrawn, and it jumped from 9:09 to 9:11 missing out the re-display!

The cure for this is to change the 6000 in line 1330 to 600, and then make the following changes:

```
225 done=TRUE
245 IF tn% MOD10>0 done=
FALSE
250 UNTIL tn% MOD10=0 AN
D NOT done:GOTO220
```

David H.Wild

We have noted Mr Wild's comments regarding passwords for our Micronet programs and we will take action where possible with any additional software. The World by Night and Day proved a very popular program from the time it was first published, so the amendments given above may well prove useful.

HIGH-RES MODE 0 DUMP

Once again the excellent BEEBUG has provided us with a superb program and article in the November issue (Vol.6 No.6) with the 'High Precision Screen Dump for Mode 0'. Unfortunately, as the article states, it does not cope very happily with text. This can be overcome by the addition of one line to the program:

```
175 IF K=31 AND loop%<>1
THEN PTR#F=PTR#F+2:REPEAT:K
=BGET#F:UNTIL K=13 OR EOF#F
```

This avoids the otherwise duplicate printing of any text, caused by reading the spooled file twice, by moving the pointer (PTR#) two places further on in the spooled file. It then simply keeps on getting a character from the file until either the end of the file is reached, or the character read is a Return, signifying the end of the printed statement.

Paul Cuthbertson.

We are pleased to publish this useful amendment to the program. We have also been asked by Design Dynamics of Bedford to point out that there is no connection between the program as published in BEEBUG and Design Dynamics' own product MODE-00 DUMP. We reviewed this in BEEBUG Vol.6 No.2, and being written in machine code it is substantially faster.

ALPHAMETICS AGAIN

Here is the solution to the alphametic in the December issue (BEEBUG Vol.6 No.7):

```
        A       2
+   MERRY   97445
+    XMAS    6928
=  TURKEY  104375
```

This solution was obtained very rapidly using a HUMAN computer!

George H.Foot

Mr.P.Barrett's program for solving this type of puzzle now makes it very easy to check that any you make up actually work. Out of various I have tried, I thought you might find the following of interest:

```
      BBC
+   MICRO
+    USER
=  BEEBUG
```

Peter Millward

We'll give the answer to this one next time around.

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

		BEEBUG & RISC USER
£ 7.50	6 months (5 issues) UK only	£23.00
£14.50	1 year (10 issues) UK, BFPO, Ch.I	£33.00
£20.00	Rest of Europe & Eire	£40.00
£25.00	Middle East	£44.00
£27.00	Americas & Africa	£48.00
£29.00	Elsewhere	

BACK ISSUE PRICES

Volume	Magazine	Cassette	5"Disc	3.5"Disc
1	£0.40	£1.00	-	-
2	£0.50	£1.00	-	-
3	£0.70	£1.50	£3.50	-
4	£0.90	£2.00	£4.00	-
5	£1.20	£2.50	£4.50	£4.50
6	£1.30	£3.00	£4.75	£4.75

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.

FURTHER DISCOUNTS

We will allow you a further discount:

Five or more:	deduct £0.50 from total
Ten or more:	deduct £1.50 from total
Twenty or more:	deduct £3.50 from total
Thirty or more:	deduct £5.00 from total
Forty or more:	deduct £7.00 from total

POST AND PACKING

Please add the cost of p&p:

Destination	First Item	Second Item
UK, BFPO + Ch.I	40p	20p
Europe + Eire	75p	45p
Elsewhere	£2	85p

BEEBUG
Dolphin Place, Holywell Hill, St.Albans,
Herts. AL1 1EX
Tel. St.Albans (0727) 40303
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 Assistant: Lance Allison
Production Assistant: Yolanda Turuelo
Membership secretary: Mandy Mileham
Editorial Consultant : Lee Calcraft
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 £40 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) 1988

Printed by Head Office Design (0782) 717161 ISSN - 0263 - 7561

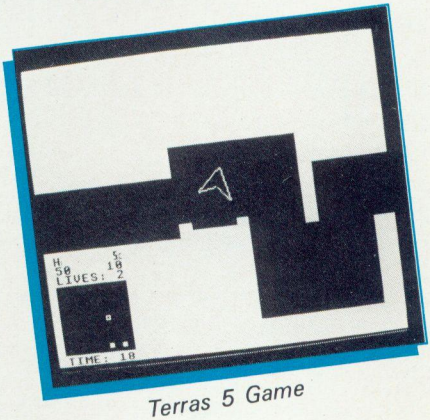
Magazine Disc/Cassette

JAN/FEB 1988 DISC/CASSETTE CONTENTS

HOPALONG - generate an infinite variety of fascinating and changing patterns.
CIRCUIT ANALYSIS - both the magazine program and an extended version to include active circuits.
TRANSPOSING AND PRINTING MUSIC - enter music via the standard keyboard for automatic transposition, display and printing.
BARRY CHRISTIE VISUALS
BEEBUG SPRITE EDITOR - all you need in one program to create multi-coloured sprites.
BEEBUG WORKSHOP
THANKS FOR THE MEMORY (Part 2) - another example of memory saving techniques in action.
TWO DFS UTILITIES - implement *FREE and *MAP for your DFS system, and a version for the Watford DDFS.
THE MASTER SERIES
TALKING TO THE ADFS - a set of routines for extracting information from the Master ADFS, all packaged up in one complete demonstration.
MACHINE CODE CROSS REFERENCER - analyse and cross reference machine code from file or ROM.
FIRST COURSE
GETTING ANIMATED - all the techniques needed for smooth animation in this short demonstration.
EXPLORING ASSEMBLER (PART 7) - complete bubble sort program demonstrating the use of CMP, CPX and CPY.
TERRAS 5 - a highly addictive and testing game of your keyboard skills.
MAGSCAN - Bibliography data for this issue of BEEBUG



BEEBUG Sprite Editor



Terras 5 Game

All this for £3 (cassette), £4.75 (5" & 3.5" disc) + 50p p&p.
Back issues (5.25" disc since Vol.3 No.1, 3.5" disc since Vol.5 No.1, tapes since Vol.1 No.10) available at the same prices.

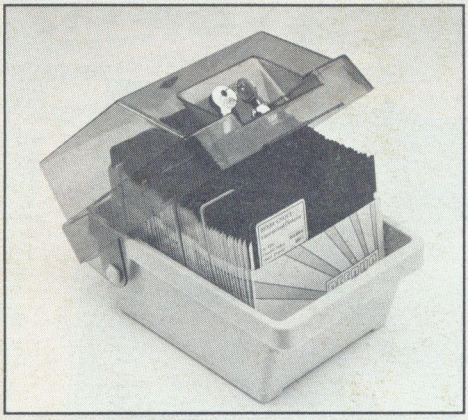
SUBSCRIPTION RATES	UK ONLY			OVERSEAS		
	5" Disc	3.5" Disc	Cassette	5" Disc	3.5" Disc	Cassette
6 months (5 issues)	£25.50	£25.50	£17.00	£30.00	£30.00	£20.00
12 months (10 issues)	£50.00	£50.00	£33.00	£56.00	£56.00	£39.00

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

Cassette subscriptions can be commuted to a 5.25" or 3.5" disc subscription on receipt of £1.70 per issue of the subscription left to run. All subscriptions and individual orders to:
BEEBUG, Dolphin Place, Holywell Hill, St.Albans, Herts. AL1 1EX.

BEEBUG Discs - The Ultimate in Quality & Reliability

PRICES REDUCED!



50 discs with free lockable storage bock

50 Single Sided 40T Double Density Discs — Only **£35.60**

50 Double Sided 80T Quad Density Discs — Only **£39.90**

Prices shown are members prices and include VAT.

BEEBUG discs are manufactured to the highest specifications and are fully guaranteed.

Our Guarantee

We confidently offer a lifetime data guarantee and will replace any disc with which you encounter problems. We have found that the standard of quality control at the factory makes this necessity very rare.

40 Track Single Sided Double Density

	Price	Members Price	Order Code
10	£9.37	£8.90	0657
25	£23.00	£21.85	0661
50	£37.50	£35.60	0665

80 Track Double Sided Quad Density

	Price	Members Price	Order Code
10	£10.42	£9.90	0660
25	£26.20	£24.90	0664
50	£42.00	£39.90	0668

Please send me _____ (qty) _____ (stock code) at £_____ (unit price)

UK post 10 £1, 25/50 £3.75. Overseas send same price inc. UK post & VAT

I enclose a cheque for £_____ /Please debit my Access/Visa card £_____

Expiry date: ☐☐

Name _____ Memb No. _____

Address _____

_____

BEEBUG
Dolphin Place,
Holywell Hill,
St. Albans,
Herts.
AL1 1EX

☎ (0727) 40303