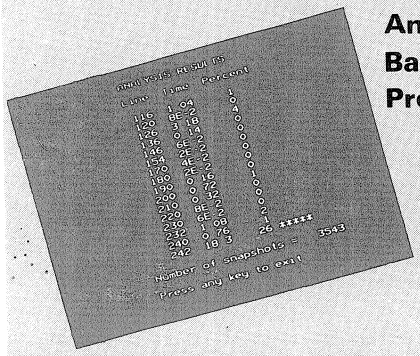
£1.20

BEEBUG

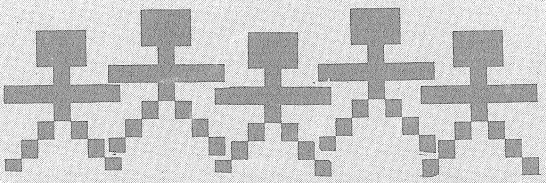
FOR THE BBC MICRO

Vol 3 No 5 OCTOBER 1984

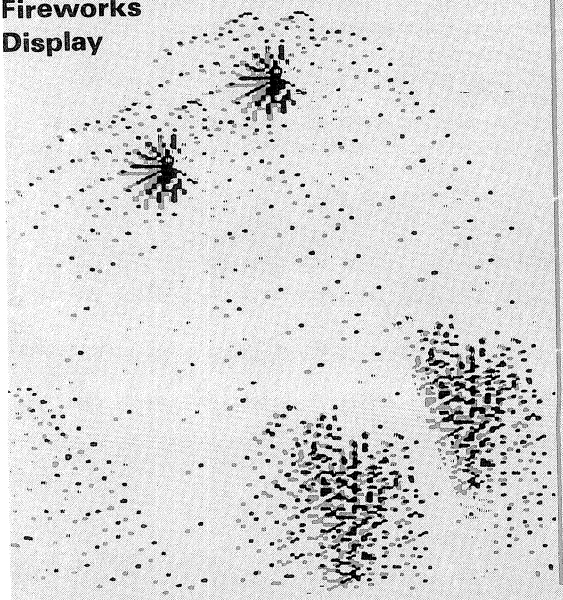


Analysing Basic Programs

Animated Graphics



Fireworks Display



- Computer products and consumer protection
- Analysing the performance of Basic programs
- Software for Acorn Z80 second processor reviewed
- Fireworks display
- Modems and bulletin boards
- Acorn Prestel adaptor reviewed
- Matching pairs game
- Kitchen Chaos game
- And much more

BRITAIN'S LARGEST COMPUTER USER GROUP
MEMBERSHIP EXCEEDS 25,000

EDITORIAL

CONSUMER PROTECTION

Amongst the many letters that we receive from BEEBUG members are those complaining of poor service in some respect from a shop or other supplier. Whilst we try to help where we can, a good appreciation of your consumer rights can be a great help. To assist readers we have commissioned an article, published in this issue, giving an expert legal assessment of your position as a consumer.

PRICE INCREASE FOR ACORN'S Z80 SECOND PROCESSOR

As we go to press we learn that Acorn have raised the price of their Z80 Second Processor package from £299 to £399, a very hefty increase which Acorn justify by saying that it will give dealers a larger profit margin and allow them to provide better customer support. You may feel that this qualifies the assessment in our review of this product in this and last month's issue. Acorn do also point out that the cost of Acorn disc drives has at the same time been reduced by £50 (single 100K drive) and £105 (double 800K drives), which in some cases may redress the balance.

BEEBUG'S ROM RULE

Various ROMs conflict when a common command is issued, for example *EDIT used by both BEEBUG's Toolkit and Watford Electronics DFS. To overcome this problem BEEBUG proposed BEEBUG's ROM Rule - here a letter can be used optionally to precede any command. More details appeared in Notice Board for the June 84 BEEBUG (Vol.3 No.2).

We are encouraging software houses to take up this idea and have had a good response so far. Computer Concepts, Dataware, Intersoft, Viglen and Watford Electronics have all agreed and been allocated a letter. Considerable interest has been shown by other ROM producers.

A - Acorn (reserved)	B - BEEBUG	C - Computer Concepts
D - Dataware	I - Intersoft	P - Pace (reserved)
V - Viglen	W - Watford	

ACORNSOFT'S VIEW FAMILY

We understand that ViewSheet (reviewed in BEEBUG Vol.3 No.3 is at last available from Vector Marketing and other dealers. In addition to View 2.1 referred to under 'Software News' in the August/September supplement, Hi-View for use with the 6502 Second Processor has also been released. This allows just over 48K of memory for text. In addition, ViewIndex (allows an author using View to compile an index) and a Printer Driver Generator package are also now available. Readers may be interested to know that we are now largely using View to produce and edit BEEBUG magazine.

BBC CHIP SHOP

A new series of the BBC Radio 4 Chip Shop is now running at 4.15pm on Saturday with a repeat at 11pm on Tuesday on BBC Radio 4 VHF. There is also a new Radio 1 Chip Shop including a weekly software Top Ten. Radio 1 is also broadcasting free software, using the Basicode format, at 5.55am on Saturdays and Sundays. An updated Basicode package is available which has been extended to cover more micros including at last the Electron. This can be obtained from BASICODE 2 +, Broadcasting Support Services, 2 Cater Road, Bristol BS13 7TW.

HINT WINNERS

This month we have selected the hints sent in by J.S.Swiszczowski and I.Malcolm as winners of £5 prizes. More hints are always welcome, including for example those for the new Acorn Second Processors.

MAGAZINE CASSETTE/DISC

All the programs from this month's magazine are included on the magazine cassette/disc together with two additional programs, the winning entry by Dave Channing in the 'Oddfactors Brainteaser' competition (see this month's supplement), and a colourful machine code arcade game called Astro Wars by Alan Malik.

BEEBUG MAGAZINE

GENERAL CONTENTS

PROGRAMS

Page Contents

HINTS, TIPS AND INFO

Page Contents

Tested on Basic I & II and O.S. 1·2

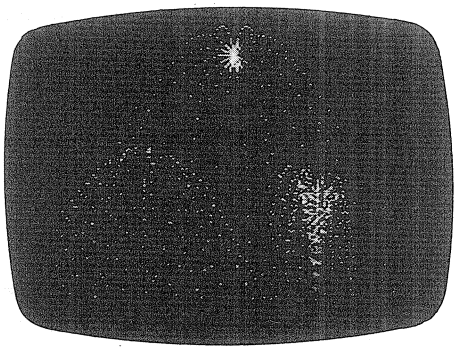
FIREWORKS DISPLAY (32k)

by D.D. Harriman

The short program listed here, by D.D. Harriman, is a good example of an animated graphics display. As such it is an excellent demonstration of the use of the VDU19 instruction described in this month's "Beginner's" article, and provides a colourful animation of a cluster of exploding fireworks.

To use the program, just type it into your Beeb, and run it. The display uses mode 2 to build up a pattern of coloured dots. After a short delay the program will then animate the display to produce a multi-coloured sequence of exploding fireworks.

The fireworks program listed here is both brief and quite simple. A short section of code (contained in the procedure PROCF) displays a series of dots on the screen in different colours to represent three different types of firework (lines 160 to 180, 190 to 210 and 220 to 260). These are then redefined in a fixed sequence (using the procedure PROCPP called repeatedly at line 270) to produce the impression of movement.

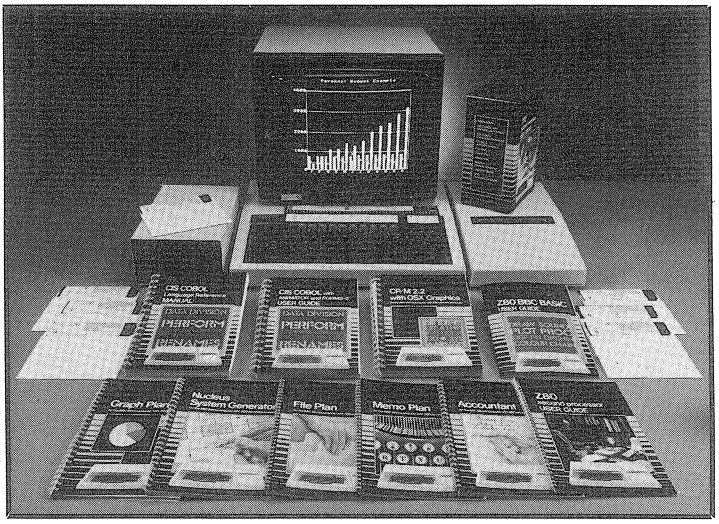


```
  10 REM PROGRAM FIREWORKS
  20 REM AUTHOR D.D. HARRIMAN
  30 REM VERSION B1.3
  40 REM BEEBUG OCTOBER 1984
  50 REM PROGRAM SUBJECT TO COPYRIGHT
  60 :
 100 ON ERROR GOTO 300
 110 MODE 2:COLOUR143:COLOUR14
 120 VDU19,15,0;0;:VDU19,14,7;0;12
 130 VDU23,1,0;0;0;0;
 140 Q%=0:PROCF(300,0,0,100,9,13,0)
 150 X2%=X1%:Y2%=Y1%:F2%=F1%
 160 FOR A=0 TO PI*2 STEP PI/8.5
 170 PROCF(X2%,Y2%,COS A*50,SIN A*50,9
,999,F2%)
 180 NEXT:Q%=1
 190 FOR A%=-8 TO 8
 200 PROCF(900,0,A%,70+RND(10),5,999,0)
 210 NEXT:F2%=0
 220 FOR A=0 TO PI*2 STEP PI/7
 230 GCOL0,F2%:MOVE600,800
 240 DRAW600+COS A*50,800+SIN A*50
 250 PROCF(600,800,COS A*50,SIN A*50,9
,999,F2%)
 260 F2%=F2%+1:NEXT
 270 REPEAT PROCPP:UNTIL INKEY0<>TRUE
 280 END
 290 :
 300 MODE 7
 310 ON ERROR OFF
 320 IF ERR<>17 REPORT:PRINT" at line
";ERL
 330 END
 340 :
1000 DEF PROCW
1010 TIME=0:REPEAT UNTIL TIME>3
1020 ENDPROC
1030 :
1040 DEF PROCPP
1050 C%=RND(7):FOR F%=0 TO 13
1060 PROCW:VDU19,F%,C%;0;19,(F%+13)MOD
14;0;0
1070 NEXT
1080 ENDPROC
1090 :
1100 DEF PROCF(X%,Y%,XM%,YM,G,N%,F%)
1110 FOR K%=1 TO N%
1120 X%=X%+XM%/1.5:Y%=Y%+YM/1.5
1130 YM=YM-G:F%=(F%+1)MOD14:GCOL0,F%
1140 PLOT69,X%,Y%
1150 IF X%>-1 AND X%<1280 AND Y%>-1 AN
D Y%<1024 NEXT ELSE K%=9999:NEXT
1160 X1%=X%:Y1%=Y%:F1%=F%
1170 ENDPROC
```

ACORN'S Z80 SECOND PROCESSOR – Part 2

Reviewed by W. D. S. C. FREEMAN

This month, Doug Freeman continues his look at Acorn's new Z80 Second Processor by considering the applications software that forms an integral part of the complete system.



THE BUSINESS SOFTWARE

There are five business packages provided with Acorn's Z80 Second Processor, and these are designed to provide an integrated environment for the small business user. In this part of the Z80 review I will be examining whether they are as good as Acorn claim. The five packages comprise a word processor, a spreadsheet with graphical output, a simple card index type database, a program generator and an accounts package.

PORTABILITY

One of the advantages in having a set of packages of this nature is that you should be able to take output from, say the spreadsheet, and incorporate this in a letter to your bank manager. Unfortunately, although possible, it is not very easy to move files about between the programs. For example, it doesn't appear particularly easy for the program generator (Nucleus) to access a file created using the card index system (File Plan).

The manuals are small, on average about 100 pages, well written, and as introductions to each package they are hard to fault. Unfortunately they only serve as introductions and do not really give sufficient technical detail to allow full use of the packages, this applied particularly to NUCLEUS. However, it is better to have a good, simple manual that gets you going than a comprehensive manual that is complete gibberish. All the packages, except Graph Plan use the function keys, on the whole very well. A spiral bound keyboard overlay, like the BBkey, is supplied.

The first three packages are from Chang Laboratories, and are roughly equivalent to the free software supplied with the Torch discpack.

MEMO PLAN - a word processing package

When using Memo Plan, the screen consists of a ruler line at the top with left and right margins and tab positions. The bottom line is for the entry of file names or new tab spacings. The line above that shows the name of the document being edited, your position in it as a percentage and the mode - i.e. right justified, unjustified, overtype or no automatic word wrap. The system uses mode 3 so that there are 22 lines available for text displayed on the screen. As with most good word processing software, documents far longer than the available memory can be produced and edited.

Like Perfect Writer (supplied with

the Torch Z80 system), Memo Plan lets you hold a number of documents in buffers so that they are available for editing at the same time. You can split the screen horizontally into two and look at any two documents together. Text can be moved from one document to another using the split screen. Every so often, updated text is written to a swap file. You can leave a session of word processing and when you restart in the same document, you start at the same place where you stopped. Should a system crash occur you only lose text entered after the last swap, though to recover a special program must be run. When testing this it did not always work with long files, so it is not a real alternative to conscientious saving every half hour or so.

Printing is done from a temporary file, and is performed as a rather slow background task. This means that it is possible to carry on editing one file whilst another is being printed (although response to rapid key presses is rather sluggish). The formatting control provided by Memo Plan is rather limited, especially when compared to the likes of Perfect Writer, View or Wordwise; this would prove a drawback to some people.

Moving around a document is performed in a fairly standard way for the Beeb. The cursor keys move the cursor up, down or left and right, and Shift and Control can be used to make this movement in terms of words, lines and paragraphs. Search, replace and global replace commands are available. Replace includes the intriguing "try it then ask me again" option. Copying, moving and deleting blocks of text are easy, with delete working on words, lines or paragraphs.

There were very few oddities with this package. The only one that really worried me was that if a line reached the right margin the last character was printed on a separate line. A glaring bug like this suggests my disc was faulty in some way. The lack of a proper file insert facility was missed but being able to take portions from other files and insert them using the split screen facility made up for it. Together with File Plan, it is possible to construct and print mailing lists quite easily. Bearing in mind the limited formatting capabilities and the slowness of its file handling I still find Memo Plan pretty good as a text editor on short documents. However I have reservations on its use as a fully blown word processor.

FILE PLAN - a card-index database

File Plan works a bit like a spreadsheet in that it assumes each record is a single line on a very big piece of paper. The normal screen has a command entry line towards the top with a block of data lines below this. A command, such as choice of a file, is selected and the appropriate number typed in. Records in the file can be displayed in two ways: as either a sequence of lines (each consisting of part or all of one record) or each record can be displayed individually with one field occupying a line on the screen. Files can be as long as available space on the disc. Records can have a length of up to 800 characters, and a file can contain up to some 32000 records.

Sorting, searching and extracting records is both easy and quite fast. A permanent copy of a sorted file can be kept by creating a list. Lists can also be user defined subsets of a file. They can be merged with other lists from the same file.

Letters specially prepared in Memo Plan can be printed from File Plan as part of a mailing list, with a file containing the names and addresses being used to fill in predetermined parts of the letter. Different information could of course be selected instead of names and addresses.

I found File Plan superior to Perfect Filer though can still only describe it as mediocre.

GRAPH PLAN - a spreadsheet package

This was a bit of a disappointment. Graph Plan is the spreadsheet part of the software, and follows the overall description of a spreadsheet given in BEEBUG Vol.3 No.2. The major exception to this is the command menu. Down the right hand side of the screen is a list of about 20 commands, each with an

associated number. Typing this number leads to a further selection of commands, until the final action has been selected - the number of commands available is quite extensive.

In Graph Plan the top few lines of the screen display the command line and summary of the sheet size plus the current cell position in the sheet. At the right of the screen is the menu. The rest of the screen contains part of the sheet plus headings for the columns and rows. The spreadsheet can have up to 1000 cells into which numbers or formulae can be entered. Unlike most other spreadsheets, Graph Plan tends to use columns or rows so that a formula automatically refers, say, to a whole column. (Note that trig functions could not be included in the formula). The number of cells to be used is defined when the spreadsheet is created. Sheets smaller than the maximum size save on recalculation time, though recalculation is not automatic, which can improve the speed of data entry. Columns and rows can be moved about or deleted and new ones inserted. It is not possible to take in data from other packages directly and consolidation is not allowed for, i.e. the results from one spreadsheet cannot be moved to another to make higher levels of analysis. To be fair this is not a common feature.

The use of numbers to select the commands within Graph Plan is not particularly helpful, and would tend to limit the appeal of the package if it were not for the graphics. These are comparable with those supplied on quite expensive packages, and allow for the production of line, bar or pie charts. It is possible to have more than one bar or line chart on the screen at once to provide a lot of information quickly.

Overall this package is fiddly to use but is as powerful as most CP/M spreadsheets, and it probably has more facilities than any spreadsheet packages available for a standard Beeb. The graphics were particularly good.

THE COMPACT SOFTWARE

Two packages from Compact Software are provided: Nucleus, an award winning program generator, and Accountant, an accountancy system written in Professional Basic by Nucleus.

ACCOUNTANT

Having only a limited amount of time, and a superficial knowledge of accountancy, I did not test this package to its limits. From my experiments, I found it easy to use, and the prompts and manual were both concise and clear.

NUCLEUS - a program generator

Nucleus seems to be the star of the bunch. A program generator works by requiring the user to select options from a series of menus and to fill in record layouts with such details as field name, length, character or numeric data etc. It can then create a set of files, file updating programs and report printing programs to be used for mailing lists, stock control, accounts etc. Accountant is itself fully compatible with Nucleus and files created by Accountant can be used in Nucleus programs. The programs generated by Nucleus are coded in Professional Basic and could be amended by the more adventurous user.

Nucleus proved to be a fairly flexible system, and I was able to get a simple program working in a couple of hours. I have only a couple of niggles with Nucleus. Dates must be typed in as DDMMYY, e.g. 211284. This is not very neat or flexible, and it would have been nicer to have a number of alternative formats. Errors in design are also difficult to correct, and require a lot of retyping. The manual is quite reasonable, and gets the user into the system quite quickly, however it does not go deep enough to support the user with more experience. All in all this is a very powerful and useful piece of software.

CONCLUSIONS

By now you will have realised that I am quite impressed with the whole of this system from Acorn.

If we compare the Beeb and Z80 with other options for business users it looks very good value for money. For instance £1499 buys the Advance computer with bundled Perfect software, but does not really equal the overall

Beeb package of quantity and quality. As far as other Z80 second processors are concerned they cost more and offer less in the way of software. The Graduate will provide some competition as it will turn the Beeb into an IBM-PC compatible machine, but software costs put it into a different price bracket. The Z80 with CP/M represents rather dated technology but they work and they work well. Although memory is limited, and some of the more sophisticated software packages are not available under CP/M, these systems still offer proven technology, and the Acorn Z80 package offers a good functional system suitable for a small business.

If you are the owner of a Beeb system with twin 80 track drives, and a printer, and you feel the need to use a computer for your business then at £300 it is difficult to see how you could go wrong. If you have less hardware, a non-standard DFS or low capacity drives then costs will be higher. For a thousand pounds there are now several good machines on the market that might be stiff competition for a complete Acorn Z80 system.

The Acorn Z80 system is at the right price, any higher price and more modern machines could begin to look attractive. It is a good product with reasonable software but is it too late?

POINTS ARISING

The gremlins have been at work again, and there are a couple of errors in BEEBUG Vol.3 No.3. The first one is in the "Freezing and Saving Screens" article. Unfortunately, the program was subject to a last minute alteration to make it appear neater in the magazine. This resulted in the lines referenced in the program being renumbered. The references to lines 345 and 435 in the text should be read as lines 350 to 380, and lines 480 to 510 respectively. These lines should be deleted in order to run the program from disc. An addition to make the program select disc is:

```
125 ds=file+&20                    370 LDA #0
350 LDX #ds AND 255                380 JSR &FFF7
360 LDY #ds DIV 256                695 c$="DISC":$ds=c$
```

When the listing to "BEEBUG plays Bach" was done, we omitted to initialise our patch to print '£' signs. The result is that all of the '£' signs in the program turned out as ` signs: very similar to an apostrophe sign, except that they lean to the left. With these typed in as any character the program functions, but the notes played are not quite correct. Changing these to '£' signs will make the program play the correct tune. A number of readers have written in asking why their version of the program pauses every so often. To cure this make sure that you have not typed any extra spaces at the end of the DATA statements.

For a quick and easy way to show up unwanted spaces in your program, type in immediate mode the following: MODE 6:VDU23,32,255,255,255,255,255,255,255,255
List the program and all the spaces will appear as white rectangles.

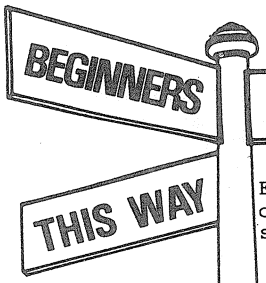
On the magazine cassette, the "Workshop" examples will not work on Basic I, as they used a semi-colon instead of a comma in the INPUT statements. This can easily be cured by changing the semi-colon to a comma, and re-running the program.

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

LIST STORED ENVELOPES - T.J.L. Young

The following function key definition will list any one of the stored envelopes on an OS 1.2 system.

```
*KEY0MO.7:INP."Number of envelope",N%:@%=0:P."ENVELOPE ";N%;:F.t%=0 TO
12:P.",";:E%=?(&8C0+(N%-1)*16+t%):IF(t%>0 AND t%<4) OR (t%>6 AND t%<11) THEN IF
E%>127 E%=E%-1:T%=E%EOR&FF:P."-";:P.T%;:N.:P.:@%=10:EL. T%=E%:P.T%;:N.:P.:@%=10|M
```



ANIMATED GRAPHICS

BEGINNERS START HERE
ANIMATED GRAPHICS (32k)

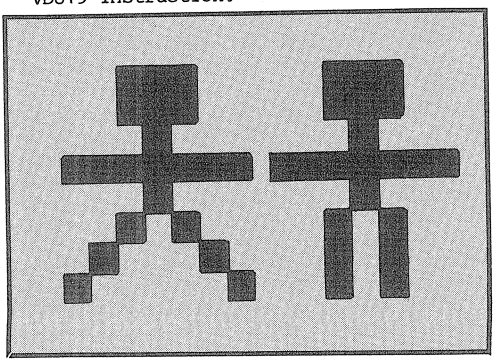
by John Wellsman

Following his introduction to the use of the VDU19 instruction for changing colour in Basic, John Wellsman shows you how to use the same powerful feature to produce fast and smooth animated graphics.

Tested on Basic I & II and O.S. 1·2

As explained in the previous discussion of the VDU19 instruction, the power and use of this command extends far beyond the simple changing of the colours relative to the mode. One of its most useful actions is to allow very fast animation, although this is at the cost of some, perhaps all colour.

The method of programming animated graphics is as follows, and this is illustrated in the first program called 'WALK'. Take a very simple case of the little (single character) man shown on pages 170/1 of the User Guide. He is defined in my program with his legs astride in line 140, and with his legs together in line 150, so as to give an illusion of walking. In order to provide as many positions as possible we use mode 2 which has sixteen relative colour values which can be defined using the VDU19 instruction.



```
10 REM Program WALK
20 REM Vesion B1.2
30 REM Author : John Wellsman
40 REM BEEBUG OCTOBER 1984
50 REM Program subject to copyright
60 :
100 MODE 7
110 ON ERROR GOTO 220
120 INPUTTAB(10,10)"Delay ",T
130 MODE 2:VDU23,1,0;0;0;0;
140 VDU23,240,28,28,8,127,8,20,34,65
150 VDU23,241,28,28,8,127,8,20,20,20
160 :
170 PROCblackall
180 PROCsetup
190 PROCwalk
200 END
210 :
220 ON ERROR OFF:MODE 7
230 REPORT:PRINT" at line ";ERL
240 END
250 :
1000 DEFPROCsetup
1010 FOR X%=0 TO 15
1020 COLOUR X%
1030 PRINTTAB(X%,10)CHR$(240+X% MOD 2)
:REM Each successive character printed
to each of the fifteen relative colour
values in this mode.
1040 NEXT X%
1050 ENDPROC
1060 :
1070 DEFPROCwalk
1080 REPEAT
1090 FOR X%=0 TO 15
1100 C%=X%:IF C%=0 THEN C%=1
1110 M=INKEY(T)
1120 VDU19,C%-1,0;0;:REM This line bla
cks out the previous character displaye
d.
1130 VDU19,C%,7;0;:REM This changes th
e relative colour value of the next cha
racter to white.
1140 NEXT X%
1150 FOR X%=15 TO 0 STEP -1
1160 C%=X%:IF C%=0 THEN C%=1
1170 M=INKEY(T)
1180 VDU19,C%+1,0;0;:REM This line bla
cks out the previous character displaye
d.
1190 VDU19,C%,7;0;:REM This changes th
e relative colour value of the next cha
racter to white.
1200 NEXT X%
1210 UNTIL FALSE
1220 ENDPROC
1230 :
1240 DEFPROCblackall
1250 FOR X%=1 TO 15
```

```
1260 VDU19,X%,0;0;:REM Every relative
colour value switched to black.
1270 NEXT X%
1280 ENDPROC
```

The procedure PROCblackall is used to define all the relative colour values except 0, the background, as 0 (or black). The procedure PROCsetup then draws alternately the two defined characters (CHR$240 & 241) progressively using each of the relative colour values. As these are all defined as black, the characters will all be invisible at this stage.

The procedure PROCwalk then re-defines each relative colour value in turn as 7 (or white) though any colour could be used (but see my warning below). At the same time, the procedure re-defines the colour of the previously displayed character as black. So at any one time, there is only one character visible on the screen but it appears, rather crudely, to be walking across the screen. The procedure includes two loops so that the man can be made to walk repeatedly backwards and forwards across the screen until Escape is pressed.

Now I mentioned in the previous paragraph that any colour could be used, but this assumes that apart from the background as black, only one other colour is being used. In the program listed here, we have used every relative colour value to provide fifteen separate positions with nothing else on the screen at all, which is a most unlikely circumstance. It is far more probable that there will be other graphics and text on the screen which will have to use some of the relative colour values, and these would also be switched on and off in moving the man across the screen. It can be seen from this that animation using the VDU19 statement requires that a number a relative values out of the fifteen available in mode 2 must be exclusively reserved for the various stages of animation, and the remaining relative colour values then used for any other text or graphics.

The example program is designed merely to show the technique and not the advantages of speed that can be obtained. In fact, a similar program which simply draws and blanks out the characters in succession is not noticeably slower. The real advantage comes when the animation or movement of large complex objects on the screen is required. If you time the operation of PROCwalk which actually does the operation, you will find that it takes about .13 secs. Using a program without VDU19 to do the same thing will take about .02 secs longer - not something that you would easily notice. But if you produce a graphics object composed of, say, four user defined characters, then without VDU19 it will take nearly twice as long, though with VDU19 it will still take about .13 secs. To move a graphic shape made up of, say, eight characters might take almost a second to draw yet the VDU19 method would still take the same time to make the shape appear or disappear. The time to change colour is always the same irrespective of the complexity of the graphics on the screen. The same time saving can be obtained when animating graphics shapes produced using MOVE, DRAW and PLOT instructions.

Animation is just one of the many uses to which VDU19 can be put. What it does, as opposed to what you make it appear to do, is to allow both the rapid and the selective change in colour of any part of the screen, and the use of black or other local background colour in printing text or graphics so that they can be hidden or displayed with no obvious delay. In addition to making things appear and disappear, it is also possible to produce fascinating colour displays which rely on frequent local changes of colour. Rippling water is a very good example and I give a simple illustration of the way in which this can be done in the second program called 'RIPPLE'. This draws a succession of short horizontal lines up the screen using colours 1, 2 and 3. The loop from line 230 to 360 then repeatedly cycles the colours so that colour 1 changes to colour 2, colour 2 to colour 3 and colour 3 back to colour 1, to give the ripple effect. The VDU 19 instructions at lines 240 to 260, 280 to 300 and 320 to 340 look more

complicated because of reference to relative and actual colour values each time.

In order to get the most out of the VDU statement, you should experiment with it first in order to acquire complete control of its effects. Study the frequent examples of its use in the programs which appear in the magazines (see the Fireworks Display in this issue). Few programs using animated colour graphics will not use VDU19.

```
  10 REM Program RIPPLE
  20 REM Version B1.1
  30 REM Author : John Wellsman
  40 REM BEEBUG OCTOBER 1984
  50 REM Program subject to copyright
  60 :
 100 MODE 2
 110 ON ERROR GOTO 410
 120 VDU23,1,0;0;0;0;
 130 colour=1
 140 REM This sets RCV 1,2 & 3 initial
ly to blue, cyan & white
 150 VDU19,1,4;0;
 160 VDU19,2,6;0;
 170 VDU19,3,7;0;
 180 FOR Y%=0 TO 1000 STEP 10:REM This
 FOR loop draws horizontal lines
 190 GCOL0,colour:REM This draws each
line in RCV 1,2 & 3 successively
 200 MOVE 100,Y%:DRAW 500,Y%
 210 colour=colour+1:IF colour=4 colou
r=1:REM Increases RCV
 220 NEXT Y%
 230 REPEAT
 240 VDU19,1,7;0;
 250 VDU19,2,4;0;
 260 VDU19,3,6;0;
 270 M%=INKEY(10)
 280 VDU19,1,6;0;
 290 VDU19,2,7;0;
 300 VDU19,3,4;0;
 310 M%=INKEY(10)
 320 VDU19,1,4;0;
 330 VDU19,2,6;0;
 340 VDU19,3,7;0;
 350 M%=INKEY(10)
 360 UNTIL FALSE
 370 REM Lines 230 to 360 progressivel
y change RCV 1,2 & 3 through blue, cyan
 and white.
 380 REM Lines 270,310 & 350 induce a
delay.
 390 END
 400 :
 410 MODE7:ON ERROR OFF
 420 IF ERR<>17 THEN REPORT:PRINT" at
line ";ERL
 430 END
```

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

WORDWISE AND SWAPPING CATALOGUES - N.J. Thorne

Wordwise loses its text on returning from Basic after executing "SWAP2" on a disc prepared with CAT2 (BEEBUG Vol. 2 No. 9). This is irritating if you wish to modify the file in one catalogue and save it in the other. So save the file on a spare disc before running the swap program. The *SWAP command in Disc Doctor can be used without loss of the file in memory.

GETTING THE RIGHT CHARACTER - J.S. Swiszczowski

The following expressions will ensure that the correct character is interpreted ignoring the state of Caps Lock and Shift Lock. This can make programs more user-friendly when the user may have selected, say, Shift-Lock mode.

Expression	Ensures correct entry of
key$=CHR$(GET OR &30)	digits
key$=CHR$(GET OR &60)	lower case
key$=CHR$(GET AND &5F)	upper case

ACORNSOFT'S AVIATOR

Reviewed by Mike Williams

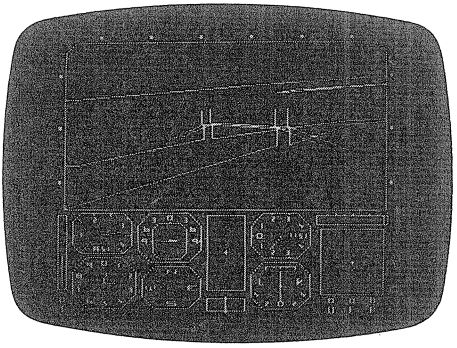
Title	: Aviator
Supplier	: Acornsoft
Price	: £14.95 (inc VAT)
Rating	: ****

Flight simulators have become a very popular feature of the games world on most home micros, and the 737 and 747 airliner simulators for the Beeb have proved no exception. Now Acornsoft have brought out something rather different in the form of a flight simulator for a World War II Spitfire.

As with other similar programs, the lower part of the screen is an instrument display, with the various instruments appropriate to a Spitfire, while the upper half at all times provides a forward cockpit view of the world outside. A map provided as part of the package and is essential for finding your way.

The landscape that you fly over (or crash into as often happened to me), is a novel feature of the whole program as it is in a three dimensional form, displaying fields, trees and mountains. There is also a river that you can follow with a bridge to fly right under when you have developed the necessary skill and panache - this scores extra points. You can score even more points by locating the town of Acornsville and flying along its streets at a frighteningly low level. Although these features are represented quite crudely, the overall impression is uncannily realistic as you see the peak of a mountain looming up in front of you, or the river twisting and yawning away beneath you.

Controlling the Spitfire is much more difficult than the typical modern airliner, and the plane is very much more sensitive to the controls; at the same time the plane is very manoeuvreable, and aerobatics are within the scope of most would-be pilots, indeed looping the loop is achieved with unintentional ease. In fact a colleague of mine found that this was the easiest way of landing back on the runway - take off and climb without changing direction until the plane is upside down - fly back over the runway in the same inverted



position before completing the loop to approach the runway again for a perfect landing! Navigation is certainly quite difficult with no radio beacons and only the crudest location finder.

There is a further feature of Aviator in the form of a game called 'The Theme', here you fly sorties in your Spitfire against waves of attacking aliens. You can choose whether or not to include 'The Theme' at the start of each flight.

I have two minor criticisms of Aviator in comparison with other flight simulators. There is no option to practice particular aspects of flying, such as take offs or landings, and there is no pause key while you go and eat your sandwiches or whatever. This is a first rate example of its kind and well worth its rating of four stars.

TESTING OUT YOUR MICRO (Part 6)
DISC DRIVES by Hugh Brown-Smith

In the final article in his series on testing out your micro, Hugh Brown-Smith turns his attention to the Beeb's disc interface and presents a program to test your disc drives.

This month I am going to deal with the testing of the disc interface and associated drive(s). The test program listed below should work on all Acorn compatible disc interfaces, that is those supporting OSWORD &7F and the 8271 commands (the 8271 is the standard Acorn disc controller chip). If you have a double density controller it is unlikely that it will work, unless the 8271 control registers are emulated correctly.

The test for the disc drive(s) consists of six parts: the first two format and verify a scratch disc. (A scratch disc is a disc which contains no files, or only unimportant ones and can safely be overwritten). The format used is to standard Acorn specification and uses non-concentric sectoring. The verify routine does not utilise the track seek command (as does the format routine), since if the disc was correctly formatted, this should not be necessary.

The next two parts are concerned with reading and writing data to the whole of the disc. First a bit pattern of Ø1Ø1Ø1Ø1 is written to the entire disc and then it is read back and checked. The test is then repeated using 1Ø1Ø1Ø1Ø so that each bit on the disc is tested in both possible states. Each of the above operations are performed a track at a time, with the current track displayed on the screen.

The final two parts test out the random access ability of your drives. On certain drives there is a known fault in the random access, which may cause problems. Your dealer should be able to correct this by means of a small hardware modification. If you have a Cumana, Teac or Viglen drive which shows this fault, the upgrade should be carried out free of charge. Note that it is not all of the drives from the above manufacturers that exhibit this fault.

To test your system, first type in the program and save it on a disc. Next insert a blank disc into the drive to be tested. It is important that the disc that you have just saved the program on is not used, as the disc used by the program will be completely erased.

Run the program and answer the questions on the screen about which drive you want to test and whether it is 4Ø or 8Ø tracks. The current test and track number being accessed is displayed on the screen while the program is running.

Should an error occur, a message will be displayed and the program will stop. Most disc errors are caused by one of two things: the first is that of damaged or corrupted discs. If the program does show an error in your system it would be as well to try it again with another scratch disc in case the fault is in the disc itself. The second problem is that of dirty heads. It is a good idea to buy a disc "head cleaning kit" and use it regularly. If you do this make sure that it is of the non-abrasive type. If errors still occur then you will have to refer the problem to your dealer.

The scratch disc used during the test can be used normally after completion, though it will need to be formatted first. It does not matter if this disc is formatted before running the program.

```
 1Ø REM Program DISC INTERFACE/DRIVE
TESTER
 2Ø REM Version B0.9
 3Ø REM Author Hugh Brown-Smith
 4Ø REM BEEBUG OCTOBER 1984
 5Ø REM Program subject to Copyright
 6Ø :
 1ØØ ON ERROR GOTO 193Ø
```

→

```
  110 DIM buff%20,fbuf%2560,comm%20:err
or=FALSE:@%=2
  120 MODE 7
  130 VDU23,1,0;0;0;0;
  140 PRINTTAB(6,5)CHR$131"TEST WHICH D
RIVE (0-3)";
  150 PROCget(47,52,55,57)
  160 drive%=A%-48
  170 PRINTdrive%
  180 PRINTTAB(0,7)CHR$134"FORTY OR EIG
HTY TRACK DRIVE (4/8) ";
  190 PROCget(51,53,55,57)
  200 PROCoscli("DRIVE "+STR$drive%)
  210 tracks%=(A%-48)*10
  220 PRINTtracks%
  230 PRINTTAB(11,10)CHR$134"FORMATTING
"SPC8
  240 TR%=1:PROCseek
  250 PROCformat(tracks%,&63):IFerror P
ROCreport(2):END
  260 PRINTTAB(11,10)CHR$131"VERIFYING"
SPC8
  270 TR%=tracks%:PROCseek
  280 TIME=0:REPEAT UNTIL TIME>30
  290 PROCformat(tracks%,&5F):IFerror P
ROCreport(3):END
  300 TR%=tracks%
  310 PRINTTAB(11,10)CHR$134"WRITE TEST
 A"SPC8
  320 char%=&AA
  330 PROCwrdata
  340 PROCwrite(&AA,&4B):IFerror PROCre
port(4):END
  350 TR%=TR%-1
  360 IFTR%<>0 GOTO340
  370 TR%=tracks%
  380 PRINTTAB(11,10)CHR$131"READ TEST
A"SPC8
  390 PROCwrite(&AA,&53):IFerror PROCre
port(5):END
  400 TR%=TR%-1
  410 IFTR%<>0 GOTO390
  420 TR%=tracks%
  430 PRINTTAB(11,10)CHR$134"WRITE TEST
 B"SPC8
  440 char%=&55
  450 PROCwrdata
  460 PROCwrite(&55,&4B):IFerror PROCre
port(4):END
  470 TR%=TR%-1
  480 IFTR%<>0 GOTO460
  490 TR%=tracks%
  500 PRINTTAB(11,10)CHR$131"READ TEST
B"SPC8
  510 PROCwrite(&55,&53):IFerror PROCre
port(5):END
  520 TR%=TR%-1
  530 IFTR%<>0 GOTO 510
  540 PROCrndacc:IF error PROCreport(6)
:END
  550 PRINT''CHR$130"  DISC TESTS COMPL
ETED AND PASSED"
  560 END
  570 :
 1000 DEF PROCget(A1,A2,A3,A4)
 1010 *FX15,1
 1020 REPEAT
 1030 A%=GET
 1040 UNTIL A%>A1 AND A%<A2 OR A%>A3 AN
D A%<A4
 1050 ENDPROC
 1060 :
 1070 DEF PROCoscli($buff%)
 1080 X%=buff% MOD 256
 1090 Y%=buff% DIV 256
 1100 CALL &FFF7
 1110 ENDPROC
 1120 :
 1130 DEF PROCformat(TR%,mode)
 1140 FORX=0TO9
 1150 fbuf%?(X*4)=TR%-1
 1160 fbuf%?(X*4+1)=0
 1170 fbuf%?(X*4+2)=X
 1180 fbuf%?(X*4+3)=1
 1190 NEXT X
 1200 comm%?0=drive%
 1210 comm%!1=-1
 1220 comm%?1=fbuf% MOD 256
 1230 comm%?2=fbuf% DIV 256
 1240 comm%?5=5+2*(mode=&5F)
 1250 comm%?6=mode
 1260 comm%?7=TR%-1
 1270 comm%?8=-21*(mode<>&5F)
 1280 comm%?9=&2A
 1290 comm%?10=0
 1300 comm%?11=16
 1310 comm%?12=0
 1320 PROCosword(&7F)
 1330 PRINTTAB(25,10)TR%-1
 1340 IF comm%?12<>0 THEN PROCreport(2)
:END
 1350 IF mode=&5F IF comm%?10<>0 THEN P
ROCreport(3):END
 1360 TR%=TR%-1
 1370 IFTR%<>0 GOTO1140
 1380 ENDPROC
 1390 :
 1400 DEF PROCseek
 1410 comm%?0=drive%
 1420 comm%!1=0
 1430 comm%?5=1
 1440 comm%?6=&69
 1450 comm%?7=TR%-1
 1460 PROCosword(&7F)
 1470 IFcomm%?8<>0 PROCreport(1):END
 1480 ENDPROC
 1490 :
 1500 DEF PROCosword(A%)
 1510 X%=comm% MOD 256
 1520 Y%=comm% DIV 256
```

→

```
 1530 CALL &FFF1
 1540 ENDPROC
 1550 :
 1560 DEF PROCreport(type)
 1570 ON type GOTO1580,1590,1600,1610,1
620,1630
 1580 PRINTCHR$129"TRACK SEEKING ERROR"
:ENDPROC
 1590 PRINTCHR$129"FORMATTING ERROR":EN
DPROC
 1600 PRINTCHR$129"VERIFY ERROR":ENDPROC
 1610 PRINTCHR$129"WRITING ERROR":ENDPR
OC
 1620 PRINTCHR$129"READING ERROR":ENDPR
OC
 1630 PRINTCHR$129"RANDOM ACCESS ERROR"
:ENDPROC
 1640 :
 1650 DEF PROCwrite(char%,mode)
 1660 comm%?0=drive%
 1670 comm%!1=-1
 1680 comm%?1=fbuf% MOD 256
 1690 comm%?2=fbuf% DIV 256
 1700 comm%?5=3
 1710 comm%?6=mode
 1720 comm%?7=TR%-1
 1730 comm%?8=0
 1740 comm%?9=&2A
 1750 PROCosword(&7F)
 1760 IFcomm%?10<>0error=TRUE:ENDPROC
 1770 PRINTTAB(25,10)TR%-1
 1780 IFmode=&53 PROCchdata
 1790 ENDPROC
 1800 :
 1810 DEF PROCwrdata
 1820 FORX%=0TO2560
 1830 fbuf%?X%=char%
 1840 NEXT X%
 1850 ENDPROC
 1860 :
 1870 DEF PROCchdata
 1880 FORX%=0TO2560
 1890 IFfbuf%?X%<>char% X%=2560:error=T
RUE
 1900 NEXT
 1910 ENDPROC
 1920 :
 1930 ON ERROR OFF:MODE 7
 1940 IF ERR<>17 REPORT:PRINT" at line
";ERL
 1950 CLOSE#0
 1960 END
 1970 :
 1980 DEF PROCrndacc
 1990 TIME=0:REPEAT UNTIL TIME>400
 2000 TR%=1:PROCseek
 2010 FORA%=0TO511STEP4
 2020 fbuf%!A%=0
 2030 NEXT
 2040 fbuf%?262=(10*tracks% DIV 256)
 2050 fbuf%?263=(10*tracks% MOD 256)
 2060 comm%?0=drive%
 2070 comm%!1=fbuf%
 2080 comm%?5=3
 2090 comm%?6=&4B
 2100 comm%?7=0
 2110 comm%?8=0
 2120 comm%?9=&22
 2130 PROCosword(&7F)
 2140 IF comm%?&A<>0 error=TRUE:PROCrep
ort(4)
 2150 TIME=0:REPEAT UNTIL TIME>400
 2160 IFtracks%=40 len=101880 ELSE len=
204280
 2170 Q%=OPENOUT(":"+CHR$(48+drive%)+".
$.TEST")
 2180 PRINTTAB(7,10)CHR$134"RANDOM ACCE
SS WRITE TEST"
 2190 FORX%=0TOlen STEP2
 2200 BPUT#Q%,&55
 2210 BPUT#Q%,&AA
 2220 NEXT
 2230 CLOSE#0
 2240 Q%=OPENUP"TEST"
 2250 PRINTTAB(7,10)CHR$134"RANDOM ACCE
SS READ TEST "
 2260 FORX%=0TOlen STEP2
 2270 IF BGET#Q% <>&55 error=TRUE:X%=len
 2280 IF BGET#Q% <>&AA error=TRUE:X%=len
 2290 NEXT
 2300 ENDPROC
```

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

CASSETTE ERRORS WITH WORDWISE - A. Nicol, J. Hitchon, D. Pickering, C.Davidson

If you have a cassette containing a Wordwise file that will not load properly due to a cassette fault, Wordwise will abort the loading process when it comes across a bad block. To get around this and allow a bad tape to be loaded type *OPT2 from the Wordwise menu and use the load command as normal. The tape will then load ignoring the errors. You must then go through the text removing the corrupted characters. If the file is badly corrupted Wordwise will not allow you to scan through it, in which case re-save it on another tape for security. Then you will have to go through the memory trying to remove the offending characters with the aid of a machine code monitor like Exmon.

BEEBUG Workshop

Tested on Basic I & II and O.S. 1.2

FLEXIBLE MENU ROUTINE

By Surac

The workshop this month presents a useful function, to use within your own Basic programs, that will display a menu of options on the screen. When an appropriate key is pressed the function returns a number corresponding to the selection made by the user.

The routine listed here is a flexible and efficient menu function that allows for the selection of up to 26 menu options on the screen, and returns the number of the option to the user's program. The function incorporates extensive traps for invalid conditions, and either exits with zero as an error flag, or takes corrective action. Coloured displays are supported by the menu, along with a double height title in all modes.

The listing given contains the function and some example data. Type it in and run it. The program will display 11 options from an imaginary menu in each of the Beeb's 8 display modes. Note that the title, and some of the options, are too long to be displayed correctly on one line of the screen in some modes, so the routine takes appropriate action. The title is simply truncated (less technically 'chopped off'), but the menu options are both truncated, to fit into the screen's width, and have a space and an asterisk appended to indicate that they have been truncated. Note that the function will, if there is room, leave a blank line between the options. This will happen in all but modes 3, 6 and 7 in the example program given.

THE PARAMETERS

The function FNmenu takes the form FNmenu(linenum%,col1%,col2%,title$) with four parameters in all. The first is the line number of a DATA statement which says how many menu options are to be displayed. These options should then follow so that successive READs will find the data. Note that if too many options are specified, the function will exit with a value of zero. The next two parameters are the foreground and background colours that the function will use when printing. Note here that the title is printed with the colours reversed. The last parameter, the string, is the title that is used when the menu is printed.

As the program is listed, the routine will display 11 options, which will appear with double spacing between the lines where possible. There are 15 data items contained within the data statements, and by altering the 11 at line 1000 to 15 you will see that the menu automatically displays the options without extra lines in all modes. By changing the 11 to 30, instead of 15, the menu will automatically exit without displaying anything. This is because it has detected that there are, according to your data statement, too many options to be displayed. You may like to expand the function to cope with situations like this, providing options for listing more options as necessary.

This workshop has been prompted by a number of readers who have sent in ideas for menu functions such as this one. Hopefully, I have combined all the ideas practical into a useful and flexible function for you to use. I would like to thank Murray Hughes, G.H. Phillips and Chris Green, who all contributed menu functions, and all the others who have written to me on other subjects. Also, a special thank you to NJMP, MB and WJH for their help.

PROGRAM NOTES

To help in an understanding of the workings of the menu function, detailed and comprehensive notes have been

included elsewhere in this issue of BEEBUG.

In summary, lines 100-190 are provided to demonstrate the use of the menu function, and the data for this is included in lines 1000-1150. The menu function itself consists of lines 2000 to 2260 and the supporting title procedure called PROCmenuhdr occupies lines 2280 to 2460.

```
   10 REM PROGRAM FNmenu
   20 REM AUTHOR SURAC
   30 REM VERSION B1.0b
   40 REM BEEBUG OCTOBER 1984
   50 REM PROGRAM SUBJECT TO COPYRIGHT
   60 :
  100 FORJJ%=0TO7
  110 MODE JJ%
  120 A%=FNmenu(1000,1,2,"Some System U
tilities")
  130 CLS
  140 PRINT"Option selected :";
  150 VDUA%+64
  160 PRINT'"Press Space:"
  170 REPEAT UNTIL GET=32
  180 NEXT
  190 END
  200 :
 1000 DATA 11
 1010 DATA Assemble file
 1020 DATA Cross reference
 1030 DATA Debug mode
 1040 DATA Printer configuration
 1050 DATA Print file
 1060 DATA Relocate module
 1070 DATA Link modules
 1080 DATA Sort data
 1090 DATA System configuration
 1100 DATA System details
 1110 DATA Edit file
 1120 DATA Create modules
 1130 DATA Restore defaults
 1140 DATA Enter system variables
 1150 DATA Exit program
 1160 :
 2000 DEF FNmenu(linenum%,coll%,col2%,t
itle$)
 2010 LOCAL A%,X%,Y%,I%,J%,A$
 2020 A%=135
 2030 !&70=USR&FFF4:J%=?&72
 2040 A%=VAL(MID$("6660660",J%+1,1))+20
 2050 VDU26,12
 2060 RESTORE linenum%
 2070 READ I%
 2080 IF I%>A% OR I%<1 VDU20:=0
 2090 PROCmenuhdr
 2100 IF A%>2*I% Y%=2 ELSE Y%=1
 2110 VDU31,0,Y%+3
 2120 FOR X%=1 TO I%
 2130 READ A$
 2140 IF J%=7 VDU31,0,Y%*X%+3,128+coll%
 2150 VDU 31,1,Y%*X%+3,X%OR64,41,32
 2160 PRINTLEFT$(A$,W%-6);
 2170 IF LENA$>W%-6 VDU&2A20;
 2180 NEXT
 2190 VDU31,0,A%+4
 2200 IF J%=7 VDU128+coll%
 2210 PRINT"Select option:";
 2220 REPEAT
 2230 A%=(GET AND &DF)-64
 2240 UNTIL A%>0 AND A%<=I%
 2250 VDUA%+64,20
 2260 =A%
 2270 :
 2280 DEF PROCmenuhdr
 2290 IF J%=7 W%=40:title$=CHR$141+CHR$
(128+col2%)+CHR$157+CHR$(128+coll%)+STR
ING$(18-LENtitle$DIV2,CHR$32)+title$:PR
INTtitle$'title$:ENDPROC
 2300 LOCAL N%,A%,X%,Y%,K%
 2310 W%=2^(2-(J%MOD3))*20
 2320 IFJ%=6W%=40
 2330 VDU28,0,1,W%-1,0,17,128+coll%,12,
17,col2%,26
 2340 IF LENtitle$>W% title$=LEFT$(titl
e$,W%)
 2350 K%=(W%-LENtitle$-1)DIV2-1
 2360 A%=&A:X%=&70:Y%=0:D%=&70
 2370 FORN%=1 TO LEN(title$)
 2380 B$=MID$(title$,N%,1)
 2390 ?D%=ASC(B$)
 2400 CALL &FFF1
 2410 VDU23,240,D%?1,D%?1,D%?2,D%?2,D%?
3,D%?3,D%?4,D%?4
 2420 VDU23,241,D%?5,D%?5,D%?6,D%?6,D%?
7,D%?7,D%?8,D%?8
 2430 VDU31,K%+N%,0,240,31,K%+N%,1,241
 2440 NEXT
 2450 VDU17,coll%,17,128+col2%,26
 2460 ENDPROC
```

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

ANOTHER ROUNDING ERROR - G.Shally

This should return 1 but it actually returns 0:

```
PRINT (40.21*100) MOD 10
```

You can investigate this further with the following two short lines:

```
PRINT INT(40.21*100)
PRINT 40.21*100-4021
```

COMPUTER PRODUCTS AND CONSUMER PROTECTION

by Richard Levene

Over the past months we have received several letters from BEEBUG members about their problems with purchases of computer hardware and software which subsequently proved to be unsatisfactory. We asked Richard Levene (a barrister by profession) to outline the legal position and your consumer rights in such matters.

INTRODUCTION

Whilst eagerly unpacking your latest brown cardboard box, you may have had some unpleasant surprises. Perhaps you were taken aback to discover that your new micro, hailed for its expansion potential and adaptability, only carried a six months guarantee! The more unfortunate might even find that their new possession fails to work properly. What can be done? What are your consumer rights?

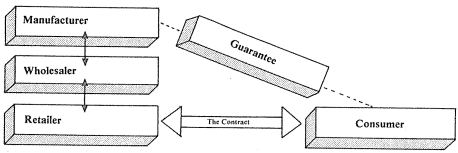
Although it cannot cover every case, this article should provide a general understanding of your legal position. Protection against faulty goods does exist and applies to micros, cassette recorders and software, just as it applies to most other purchases. We shall clear up some widespread misconceptions and hope to prevent problems from arising.

First, let us see who is and who is not covered by the statutory protection outlined below. It doesn't apply to private sales. So if you buy from a friend, or from anyone not trading in the course of a business, what follows does not assist you. The statutory protection applies to goods purchased from someone selling in the course of business. We'll use the term shop to include all such sellers, be they large chain stores, small traders, manufacturers or wholesalers.

It's interesting to note that the same protection applies to second hand goods providing these are purchased from such shops. However, much would depend on the price paid and the way the goods were described. Second hand items should still be usable, but you should otherwise expect a lower standard.

GUARANTEES

We start with manufacturers' guarantees since a widely held misconception exists about these. Although of help in some situations they are not your first remedy when a fault exists with your goods. Even some retailers seem to think that customers should contact manufacturers when faults exist. This is simply not the case (unless, of course, you purchased direct from the manufacturer).



As the diagram shows, you made your contract with the retailer when you gave him your money. In law he is responsible for the quality of the goods. Think of the manufacturer's guarantee as a secondary safeguard and focus on the retailer as your primary source of help.

THE PROTECTION

When you hand over your cash to the shop (or your flexible friend for that matter) you are entitled to goods of a certain quality. The Sale of Goods Act of 1979 says that goods must be of "merchantable quality" and "fit for their purpose". In plain language this means that they must look alright and work properly. This is so for all the goods you buy from shops. So, micros should compute, disc drives should record and transfer data, and software, of the Space Invaders ilk, should enable you to inflict genocide on

unsuspecting aliens, should that be your dishonourable desire!

Sellers cannot avoid these obligations unless your attention is drawn to a specific defect at the time of purchase. Even so, if a defect was misdescribed as superficial but is in fact more substantial, the seller could still be liable. For example, if a dent in a cassette recorder was described as a fault in appearance only, but the mechanism is also defective you would still be protected.

Furthermore, you may stipulate to the seller that you want a product to perform a particular purpose. If he advises you to purchase a particular item stating that it will perform that function and you buy it relying on that advice, then if the product is incapable of performing that function, the seller is liable. However, the seller will not be liable if it can be shown that it was unreasonable to rely on the seller's skill and judgement.

COURSE OF ACTION

It's best to avoid problems from the start, so thoroughly examine the goods at the first opportunity. Test every aspect you can think of. Check to see everything works as it should. This is important since goods must be of "merchantable quality" and "fit for their purpose" at the time of the sale. A faulty component or a weak connection may fail at a later stage but it may be because it was unreliable from the beginning.

If you discover a fault, contact the seller immediately. After all, the problem may easily be solved with a little sound advice: such as "try plugging the right lead in..."! On the other hand, if there is a defect and you notify the seller at once, you are entitled to your money back or a replacement. You may be informed that a simple repair will suffice. If you agree to this remember three points. First, you paid a fair price for the goods thus entitling you to goods in proper working order. Therefore, any costs incurred in rectifying matters are not your responsibility. Second, it is only right that the repairs be carried out promptly and efficiently since your use of the goods is running out; as is the guarantee period. Third, state clearly, in writing, that should the repair prove unsatisfactory you retain the right to have the goods replaced. Be careful, as once you agree to a repair, you lose your right to a refund or a replacement, unless this is agreed in writing with the seller.

GUARANTEES

Returning to manufacturers' guarantees, these may help if your supplier is no longer trading or is particularly unhelpful. Whatever a guarantee may say, they cannot reduce your protection by virtue of the Unfair Contract Terms Act 1977. A guarantee might claim that you are responsible for the cost of returning faulty goods. This is not so if the goods were not of "merchantable quality" at the outset. (However, if a fault develops later and is not a consequence of substandard goods from the start, then such conditions as stated in the guarantee could apply.)

Some retailers provide their own, additional guarantee. These can only increase your rights; not reduce them. One which I have seen provides for an initial 30 day period during which faulty goods will be replaced. This is followed by an eleven month period which allows for repairs. This seems a fair and reasonable attitude and it would be helpful if others offered similar terms.

One further protection applies if you use extended credit on your credit cards, such as Access or Barclaycard (but not charge cards such as American Express or Diners Club). Providing certain conditions are met (cards issued before 1977 do not provide the same protection - contact your credit card company if in doubt) you may have the same rights against the credit companies as you have against the retailer, if goods are defective. This is helpful where a shop is no longer trading or is otherwise unco-operative. The Consumer Credit Act 1974 is the relevant legislation in such cases. We can't delve into all the details here but your local Citizens Advice Bureau or Law Centre could help by pointing you in the right direction.

BE PREPARED

Of course, it's your decision where you buy goods from. I for one am extremely grateful to some of the smaller shops, for the helpful advice both before and after the sale. But wherever you go, ensure that these steps are taken. Keep receipts. Mark on them a) the name of the product; b) the date of purchase; c) the name and address of the shop. If you wish to rely on the seller's assertion that a specific function is available then ask him to put this in writing for you.

If a fault occurs, or the function is not served, return to the shop immediately and speak to the person in charge. Ask for a replacement or a refund but, if you have to leave the goods for inspection, request another receipt with the reason for the retention marked on it. For example, "Faulty keyboard to be inspected by manager". Be firm but polite and remember that the retailer should seek redress from his supplier.

CONCLUSION

These obligations are not over burdensome since if one pays a fair price for goods the least you expect is that they will work! In all trades there are those who profit by guile and cunning but the really successful business prospers by treating their customers fairly. We all respect good service and readily recommend such shops to our friends. Thus, business is increased for the retailers, whilst a fair deal is given to the providers of future income for the shops. That source of income is, of course, us - the punters!

PROGRAM NOTES FOR WORKSHOP

The notes on the workings of the menu function have been included here due to space limitations.

MAIN MENU FUNCTION

2000-2460: This is the menu function, and support procedure. These are the lines of code that you would need to incorporate into your own program. The function can be divided into a number of distinct sections, each concerned with performing a specific task.
2000: Function definition, and variable passing.
2010: Local variables.
2020,2030: J%=current screen mode.
2040: A%=number of usable lines per screen.
2050: Default window, and CLS.
2060-2080: Get number of options, and exit with 0 if too many for mode.
2090: Call procedure to display the title centred in a reverse box at the top of screen.
2100: Double line spacing?
2110: Initial cursor position.
2120,2130: Loop round, reading options.
2140: If mode 7, display teletext code.
2150: Position cursor, and do letter option, bracket and space.
2160,2170: Print the option, truncated and with an asterisk if necessary.
2180: End loop.
2190-2210: Display prompt, with teletext code if necessary.
2220-2240: Get option, with lower-upper case conversion.
2250,2260: Print option, default various VDU options, and exit.

TITLE PROCEDURE

2280: Procedure to print the title in double height, and in reverse.
2290: Display title centred, in colour in mode 7 if that is the current mode. Exit the procedure after this.
2300: Not mode 7, so local variables.
2310,2320: Calculate how many chars per line.
2330: Display reverse box at top of screen.
2340: Perform truncation of title if necessary.
2350: Calculate the start position for the centred title.
2360: Set up the variables used to expand the character definitions.
2370-2400: Get each character in turn, and read the definition.
2410,2420: Define character double size.
2430: Print the characters double size.
2440-2460: Loop, set up user's colours, cancel window and exit.

Tested on Basic I & II and O.S. 1.2

DOTTY GRID for GRAPHICS

by M. E. Williams

If you find designing screen layouts is difficult, particularly when using the instruction PRINT TAB(x,y) then this short utility will be a real help by displaying a grid of dots on the screen at the touch of a key. Now positioning all those characters is very easy.

After running the program listed below try pressing the Tab key. Instantly the screen displays a grid of dots which show actual character positions. Now you can easily count the required PRINT Tab coordinates. Press Tab again and the dots disappear. Even while another program is running the grid can be produced when required, provided modes 0, 1 or 2 are being used. You may also find this a useful aid when using one of the many graphics packages, such as ASTAAD published in BEEBUG Vol.2 Nos.7 and 9.

The program works by enabling the 'key pressed' event so that pressing any key causes a jump to the grid routine. If Tab has not been pressed the routine returns to BASIC and continues as normal. If Tab has been pressed, and you are in mode 0, 1 or 2 then the routine displays the grid on the screen. The listing has a lot of comments in it and should be self-explanatory.

Once run, the code can be *SAVEd by typing

```
*SAVE GRID D00 +60
```

and can be recalled by typing

```
*RUN GRID (or *GRID on disc)
```

Disc users will need to change the 'D00' above to 900, and similarly in the program at line 130.

You can also change the program very easily so that it responds to a key other than Tab. Just include the appropriate ASCII code in place of the '9' at line 190.

NOTE

The grid pattern is 40 dots wide, ideal for mode 1. In mode 0 there are 2 characters per 'box' and in mode 2 there are 2 'boxes' per character. The dots may affect displayed characters, so you may prefer to remove the grid before entering any characters.

```
  10REM PROGRAM Dotty Grid
  20REM VERSION B0.2
  30REM AUTHOR  M.E.Williams
  40REM BEEBUG  OCT 1984
  50REM PROGRAM SUBJECT TO COPYRIGHT
  60:
 100MODE1
 110screen=&70:REM routine uses &70 an
d &71 in zero page.
 120FOR PASS = 0 TO 2 STEP 2
 130P%=&D00:REM choose your own place.
 140[ OPT PASS
 150.code%  LDA #(code%+18) MOD 256:ST
A &220 \ set event vector.
 160LDA #(code%+18) DIV 256:STA &221
 170LDA #14:LDX #2:JSR &FFF4 \ enable
keyboard event
 180RTS
 190CPY #9 \ number of TAB
 200BNE home \ TAB not pressed.
 210LDY &355:CPY #3:BPL home \ Modes 3
 or above
 220LDA bytes,Y
 230STA e+1 \ adjust dots to suit Mode
 240LDA #&32:STA screen+1
 250LDA #&88:STA screen
 260.j1
 270LDY #&00 \ loops to print grid.
 280.j2:TYA:SEC
 290SBC #&10 \ dot spacing.
 300TAY:LDA (screen),Y
 310.e
 320EOR #1 \ dot on or off.
 330STA (screen),Y
 340CPY #&00:BNE j2
 350INC screen+1
 360LDA screen+1
 370CMP #&80 \ reached screen bottom?
 380BNE j1:LDA #8
 390JMP &FFEE
 400.home RTS
 410.bytes BRK:]
 420NEXT PASS
 430!bytes=&00151101 :REM these values
give suitable dots for each mode.
 440:
 450CALLcode%
 460END
```

MODEMS AND BULLETIN BOARDS

by James Fletcher

Making your Beeb communicate - just how practicable is it? In his latest contribution, James Fletcher throws some light onto this important but perhaps confusing subject.

All the popular "non-technical" computer magazines and television programmes have been telling us for a long time that it is child's play to make our computers talk to each other over the telephone lines, and many business users have grown accustomed to using their office computer terminal to communicate with a mainframe computer many miles away, sometimes even on the other side of the Atlantic. The advantages are legion. As well as the purely "fun" side of getting your computer to talk to that of your friend in the next town, it would be great to avoid postal delays by sending written messages over the phone, and think of the trouble it would save to be able to send Wordwise files directly to the BEEBUG editor.

This article takes a look at the reality of the current position of inter-computer communications. It tells the tale of one man's successful struggle against his own ignorance, and the horrendous jargon that surrounds every inch of the communication minefield. Take heart though, it can be done!

A quick browse through the appropriate section of the User Guide might suggest that all you need to get your machine talking to that of a friend, is to connect a cable from the RS423 output of your machine to the phone line, and get him to connect his phone line to the RS423 input socket of his machine. Forget it, it won't work, and it can't work as simply as that. Firstly, it is not permitted to connect your computer directly to BT's precious lines because of the damage to men and telephone equipment that some fault on your computer could unwittingly cause. Secondly, if signals are to travel down a standard telephone line, they must be within the audio bandwidth that the line can cope with, so it is necessary to convert the "ones" and "noughts" that your computer produces as its output into audio tones of specified frequencies that can travel comfortably along BT's cables.

The black box which does this conversion is known as a MODEM, a shortened form of MOdulator/DEModulator. One common standard, known as V21, is intended for operation with data being transferred at a speed of 300 bits/second, usually called '300 Baud'. This data rate is slow by commercial computing standards, but is adequate for many home computer communications, allowing around 400 words a minute to be received. For some years now computer enthusiasts throughout the world (but especially in the USA where telephone charges are low) have been using this 300-Baud system to get their home computers to communicate with each other. There are dozens of telephone numbers which you can ring to connect you to 'bulletin boards' - electronic magazines run by and for enthusiasts and computer clubs. In addition, some of the major component suppliers (firms like Maplin and REW) have their catalogues available on this system, so you can use your computer to see what is on offer and then you can, if you have been previously registered with their computer, order components, paying for them with a credit card.

Unfortunately the 300-Baud system is by no means the only system used for telephone-based computer communications and another popular system is based on the Prestel standard of transmitting from your computer at 75 Baud (slow, but OK for home typing) and receiving data at 1200 Baud. The Micronet 800 computer database (described last month) uses this system. Another

popular method of user-to-user communications uses 1200 Baud for both transmission and reception. All this means that if you want to make the fullest use of the various communications networks you will need a multi-standard modem, and these are available from around £100. If you buy a modem only for Micronet you will be cut-off from the great band of 300-Baud bulletin-board users, and if you buy only a cheap 300-Baud modem you won't be able to make use of Prestel or Micronet.

Let's assume then you have acquired a modem - what do you do with it? The first thing that you will have to do is to get it connected to your telephone line. This must be done by BT and these days they usually prefer to install one or more of the new-style telephone sockets in your home. Your modem, fitted with the appropriate plug, can then simply be plugged into one of these sockets.

If you are reluctant to make changes to your telephone system it is possible to buy a special sort of modem which is connected to an acoustic coupler and which requires no wiring to the telephone system. An acoustic coupler has two holes into which your telephone handset fits tightly, and by using a suitable microphone and miniature speaker it sends the appropriate tones directly through your handset. This system can work well, and acoustic couplers are available for Micronet/Prestel and for the 300-Baud systems, although I have not seen any multi-standard ones. [NOTE: Most acoustic couplers will not work with 'trim phone' type handsets.]

You can use them to communicate with the whole range of available databases, but it is important to note that they are not usually suitable for connecting two microcomputers together directly. I discovered this after hours of fruitless effort trying to achieve Beeb to Beeb communication. I found by talking to "experts" that this is because they are 'originate-only' devices and cannot cope with the interchanging of frequencies that is required if each micro is to hear the correct tones for transmit and receive.

Apart from this limitation, acoustic couplers provide a very simple and satisfactory method of getting into computer communications. With a cheap 300-Baud acoustic coupler I have been able to use my Beeb to communicate with my works computer database many miles away. All this first started going on at the same time that the film "War-Games" was putting naughty ideas into the heads of computer freaks. So lest you get any wrong ideas I must stress that the computer database in question is actually designed for remote access, and I was only able to obtain access to the files which my password permitted. I wasn't able to persuade it to pay two salary cheques into my account each month!

Having achieved one success I looked around for other places to contact, and the columns of various computer magazines (sorry Editor, not BEEBUG!) revealed a whole string of different telephone numbers, although the usual scientific law decreed that none were local to me, and all would involve trunk calls. Phone calls to Liverpool, Manchester, Birmingham and Hull revealed a whole network of 'bulletin-boards' and I was soon sending messages to other computer enthusiasts, the cost of the telephone call being the limiting factor!

SOME USEFUL PHONE NUMBERS.
All full duplex.

Distel.....01-679-1888
(Display Electronics)
Maplin.....0702-552-941
Rewtel.....0277-232-628
(Radio and Electronics World)
Mailbox 80 Liverpool..051-428-8924
Forum 80 Hull.....0482-859169
Forum 80 London.....01-902-2546
TBBS London.....01-348-9400
Southern BBS.....0243-511077
Computer Answers
Magazine..01-631-3076

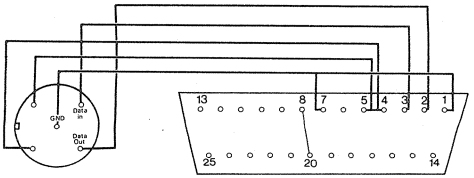
It would be nice to say "just plug in your modem and away you go" but as with all computer applications, some form of software is required so that

your computer can understand the data that is coming in and going out of its RS423 socket. Different systems use different numbers of bits to send their messages and there are various different forms of error-checking. Different systems use the so-called 'full' or 'half-duplex' modes, which means that they both send and receive data simultaneously or they can only deal with communication in one direction at a time. Most modem manufacturers provide details of suitable software, and Micronet provides all that is needed to use its system with the BBC on cassette or, more conveniently in ROM.

If you are intending to use a multi-standard modem to investigate communications with all sorts of different computers, there is a lot to be said for obtaining one of the ROM-based communications software packages that are now available. As an example, the Computer Concepts 'TERMI' ROM enables you to do everything that you could possibly want in the communications field, and even allows you to configure your BBC micro so that a mainframe computer will regard it as just another of its own 'slave' terminals. 'TERMI' appears fairly complicated to use at first, but a little practice enables you to build up a number of custom-built programs that can be used to give instant access to every type of database. The instruction book is good and clear and actually provides a good deal of general information about the use of home computers as remote terminals.

Most professional-type modems use a D-type connector, rather like a larger version of the BBC's 'analogue input' socket, and you will need a lead to connect this to a five-pin 'domino' connector suitable for the BBC's RS423 socket.

There were no modems specifically designed for the home user until the low-cost 'Prism' units became available



BBC – MODEM CONNECTING LEAD DETAILS

for Micronet users, but the 300/300 Baud types have been around for a long time and can readily be bought on the 'surplus' market. Companies like 'Display Electronics' usually have a fair selection of different types, secondhand but working. Tantel have a selection of "intelligent" modems which come with a good deal of built-in software which can make the units "user-friendly", permitting automatic dialling and answering. Several other makers are now coming forward with the very desirable multi-standard modems, and if you want to really 'dabble' these are the modems for you! It is possible to build your own, using either a complete kit of parts (Maplin) or a single-chip which carries out all the basic modem functions. Some of the home-construction magazines have recently carried articles about the building of both modems and acoustic couplers, but most of the designs which I have seen have been a lot less sophisticated than the modern ready-built units which are on sale. [We shall be reviewing modems in a future issue of BEEBUG -Ed.]

Don't be put off, though, if you can't afford an all-singing, all-dancing model: you can still get a great deal of pleasure from a cheap, single-standard unit. Modems are fun, and open up a whole new world for computer enthusiasts - get out there and communicate!

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

BBC CLOCK IMPRESSION - R. Angus

The following program will enable a BBC or Electron to do an impression of a clock. It uses the cassette motor relay to produce the ticking noise.

```
10 REPEAT TIME=0:*M.1
20 REPEAT UNTIL TIME>100:TIME=0:*M.0
30 REPEAT UNTIL TIME>100:UNTIL 0
```

Tested on Basic I & II and O.S. 1.2

A VERSATILE MOVE DOWN ROUTINE FOR DISC SYSTEMS (DFS)

by Alan Webster

In BEEBUG Vol.3 No.1 we included a useful 'movedown' routine on the magazine cassette, primarily for the benefit of disc users. We have now updated and improved this very useful utility to provide automatic loading and relocation of both Basic and machine code programs.

One of the problems that disc users of the Beeb encounter is the additional loss of some 2.75k of memory as disc work space over and above the memory already lost on a cassette system. This means that many programs that have sufficient space to run on a cassette only system will not run on a normal disc system, and that disc users are more limited in the size of program that they themselves can write.

There is a solution to this problem, as the extra workspace is often only used while loading the required program from disc, and that is to move the program down in memory after the program has been loaded. This can be done by typing in appropriate instructions at the time or, more often, appending a so called 'movedown' routine to the front of the program to be loaded. We last described this process in conjunction with the DOMINOES program in BEEBUG Vol.3 No.1.

Although this will work quite well, it is somewhat tedious, and it is easy to make a mistake and waste time. This short program will do all the necessary work for you with the result that any program you choose will be loaded from disc and then moved down in memory, byte by byte, until it occupies the memory it would have used if it had been loaded on a cassette system. Once relocated, the 'movedown' routine then starts the program rumnning.

The 'movedown' routine is very easy to use. Just type the program in and save it to disc. When you want to use the routine, simply CHAIN to it(CHAIN "MOVED") and when asked type in the name of the program to be run. The 'movedown' routine will automatically work out whether your program is in Basic or machine code and deal with it accordingly.

NOTE: to work correctly, the 'movedown' routine selects cassette as the current filing system as well as changing the value of PAGE - the start of user memory. Pressing Break after using the 'movedown' routine will reset the micro as a standard disc system, and any 'moved-down' program may be left in a corrupted state as a result.

PROGRAM NOTES

The 'movedown' routine uses the OSFILE call with A=5 (see User Guide page 454) to read the information in the disc catalogue for the selected program, and tests whether this program is in Basic or machine code. If it is in Basic, then it is moved down to &E00 (the default setting of PAGE on a cassette system) and then run. If it is a machine code program, it is moved down to its load address and then run by calling the execution address.

```
 10 REM PROGRAM MOVEDOWN
 20 REM AUTHOR  ALAN WEBSTER
 30 REM VERSION B1.1
 40 REM BEEBUG  OCT 1984
 50 REM PROGRAM SUBJECT TO COPYRIGHT
 60:
100 MODE 7
110 IF PAGE=&7000 GOTO 160
120 *K.0FORA%=0TO(TOP-PAGE)S.4:A%!&70
00=A%!PAGE:N.|MPAGE=&7000|MOLD|MRUN|M
130 *FX138,0,128
140 END
150:
160 MODE 7
170 PRINT''CHR$134;"Filename";CHR$129
;":";
180 INPUT""filename$
190 PROCgetinfo
200 PROCload
210 IF bas% PROCbasic ELSE PROCmcode
220 END
230:
```

```
1000 DEF PROCgetinfo
1010 A%=5:X%=&A0:Y%=&7B
1020 F%=&7BE0:$F%=filename$
1030 !&7BA0=F%:CALL &FFDD
1040 load$=STR$~(!&7BA2 AND &FFFF)
1050 ld%=EVAL("&"+load$)
1060 exec$=STR$~(!&7BA6 AND &FFFF)
1070 ex%=EVAL("&"+exec$)
1080 lenth%=!&7BAA AND &FFFF
1090 IF ex%<32767 bas%=0 ELSE bas%=-1
1100 ENDPROC
1110:
1120 DEF PROCbasic
1130 FORS%=0TOlenth%STEP4
1140 S%!&E00=S%!&1900:NEXT
1150 *KEY0 PAGE=&E00|MOLD|MRUN|M
1160 *FX138,0,128
1170 ENDPROC
1180:
1190 DEF PROCmcode
1200 FORS%=0TOlenth%STEP4
1210 S%!ld%=S%!&1900:NEXT
1220 *K.0 CA.ex%|M
1230 *FX138,0,128
1240 ENDPROC
1250:
1260 DEF PROCload
1270 G$="LOAD "+filename$+" 1900"+CHR$
13
1280 $&7B80=G$
1290 X%=&80:Y%=&7B:CALL &FFF7
1300 *TAPE
1310 ENDPROC
```

COMPACT FUNCTION KEY DEFINITIONS UPDATED

by P. J. LeSueur & J. P. Jakubovics

The article "Compact Function Key Definitions" in the March issue of BEEBUG (Vol.2, No.9, p.34) outlined a direct method of using less memory space for function key definitions (see also the article, "Function Key Editor", BEEBUG Vol.3, No.1, p.20). This was done by substituting a tokenised Basic keyword for the more normal ASCII form. The example given to illustrate this was the equivalence of

```
*KEY0 |!L|M

*KEY0 RENUMBER|M
```

- as explained before, there being an obvious gain to be had from the former.

Each of the keywords previously listed can be abbreviated by the use of |! followed by a printable character. All the remaining keywords can also be abbreviated, though by two rather than one byte through the use of control characters as shown in the table below. For convenience, some keywords have been repeated from the previous table with a further two previously omitted.

OPT is not tokenised, but stored in ASCII, TOP is stored as a tokenised TO followed by the ASCII for P. In Basic I, OPENIN has a token |!-

KEYWORD	TOKEN	KEYWORD	TOKEN
ABS	|!|T	LINE	|!|F
ACS	|!|U	LOMEM(left)	|!|R
ADVAL	|!|V	LOMEM(right)	|!R
ASC	|!|W	MOD	|!|C
ASN	|!|X	OFF	|!|G
ATN	|!|Y	OPENIN	|!|N
BGET	|!|Z	OPENUP	|!-
COS	|!|[	OR	|!|D
COUNT	|!|\	OSCLI	|!|?
DEF	|!\]	PAGE(left)	|!|P
DEG	|!|]	PAGE(right)	|!P
DIV	|!|A	PTR(left)	|!|O
ELSE	|!|K	PTR(right)	|!O
EOR	|!|B	SPC	|!|I
ERL	|!|^	STEP	|!|H
ERR	|!|_	TAB(	|!|J
ERROR	|!|E	THEN	|!|L
HIMEM(left)	|!|S	TIME(left)	|!|Q
HIMEM(right)	|!S	TIME(right)	|!Q
		VPOS	|!<

ACORN'S PRESTEL ADAPTOR

Reviewed by Sheridan Williams

We have been waiting for some time now for the release of Acorn's Prestel adaptor as it promised to offer good value for money. Sheridan Williams assesses whether paying £113.85 (inc. VAT) for Acorn's adaptor, instead of around £50 for an acoustic coupler to access Prestel, is really as expensive as it seems.

As we have previously mentioned in BEEBUG Vol.3 No.4, accessing Prestel via a normal acoustic coupler is not as reliable as it could be, this is due to the nature of the link. When using the Prestel adaptor it is very rare to find a corrupted frame, indeed I have never needed to re-dial and find frame reliability around 98%. This convenience alone has increased my use of Prestel considerably.

THE PACKAGE

The Prestel adaptor comes in the now "standard box" used for most of Acorn's other add ons for the Beeb, e.g. second processors etc. You also get a 60 page User Guide, a key strip, and an EPROM to add to your ever growing collection. If you have done without a ROM extension board so far, this may mean adding the cost of such a board to the cost of the adaptor. To use the Prestel adaptor, first plug in this EPROM and place the keystrip above the function keys. Next plug the adaptor into the mains, Beeb and telephone socket, and you are away. The contents of the keystrip are repeated at the bottom of the screen when initially entered, and they can be called up at any time.

The EPROM sets up the function keys to provide 15 very useful features, these are:
CALL, LEAVE, CONFIG, PAUSE, SEND, HOLD, SAVE PROGRAM, HELP, OS COMMAND, MENU, DOWNLOAD PROGRAM, LOAD FRAME, EDIT FRAME, SAVE FRAME, PRINT FRAME

The first normally used is the CALL function key, which prompts you to type in a phone number. The unit then dials the number for you and you can hear the sequence through the adaptor's built in speaker. The speaker remains active until a contact is made, this is useful because it allows you to hear any engaged signals that there might be.

It is a pity that the leads supplied were so short: one can accept short data and mains leads if the machine is to be sited next to the computer, but one cannot guarantee that there will be a handy telephone socket. So add to your costs British Telecom's charge for installing another socket, unless the 2 metre lead is adequate for your needs. (The telephone lead comes with the new-type plastic plug connection.) The unit connects to the computer through the RS423 interface, so those who use a serial printer cannot print Prestel pages (although you could save the screens to disc or cassette, and subsequently print them).

PRINTING FRAMES

As the unit comes you can print any frame, but teletext graphics characters are replaced with asterisks. This can be changed, but because the method of printing is 'printer dependent' you will need to do the patching yourself. The printing characteristics are set using CONFIG. To dump a screen exactly as it appears on the screen, the printer must be capable of either printing teletext characters, or of defining them.

SAVING, LOADING, AND EDITING

Downloading telesoftware could not be easier: the operation is fully menu driven, with prompts all the way. Programs can be saved once loaded. In fact all frames and software can be saved under their default page reference, or the files can be given another name. Loading and editing frames is equally simple.

TELEX

One of the main uses for our office system is Telex. Telex terminals are normally expensive and can only really be used for Telex purposes. Thus it is particularly useful for one of the

office machines to double as a Telex terminal when needed. Telex frames may be prepared off-line using the Save, Load and Edit Frame features, thus saving on both telephone and Prestel computer time.

AUTO START

It is convenient that using a !BOOT file the unit will auto-dial and automatically supply everything that you normally type to enter Prestel, including the Prestel (or Micronet etc) telephone number, your user code and your password. These are not displayed on the screen, so remain secret to the casual viewer.

CONCLUSION

It is a shame that the Prestel adaptor is only a 1200/75 baud modem, as this makes it less versatile than others which allow an additional 300/300 baud mode. It should be mentioned that the Prestel adaptor meets BT approval.

Acorn's Prestel adaptor can be recommended for its software alone; the hardware is reasonable value for money, but one can buy a more versatile modem for not much more - however this is unlikely to include such good software.

POSTBAG POSTBAG POSTBAG POSTBAG POSTBAG POSTBAG

Dear Sir,

I wrote to you recently concerning a problem I was having with the Disc version of Masterfile. In your letter you suggested that the problem may be due to either a faulty disc or a faulty drive. You further suggested that I return the disc and you would send me a disc that you had previously tried and tested. Bearing in mind that the problem only occurred after the computer had been on for some time seemed to rule out the possibility that the problem was software rather than hardware. Soon after I received your letter I read an article in Acorn User about a problem that seemed similar to mine and which was caused by a power feed problem to the Sideways ROM board. I carried out all the checks mentioned in the article and while the problem was less severe afterwards, none the less it was still there.

I am fortunate that as a service engineer working in Cambridge I get to visit the Cambridge University Computing section. Coincidentally this is where Dr. Andy Hopper works. But by visiting the Electronic Workshops there I discovered that a fault on one machine that they had repaired had been caused by IC14. This had been giving the same problem that I was complaining of, namely the computer thought that there was a fault in the program. I have now replaced this IC and despite trying to simulate the problem by using a hair drier I have been unsuccessful. I have concluded therefore that the problem has been fixed. It was pointed out that on later machines this IC has been fitted into a holder whereas on my Issue 3 board it has been soldered in, pointing perhaps to there being a problem in this area. I have now fitted a socket and a new chip and as mentioned it seems to have done the trick.

I hope this information is of some use to you or a member.

Regards

Mr C. Walker from Wymondham, Norfolk.

Thanks for your letter Mr.Walker. We are always interested to know that matters have reached a successful conclusion.

Dear Sir,

Here are some short improvements on the Grand Prix game that appeared in BEEBUG June 1984.

```
380 IF P%(0)>0 A%(0)=0
390 IF P%(1)>0 A%(1)=0
395 UNTIL L%(0)>TL% OR L%(1)>=TL%
2085 VDU 19,1,0;0;
2135 VDU 19,1,2;0;
```

And delete the following lines: 470, 510, 520, 1620 - 1720

These few changes make the screen "lighten" immediately before the race is to start. It also has the effect that the game doesn't end if you hit a wall, you just loose speed and you have to start accelerating again. We think it's much more fun playing the game, and you don't have to risk crashing in the last curve at lap 19!

Yours faithfully

H.Andersson & J.van Egmond. Sweden

Tested on Basic I & II and O.S. 1.2

MATCHING PAIRS (16k)

by Michael Quinion

Pairs is a version of the old card game where you have to turn over the cards one at a time, and match up pairs. The game makes good use of the teletext graphics to display the board, and features three levels of difficulty.

In Pairs, you are presented with a 7 by 7 board of dots. Behind each is a letter, which you can briefly reveal by moving the cursor (with the cursor keys) to that position and pressing 'W'. There are 24 pairs of letters on the board and one odd one. The object of the game is to find and match all 24 pairs and so find the odd letter within a limited period of time.

You match a pair of letters, once you have found them, by moving the cursor to the first and pressing 'Q', and then moving it to the other matching letter, and pressing 'Q' again.

To make things more interesting, you have to work against a timer that counts down from 20 to zero. If you match a pair before the clock reaches zero, you get twice the clock's reading added to your score. Once the clock has ticked down to zero, you will get no points for locating the next pair, but you must still match them in order to restart the clock and gain further points on finding subsequent pairs.

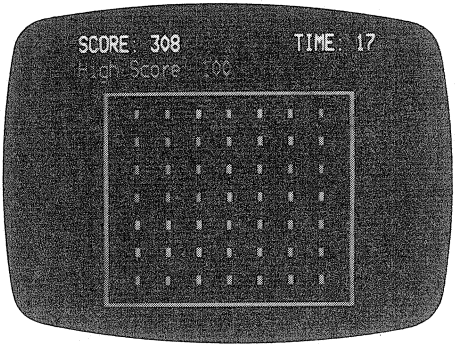
You lose points for trying to reveal a letter already matched, or for trying to match two letters which are not a pair.

LEVELS OF DIFFICULTY

There are three levels of difficulty, and this affects how many times a given letter is repeated: either 8 times, 4 times, or just twice. Level one is not too difficult, but level three takes some doing!

OMITTING THE INSTRUCTIONS

The instructions for playing the game are included in the program listing. If you don't want to type them in, you should leave out lines 2160 to 2570 (PROCinsth and PROCinst) and remove the reference to PROCinst in line 130. Keep lines 2580 and 2700, however, or you will get irrelevant messages when you run the program!



This is because the data statements in lines 2580-2700 are read into a set of arrays (in line 1960), which give the X and Y co-ordinates of the start of each message, its colour and its text. Each is printed as required by PROCdisplay (lines 1550-1570) which only needs the number of the relevant message to be passed to it.

PROGRAM NOTES

If you check PROCmove (lines 1170-1210), you will find that the cursor position is kept in the variable T%, which can take values from 0 to 48. The row and column position at any moment are derived from this using MOD7 and DIV7 operations respectively. This may seem a devious way of doing things, but it actually saves about 300 bytes of coding compared with the more obvious X,Y co-ordinate method.

If you think 20 seconds is too generous, or otherwise, change the value of MAXTIME% in line 1900. The instructions will automatically take note of any change you make.

PROCEDURE	ACTION
board	Prints the board
random	Chooses the appropriate number of letters at random and mixes them up
move	Checks for valid move of cursor and moves it
print	Prints character at board position given by T%
lock	Tests for valid entry of Q and reveals letter
incscore	Increases score when pair found
decscore	Deducts penalty points
dectime	Counts the seconds back from MAXTIME% to zero
display	Displays message
dd	Displays message in double height
showscore	Assembles score header and displays it
blink	Checks for valid entry of W and reveals letter briefly
yesno	Get Y or N from keyboard
init	Initialisation
delay	Forces time delay
end	Checks for new high score
wait	Waits for operator key entry
insth	Puts up header to instructions
inst	Displays instructions
rd	
init	Initialisation
delay	Forces time delay
end	Checks for new high score
wait	Waits for the operator key entry

```
   10 REM Program: PAIRS
   20 REM Version: B1.1
   30 REM Author: MICHAEL QUINION
   40 REM BEEBUG: OCTOBER 1984
   50 REM Program subject to copyright
   60 :
  100 ON ERROR GOTO 2740
  110 MODE7:VDU23,1,0;0;0;0;:PROCinit:P
ROCdd(10)
  120 FOR A%=0 TO 3 STEP 3:PRINTTAB(13,
10+A%)CHR$(150)STRING$(5,CHR$(172)):NEX
T
  130 A%=INKEY(300):MODE7:PROCinst
  140 MODE7:FOR A%=1 TO 5:PROCdisplay(A
%):NEXT
  150 Q%=GET:IF Q%<49 OR Q%>51 THEN 150
 ELSE NOPAIRS%=NP%(Q%-48):NOREPEATS%=NR
%(Q%-48):PRINTCHR$(Q%);
  160 PROCrandom:PROCboard
  170 SCORE%=0:ETIME%=0:STIME%=TIME:SEC
S%=MAXTIME%:PAIRS%=0:PROCshowscore
  180 HR$(8)="High Score: "+STR$(HISCOR
E%):PROCdd(8)
  190 T%=RND(49)-1:PROCprint(T%,BS$)
  200 REPEAT
  210 PROCdectime
  220 Q%=INKEY(0)AND&DF
  230 IF Q%=81 THENPROClock
  240 IF Q%=87 THENPROCblink
  250 IF Q%>135 AND Q%<140 THEN PROCmov
e
  260 UNTIL PAIRS%=24
  270 PROCend
  280 PROCdisplay(7):PROCyesno:IF YN% T
HEN 140
  290 CLS:*FX4,0
  300 END
  310 :
 1000 DEFPROCboard
 1010 CLS:PRINTTAB(XC%,YC%)B1$
 1020 FOR A%=1 TO 13 STEP 2:PRINTTAB(XC
%,YC%+A%)B2$TAB(XC%,YC%+A%+1)B3$:NEXT
 1030 PRINTTAB(XC%,YC%+15)B2$:PRINTTAB(
XC%,YC%+16)B4$:ENDPROC
 1040 :
 1050 DEFPROCrandom
 1060 ZA$=AZ$:FOR A%=1 TO NOPAIRS%
 1070 CHOICE%=RND(LEN(ZA$)):CHOICE$=MID
$(ZA$,CHOICE%,1)
 1080 ZA$=LEFT$(ZA$,CHOICE%-1)+MID$(ZA$
,CHOICE%+1)
 1090 FOR B%=(A%-1)*NOREPEATS% TO (A%*N
OREPEATS%)-1:BOARD$(B%)=CHOICE$:NEXTB%
 1100 NEXT A%
 1110 CHOICE%=RND(LEN(ZA$)):CHOICE$=MID
$(ZA$,CHOICE%,1):BOARD$(48)=CHOICE$
 1120 FOR A%=1 TO 100
 1130 CH1%=RND(49)-1:CH2%=RND(49)-1:IF
CH1%=CH2% THEN 1130
 1140 TEMP$=BOARD$(CH1%):BOARD$(CH1%)=B
OARD$(CH2%):BOARD$(CH2%)=TEMP$
 1150 NEXT:ENDPROC
 1160 :
 1170 DEFPROCmove
 1180 TQ%=T%+TM%(Q%-136):IF TQ%<0 OR TQ
%>48 THEN ENDPROC
 1190 IF T%MOD7=0 AND TM%(Q%-136)=-1 TH
EN ENDPROC
 1200 IF T%MOD7=6 AND TM%(Q%-136)=1 THE
N ENDPROC
 1210 T%=TQ%:PROCprint(T%,BS$):ENDPROC
 1220 :
 1230 DEFPROCprint(TZ%,TZ$)
 1240 PRINTTAB(XO%+(TZ%MOD7)*3,YO%+(TZ%
DIV7)*2)TZ$CHR$(8);:ENDPROC
```

```
 1250 :
 1260 DEFPROCclock
 1270 IF LOCK% THEN 1300
 1280 IF BOARD$(T%)=" " THEN PROCdecsco
re(DEC1%):LOCK%=FALSE:ENDPROC
 1290 PROCprint(T%,BOARD$(T%)):TL%=T%:L
OCK%=TRUE:ENDPROC
 1300 IF TL%=T% THEN ENDPROC
 1310 IF BOARD$(T%)<>" " THEN PROCprint
(T%,BOARD$(T%))
 1320 IF BOARD$(T%)=BOARD$(TL%) THEN 13
70
 1330 PROCdecscore(DEC2%):PROCdelay(4):
LOCK%=FALSE
 1340 PROCprint(TL%,"*")
 1350 IF BOARD$(T%)<>" " THEN PROCprint
(T%,"*") ELSE PROCprint(T%,CL$)
 1360 ENDPROC
 1370 PROCincscore:PROCdelay(4)
 1380 BOARD$(T%)=" ":BOARD$(TL%)=" "
 1390 PROCprint(TL%,CL$):PROCprint(T%,C
L$)
 1400 LOCK%=FALSE:ENDPROC
 1410 :
 1420 DEFPROCincscore
 1430 PAIRS%=PAIRS%+1:SOUND1,1,101,5
 1440 SCORE%=SCORE%+SECS%*2:STIME%=TIME
:SECS%=MAXTIME%:PROCshowscore:ENDPROC
 1450 :
 1460 DEFPROCdecscore(SS%)
 1470 SCORE%=SCORE%-SS%:IF SCORE%<0 THE
N SCORE%=0
 1480 SOUND0,-15,20,5:PROCshowscore:END
PROC
 1490 :
 1500 DEFPROCdectime
 1510 SECSTEST%=MAXTIME%-(TIME-STIME%)D
IV100:IF SECSTEST%<=0 THEN SECSTEST%=0
 1520 IF SECSTEST%<SECS% THEN SECS%=SEC
STEST%:PROCshowscore:SOUND1,-8,101,2:PR
OCprint(T%,BS$)
 1530 ENDPROC
 1540 :
 1550 DEFPROCdisplay(NN%)
 1560 PRINTTAB(HX%(NN%),HY%(NN%))CHR$(H
COL%(NN%))HR$(NN%);
 1570 ENDPROC
 1580 :
 1590 DEFPROCdd(NN%)
 1600 FOR A%=0 TO 1
 1610 PRINTTAB(HX%(NN%),HY%(NN%)+A%)CHR
$(141)CHR$(HCOL%(NN%))HR$(NN%)
 1620 NEXT:ENDPROC
 1630 :
 1640 DEFPROCshowscore
 1650 HR$(6)=LEFT$("SCORE: "+STR$(SCORE
%)+STRING$(15," "), 21)+"TIME: "+RIGHT$
("  "+STR$(SECS%),2)
 1660 PROCdd(6)
 1670 PROCprint(T%,BS$):ENDPROC
 1680 :
 1690 DEFPROCblink
 1700 IF LOCK% THEN PROCprint(TL%,"*"):
LOCK%=FALSE
 1710 IF BOARD$(T%)=" " THEN PROCdecsco
re(DEC1%):ENDPROC
 1720 PROCprint(T%,BOARD$(T%)):PROCdela
y(3)
 1730 PROCprint(T%,"*"):ENDPROC
 1740 :
 1750 DEFPROCyesno
 1760 YN%=FALSE
 1770 CHARA%=GETAND&DF:IF CHARA%<>89 AN
D CHARA%<>78 THEN 1770
 1780 IF CHARA%=89 THEN YN%=TRUE
 1790 PRINTCHR$(CHARA%);:ENDPROC
 1800 :
 1810 DEFPROCinit
 1820 *FX4,1
 1830 AZ$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
 1840 B1$=CHR$(146)+CHR$(224)+STRING$(2
3,CHR$(240))+CHR$(240)
 1850 B2$=CHR$(146)+CHR$(234)+CHR$(130)
+STRING$(21," ")+CHR$(146)+CHR$(234)
 1860 B3$=CHR$(146)+CHR$(234)+" "+STRIN
G$(7," * ")+" "+CHR$(234)
 1870 B4$=CHR$(146)+CHR$(162)+STRING$(2
3,CHR$(163))+CHR$(163)
 1880 A%=0:B%=0:BS$=CHR$(9):C$="":C%=0:
CH1%=0:CH2%=0:CHARA%=0:CHOICE$=""
 1890 CHOICE%=0:CL$=CHR$(8)+CHR$(148)+C
HR$(42)+CHR$(146):D1%=0:D2%=0:DEC1%=10
 1900 DEC2%=20:ETIME%=0:HR%=12:HISCORE%
=100:LOCK%=FALSE:MAXTIME%=20
 1910 PAIRS%=0:SCORE%=0:SECS%=0:STIME%=
0:T%=0:TQ%=0:TZ%=0:TZ$="":XC%=5
 1920 XO%=9:YC%=5:YN%=FALSE:YO%=7
 1930 ENVELOPE1,1,0,0,0,0,0,0,0,127,-2,-3
,-2,126,65
 1940 ENVELOPE2,1,0,0,0,0,0,0,0,127,-2,-1
,-1,126,30
 1950 DIM BOARD$(48),HX%(HR%),HY%(HR%),
HCOL%(HR%),HR$(HR%),NP%(3),NR%(3),TM%(3
)
 1960 FOR A%=0 TO HR%:READ HX%(A%),HY%(
A%),HCOL%(A%),HR$(A%):NEXT
 1970 FOR A%=0 TO 3:READ TM%(A%):NEXT
 1980 FOR A%=1 TO 3:READ NP%(A%),NR%(A%
):NEXT
 1990 ENDPROC
 2000 :
 2010 DEFPROCdelay(DD%)
 2020 FOR A%=1 TO DD%:FOR B%=1 TO 400:N
EXTB%:PROCdectime:NEXT A%:ENDPROC
 2030 :
 2040 DEFPROCend
 2050 A%=-1:REPEAT:A%=A%+1:UNTIL BOARD$
(A%)<>" ":PROCprint(A%,BOARD$(A%))
 2060 HR$(6)=LEFT$("FINAL SCORE: "+STR$
(SCORE%)+STRING$(40," "), 38):PROCdd(6)
```

```
 2070 IF SCORE%<=HISCORE% THEN ENDPROC
 2080 HISCORE%=SCORE%:PROCdd(9):FOR A%=
1 TO 2000:NEXT
 2090 SOUND1,1,101,10:SOUND1,1,117,5:SO
UND1,1,129,5:SOUND1,1,121,10
 2100 SOUND1,1,129,10:SOUND1,1,117,5:SO
UND1,1,117,5:SOUND1,1,109,10
 2110 SOUND1,2,101,10:FOR A%=1 TO 7000:
NEXT:ENDPROC
 2120 :
 2130 DEFPROCwait
 2140 PROCdisplay(11):A%=GET:ENDPROC
 2150 :
 2160 DEFPROCinsth
 2170 CLS:PROCdd(12):PRINTTAB(1,5):ENDP
ROC
 2180 :
 2190 DEFPROCinst
 2200 PROCdisplay(0):PROCyesno:IF NOTYN
% THEN ENDPROC
 2210 VDU23;8202;0;0;0;:PROCinsth
 2220 PRINT"Pairs is a test of your mem
ory."
 2230 PRINT'"You will be shown a 7 by 7
 board of"
 2240 PRINT"coloured blobs.  Behind eac
h you must"
 2250 PRINT"imagine there is a letter,
which you"
 2260 PRINT"can reveal briefly."
 2270 PRINT'"There are 24 pairs of lett
ers and"
 2280 PRINT"one odd one.  You must find
 and"
 2290 PRINT"eliminate all 24 pairs."
 2300 PRINT'"To make things more intere
sting,"
 2310 PRINT"you must work against the c
lock."
 2320 PROCwait:PROCinsth
 2330 PRINT"You have ";MAXTIME%;" secon
ds to find and"
 2340 PRINT"delete a pair of letters.
Your"
 2350 PRINT"score depends on how quickl
y you"
 2360 PRINT"manage it.  After ";MAXTIME
%;" seconds, your"
 2370 PRINT"score drops to zero, but yo
u must"
 2380 PRINT"still find a pair to restar
t the"
 2390 PRINT"clock."
 2400 PRINT'"Use the cursor keys to mov
e around"
 2410 PRINT"the board.  Press 'W' to br
iefly"
 2420 PRINT"reveal a letter.  Press 'Q'
 to"
 2430 PRINT"select one letter of a pair
.  Move"
 2440 PRINT"the cursor to the other and
 press"
 2450 PRINT"'Q' again to delete them bo
th."
 2460 PROCwait:PROCinsth
 2470 PRINT"There are penalties for sel
ecting"
 2480 PRINT"a pair that doesn't match,
and for"
 2490 PRINT"selecting a letter already"
 2500 PRINT"eliminated."
 2510 PRINT'"According to the level of
difficulty"
 2520 PRINT"you choose, a given letter
will"
 2530 PRINT"occur on the board eight ti
mes,"
 2540 PRINT"four times, or twice."
 2550 PRINT'"GOOD LUCK!"
 2560 PROCwait:ENDPROC
 2570 :
 2580 DATA3,12,128,"Do you want instruc
tions? "
 2590 DATA7,9,128,"How difficult a game
?"
 2600 DATA9,11,128,"1. Moderate"
 2610 DATA9,12,128,"2. Difficult"
 2620 DATA9,13,128,"3. Horrible"
 2630 DATA9,17,128,"Your choice: "
 2640 DATA2,1,134,"Score: "
 2650 DATA4,23,134,"Do you want another
 game? "
 2660 DATA2,3,130,"High Score: "
 2670 DATA2,3,134,"Well done!  A New h
igh score"
 2680 DATA12,11,134,"Pairs"
 2690 DATA3,23,134,"Press any key to co
ntinue"
 2700 DATA8,1,134,"INSTRUCTIONS"
 2710 DATA-1,1,7,-7
 2720 DATA6,8,12,4,24,2
 2730 :
 2740 ON ERROR OFF
 2750 MODE7:*FX4,0
 2760 IF ERR<>17 REPORT:PRINT" at line
";ERL
 2770 END
```

ACKNOWLEDGEMENT

In the review of Acorn's Bitstik published in the previous issue (BEEBUG Vol.3 No.4) we omitted to acknowledge the help and assistance given to our reviewer by CJE Micros of Worthing with the loan of additional equipment. Our thanks to this company.

Tested on Basic I & II and O.S. 1.2

KITCHEN CHAOS (32k)

by Kai Sui Ng

Kitchen Chaos is a very original multicoloured action game for one player requiring some quick reactions and a frying pan! The game requires you to 'guide' your breakfast on to the cooker taking care not to drop any food on the way.

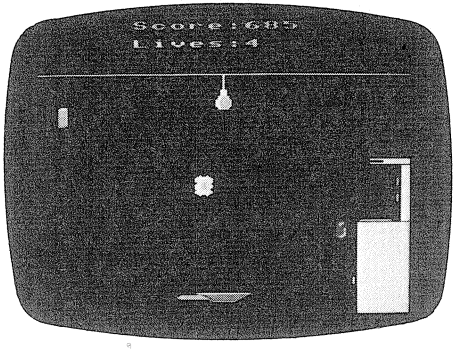
In this game you have to 'bounce' the ingredients for your breakfast fry-up across the room to the cooker. You do this by guiding a frying pan along the bottom of the screen to catch and toss the food. Your breakfast fry-up consists of eggs, chips and sausages.

You begin the game with five lives and just one item of food to guide across to the cooker. If you drop any food on the floor then you lose a life. After losing three lives the game is over and the high scores are shown.

As the game progresses the amount of food that appears on the screen at any one time increases, and when there are about four or five items on the screen the game becomes very hard, demanding quick thinking and fast reactions.

The keys for controlling movement are simply 'Z' and 'X' for left and right. Take care when typing in the program, particularly with the character and string definitions in the procedure character definitions (between lines 1260 and 1820) or you may find that the contents of your breakfast have a most peculiar appearance (and it is not easy to sort out mistakes in this part of the program). Various REM statements in the program listing will also assist in understanding the program.

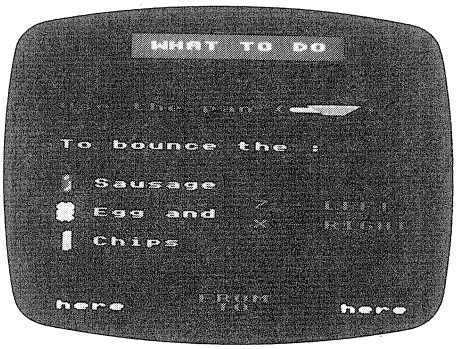
All in all this is a most exhilarating and satisfying game.




```
  10 REM PROGRAM Fry Up
  20 REM VERSION B0.4
  30 REM AUTHOR  KAI SUI NG
  40 REM BEEBUG  OCT 84
  50 REM PROGRAM SUBJECT TO COPYRIGHT
  60:
 100 ON ERROR GOTO 3140
 110 *FX 9,15
 120 *FX 10,15
 130 *FX 11,0
 140 DIM mapX%(21),mapY%(21),Fpos%(6),
snd%(6),frt%(6),food$(3),hi%(11)
 150 FOR L=1 TO 10:hi%(L)=100*(11-L):N
EXT
 160 PROCcharacterdefinitions
 170 Fpos%(0)=0
 180 REPEAT
 190 MODE 7:VDU 23;11,0;0;0;0
 200 RESTORE 2130
 210 PROCtitle
 220 MODE 2:VDU 23;11,0;0;0;0
 230 PROCtitle2
 240 MODE 2:VDU 5
 250 PROCinitialise
 260 REPEAT
 270 FOR L=1 TO 6
 280 frt%(L)=RND(3)
 290 snd%(L)=100+(RND(10)*8):NEXT
 300 Fpos%(1)=0:FOR L=2 TO 6:Fpos%(L)=
Fpos%(L-1)-RND(40/no%):NEXT
 310 U%=1
 320 *FX 15,1
 330 REPEAT
 340 lose%=0
 350 FOR grub%=1 TO no%
 360 PROCpan
 370 PROCfood
 380 IF lose%=0 AND (Fpos%(grub%)=5 OR
 Fpos%(grub%)=13 OR Fpos%(grub%)=19) SO
UND 0,1,5,2:score%=score%+5
 390 IF U% IF lose%=1 foodno%=grub%:U%
=0:MOVE mapX%(Fpos%(grub%)),mapY%(Fpos%
(grub%)):PRINT Fwipe$:grub%=no%
```

→

```
  400 NEXT grub%
  410 PROCscore
  420 UNTIL U%=0 OR Fpos%(no%)=22
  430 IF Fpos%(no%)=22 PROCincrease:UNT
IL FALSE
  440 PROCsplat
  450 FOR L=1 TO 21:MOVE mapX%(L),mapY%
(L):PRINT Fwipe$:NEXT
  460 UNTIL lives%=0
  470 PROCscores:UNTIL FALSE
  480 END
  490:
 1000 DEF PROCtitle2
 1010 COLOUR 129
 1020 PRINT TAB(4,1);SPC(12)
 1030 PRINT TAB(4,2);" WHAT TO DO "
 1040 PRINT TAB(4,3);SPC(12)
 1050 COLOUR 1:COLOUR 128
 1060 PRINT TAB(0,9);"Use the pan (
)"
 1070 COLOUR 3:PRINT''''"To bounce the :
"''''
 1080 PRINT"  Sausage"''''"  Egg and"''''
"  Chips"'
 1090 COLOUR 1
 1100 PRINT 'TAB(11,19);"Z - LEFT"
 1110 PRINT 'TAB(11);"X - RIGHT"
 1120 PRINT TAB(8,29);"FROM"'TAB(9);"TO"
```




```
 1130 COLOUR 8:PRINT TAB(0,30);"here"
 1140 COLOUR 15:PRINT TAB(16,30);"here";
 1150 VDU 5
 1160 GCOL 0,3
 1170 MOVE 836,732:PRINT pan$
 1180 GCOL 0,4
 1190 MOVE 836,732:PRINT pan2$
 1200 MOVE 0,492:PRINT food$(1)
 1210 MOVE 0,396:PRINT food$(2)
 1220 MOVE 0,300:PRINT food$(3)
 1230 A=INKEY(3000)
 1240 ENDPROC
 1250:
 1260 DEF PROCcharacterdefinitions
 1270 REM sausage (main)
 1280 VDU 23,224,28,62,62,62,62,62,62,62
 1290 VDU 23,225,62,62,62,62,62,62,62,28
 1300 REM sausage (shade)
 1310 VDU 23,226,0,0,8,24,16,0,0,4
 1320 VDU 23,227,4,0,4,4,12,24,0,0
 1330 REM egg      (main)
 1340 VDU 23,228,36,126,255,255,255,127
,126,255
 1350 VDU 23,229,255,255,254,126,255,25
5,126,36
 1360 REM egg      (shade)
 1370 VDU 23,230,0,0,0,0,0,28,56,28
 1380 VDU 23,231,56,28,56,0,0,0,0,0
 1390 REM chip     (main)
 1400 VDU 23,232,0,28,28,28,28,28,28,28
 1410 VDU 23,233,28,28,28,28,28,28,28,28
 1420 REM chip     (shade)
 1430 VDU 23,234,14,30,2,2,2,2,2,2
 1440 VDU 23,235,2,2,2,2,2,2,2,0
 1450 REM pan      (main)
 1460 VDU 23,244,0,0,127,255,255,127,0,0
 1470 VDU 23,245,255,127,255,255,255,25
5,3,1
 1480 VDU 23,246,255,255,255,255,255,25
5,255,255
 1490 VDU 23,247,255,254,252,248,240,22
4,192,128
 1500 REM pan      (shade)
 1510 VDU 23,248,0,63,31,15,7,3,1,0
 1520 VDU 23,249,0,255,255,255,255,255,
255,0
 1530 VDU 23,250,0,252,248,240,224,192,
128,0
 1540 REM sizzle
 1550 VDU 23,236,74,4,161,74,4,161,4,161
 1560 REM light
 1570 VDU 23,237,56,56,56,56,56,124,124
,124
 1580 VDU 23,238,254,254,254,254,254,25
4,124,124
 1590 VDU 23,239,0,0,0,0,0,56,56,56
 1600 VDU 23,240,56,124,124,124,124,56,
56,0
 1610 REM strings
 1620 REM pan
 1630 pan$=CHR$244+CHR$245+CHR$246+CHR$
247
 1640 pan2$=" "+CHR$248+CHR$249+CHR$250
 1650 REM food
 1660 food$(1)=CHR$18+CHR$0+CHR$1+CHR$2
24+CHR$10+CHR$8+CHR$225
 1670 food$(1)=food$(1)+CHR$18+CHR$0+CH
R$5+CHR$8+CHR$11+CHR$226+CHR$10+CHR$8
 1680 food$(1)=food$(1)+CHR$227
 1690 food$(2)=CHR$18+CHR$0+CHR$7+CHR$2
28+CHR$10+CHR$8+CHR$229
 1700 food$(2)=food$(2)+CHR$18+CHR$0+CH
R$3+CHR$8+CHR$11+CHR$230+CHR$10+CHR$8
 1710 food$(2)=food$(2)+CHR$231
```

```
1720 food$(3)=CHR$18+CHR$0+CHR$3+CHR$2
32+CHR$10+CHR$8+CHR$233
1730 food$(3)=food$(3)+CHR$18+CHR$0+CH
R$7+CHR$8+CHR$11+CHR$234+CHR$10+CHR$8
1740 food$(3)=food$(3)+CHR$235
1750 REM wipe food
1760 Fwipe$=CHR$18+CHR$0+CHR$0+CHR$246
+CHR$8+CHR$10+CHR$246
1770 REM sizzle effect
1780 sizz$=CHR$18+CHR$0+CHR$1+CHR$236+
CHR$8+CHR$10+CHR$236
1790 REM light bulb
1800 light$=CHR$18+CHR$0+CHR$7+CHR$237
+CHR$8+CHR$10+CHR$238
1810 light$=light$+CHR$11+CHR$8+CHR$18
+CHR$0+CHR$3+CHR$239+CHR$8+CHR$10+CHR$2
40
1820 ENDPROC
1830:
1840 DEF PROCinitialise
1850 VDU 23;11,0;0;0;0
1860 ENVELOPE 1,1,0,0,0,0,0,0,30,-1,0,
-3,126,90
1870 ENVELOPE 2,130,-3,-2,1,5,5,50,60,
0,0,-6,60,60
1880 @%=0:lives%=5:score%=0:no%=1:C%=0
1890 GCOL 0,7:MOVE 0,772:DRAW 1280,772
1900 MOVE 632,772:DRAW 632,720
1910 MOVE 608,720:PRINT light$
1920 GCOL0,7:MOVE 1100,0:MOVE 1272,0
1930 PLOT 85,1100,300:PLOT 85,1272,300
1940 MOVE 1248,308:MOVE 1272,308
1950 PLOT 85,1248,480:PLOT 85,1272,480
1960 MOVE 1140,488:MOVE 1272,488
1970 PLOT 85,1272,502:PLOT 85,1140,502
1980 GCOL0,5:MOVE 1086,100:PLOT 1,0,20
1990 MOVE 1232,370:PLOT 1,0,20
2000 MOVE 1232,420:PLOT 1,0,20
2010 GCOL 0,6:MOVE 1108,308:PLOT 1,104
,0
2020 FOR loop=1 TO 21
2030 READ X%,Y%:mapX%(loop)=X%
2040 mapY%(loop)=Y%
2050 NEXT loop
2060 GCOL0,9:MOVE 0,68:PRINT pan$:GCOL
0,10:MOVE 0,68:PRINT pan2$
2070 GCOL0,11:MOVE 486,68:PRINT pan$:G
COL0,12:MOVE 486,68:PRINT pan2$
2080 GCOL0,13:MOVE 808,68:PRINT pan$:G
COL0,14:MOVE 808,68:PRINT pan2$
2090 pa%=2
2100 ENDPROC
2110:
2120 REM positions for food
2130 DATA 0,752,48,664,80,528,112,336,
138,136
2140 DATA 192,308,262,472,320,568,400,
584,480,560,542,448,568,304,608,136
2150 DATA 656,280,718,416,798,440,878,
384,910,264,950,136
2160 DATA 1010,308,1128,376
2170:
2180 DEF PROCpan
2190 K$=INKEY$(1)
2200 IF K$="Z" pa%=pa%-1:IF pa%=0 pa%=1
2210 IF K$="X" pa%=pa%+1:IF pa%=4 pa%=3
2220 IF pa%=1 VDU 19,9,3;0;19,10,4;0;1
9,11,0;0;19,12,0;0;19,13,0;0;19,14,0;0;
2230 IF pa%=2 VDU 19,11,3;0;19,12,4;0;
19,9,0;0;19,10,0;0;19,13,0;0;19,14,0;0;
2240 IF pa%=3 VDU 19,13,3;0;19,14,4;0;
19,11,0;0;19,12,0;0;19,9,0;0;19,10,0;0;
2250 ENDPROC
2260:
2270 DEF PROCfood
2280 IF grub%>no% OR Fpos%(grub%)=22 E
NDPROC
2290 IF Fpos%(grub%)<0 Fpos%(grub%)=Fp
os%(grub%)+1:ENDPROC
2300 MOVE mapX%(Fpos%(grub%)),mapY%(Fp
os%(grub%)):PRINT Fwipe$
2310 Fpos%(grub%)=Fpos%(grub%)+1
2320 IF Fpos%(grub%)=22 PROCsizzle:END
PROC
2330 SOUND 2,2,snd%(grub%),1
2340 MOVE mapX%(Fpos%(grub%)),mapY%(Fp
os%(grub%)):PRINT food$(frt%(grub%))
2350 IF mapY%(Fpos%(grub%))>136 ENDPROC
2360 FOR W%=1 TO 8:PROCpan:lose%=0
2370 IF Fpos%(grub%)=5 AND pa%>1 lose%
=1
2380 IF Fpos%(grub%)=13 AND pa%<>2 los
e%=1
2390 IF Fpos%(grub%)=19 AND pa%<3 lose
%=1
2400 IF lose%=0 W%=8
2410 NEXT
2420 ENDPROC
2430:
2440 DEF PROCscore
2450 VDU 4:COLOUR 5:PRINT TAB(5,2);"Sc
ore:";:COLOUR 3:PRINT score%
2460 COLOUR 6:PRINT TAB(5,4);"Lives:";
:COLOUR 3:PRINT lives%:VDU5
2470 ENDPROC
2480:
2490 DEF PROCsplat
2500 FOR L=170 TO 50 STEP-8
2510 MOVE mapX%(Fpos%(foodno%)),0:GCOL
0,RND(7)
2520 PLOT 1,81-RND(161),RND(28)
2530 SOUND 1,-8,L,2:SOUND 0,1,3,2
2540 PROCpan
2550 NEXT
2560 lives%=lives%-1:PROCscore
2570 IF lives%<1 ENDPROC
2580 VDU 4:COLOUR128
2590 PRINT TAB(7,12);"Ready!"
2600 FOR W%=1 TO 80:PROCpan:NEXT
2610 PRINT TAB(7,12);"       "
```

```
 2620 PRINT TAB(0,31);SPC(17);:VDU5
 2630 ENDPROC
 2640:
 2650 DEF PROCsizzle:score%=score%+20
 2660 PROCscore
 2670 FOR L=255 TO 150 STEP-30
 2680 MOVE mapX%(21),mapY%(21):PRINT si
zz$
 2690 SOUND 1,0,L,1:SOUND 0,1,7,1
 2700 MOVE mapX%(21),mapY%(21):PRINT Fw
ipe$:PROCpan
 2710 NEXT
 2720 ENDPROC
 2730:
 2740 DEF PROCscores
 2750 CLG:VDU 4:CLS:COLOUR 2
 2760 place%=11
 2770 PRINT "Your score:";score%''
 2780 FOR L=10 TO 1 STEP -1:IF score%>h
i%(L) place%=L
 2790 NEXT
 2800 FOR L=11 TO place% STEP -1
 2810 hi%(L)=hi%(L-1)
 2820 NEXT
 2830 hi%(place%)=score%
 2840 COLOUR 3:PRINT''"High Scores :"'''
 2850 FOR L=1 TO 10
 2860 COLOUR 5:IF L=place% COLOUR 8:SOU
ND 2,2,200,10:SOUND 1,2,100,10
 2870 @%=2:PRINT L;" >";TAB(10);:@%=5:P
RINT hi%(L)':NEXT
 2880 *FX 15,1
 2890 A=INKEY(2000)
 2900 ENDPROC
 2910:
 2920 DEF PROCtitle
 2930 PRINT TAB(8,2);CHR$132;CHR$157;CHR
$131;CHR$141;"Kitchen Chaos   ";CHR$156
 2940 PRINT TAB(8,3);CHR$132;CHR$157;CHR
$131;CHR$141;"Kitchen Chaos   ";CHR$156
 2950 PRINT ''CHR$131;"  It's hell in t
he kitchen !! All the"
 2960 PRINT CHR$(131);"food has decided
  to hurl itself from"
 2970 PRINT CHR$(131);"the ceiling. Wou
ld you help me to"
 2980 PRINT CHR$(131);"regather it all
? Unfortunatly, I have"
 2990 PRINT CHR$(131);"only five old fr
ying-pans and you'll "
 3000 PRINT CHR$(131);"have to guide th
e food with it, along"
 3010 PRINT CHR$(131);"to the cooker."
 3020 PRINT 'CHR$(130);"  Because of yo
ur kindness I will"
 3030 PRINT CHR$(130);"reward you for s
aving my food"
 3040 PRINT CHR$(130);"- I will give yo
u five (5) noogies for"
 3050 PRINT CHR$(130);"every good bounc
e, and twenty (20)"
 3060 PRINT CHR$(130);"noogies for ever
y item of food"
 3070 PRINT CHR$(130);"transported to t
he cooker."
 3080 PRINT 'CHR$(129);"  I'm afraid I'
ll have to find some-"
 3090 PRINT CHR$(129);"one else if you
miss five (5) times."
 3100 PRINT 'CHR$(133);TAB(15);"GOOD LU
CK!";
 3110 A=INKEY(5000)
 3120 ENDPROC
 3130:
 3140 ON ERROR OFF
 3150 MODE 7:*FX4,0
 3160 *FX12,0
 3170 IF ERR=17 END
 3180 REPORT:PRINT" at line ";ERL
 3190 PRINT
 3200 END
 3210:
 3220 DEFPROCincrease
 3230 C%=C%+1
 3240 IF C%=2 OR C%=4 OR C%=7 OR C%=11
OR C%=16 no%=no%+1
 3250 ENDPROC
```

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

MULTIPLE COPIES WITH WORDWISE - K. Walker

Whilst Wordwise has no inbuilt facility to print more than one copy of a document a simple program will do this task on a disc system. The procedure is:

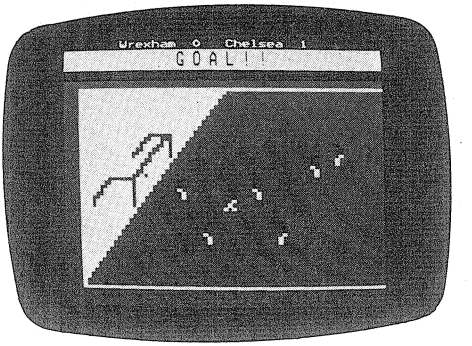
Write the document
Spool a copy as say "TEXT"
Return to Basic
Run this program

```
 10  VDU 2
 20  INPUT"How many copies",x
 30  FOR copy = 1 TO x
 40  *TYPE TEXT
 50  CLS
 60  NEXT copy
 70  VDU 3
```

FOOTBALL MANAGER BY ADDICTIVE SOFTWARE

Reviewed by Alan Webster

Name	: Football Manager
Supplier	: Addictive Games
Price	: £7.95
Reviewer	: Alan Webster
Rating	: ****



Football Manager is a comprehensive simulation game of the world of professional football, in which you play the role of the manager of your favourite football team. You start at the bottom of Division Four and have to try to work your way up through each division until you reach Division One. You also play in F.A.Cup matches and there's even a transfer market, and an injury list. The program also provides graphical representations of the most important highlights of each game.

There are 7 levels of skill, and I must say that I found the game quite easy at the lowest level, winning the Cup two seasons in a row and winning the First Division championship, but at the highest level I just avoided relegation three seasons running.

The more successful you are, the more money you make. This means that you can afford to buy better players. It is wise to start at an easy level and to build up a good side before attempting one of the harder levels.

Overall, I think this game will sell very well. I am more of an arcade games fanatic, but I certainly enjoyed the almost realistic feel to this game. So if you're bored with 'zappo' games, give this a try - it will keep you amused for some time to come.

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

MULTI-LINE PAGE HEADINGS IN WORDWISE - I. Malcolm

To get a multiple line heading use Wordwise to create the text for the heading you want in a file by editing it in the normal way and saving with the SPOOL option. Now define the heading with:

```
DH GF"heading"
```

Where "heading" is the name of the file.

CLEARING WORDWISE - Tim Powys-Lybbe

To clear Wordwise without pressing the Break key (which destroys your drive and directory settings etc.) use *W. from the menu to re-start Wordwise. Then simply press N to the prompt.

REVERSING FLAGS - R. Sterry

To change a boolean flag between its states (FALSE and TRUE) you could use:

```
IF FLAG%=TRUE THEN FLAG%=FALSE ELSE FLAG%=TRUE
```

More elegant however is:

```
FLAG%=NOT FLAG%
```

Tested on Basic I & II and O.S. 1·2

ANALYSING THE PERFORMANCE OF BASIC PROGRAMS (16k)

by K. H. Craig

The utility this month is a clever application of the Beeb's timers to provide information on how long is spent running each part of your Basic program. With it, you should be able to find the time wasting sections of your program, and thus make them more efficient.

INTRODUCTION

Have you ever wondered which parts of a Basic program were taking the most time, or how to speed up the program. The Basic variable TIME can of course be used to time the execution of parts of programs, but although it is convenient for timing whole programs or procedures, this method is rather tedious if you want a line by line account of where the time is going.

The programs given here allow such an analysis to be easily carried out. The programs are based on an interrupt routine that takes 'snapshots' of your Basic program as it runs and then prints out a line by line analysis of the running time. This information can be quite informative. It is surprising how many programs spend most of their time executing only one or two of their lines. Once these lines have been identified, it is usually quite easy to substantially speed up the program by recoding these lines (perhaps in assembler if speed is really crucial).

USING THE PROGRAMS

The programs should first be typed in and saved to tape or disc. Since the programs are automatically chained in, it is important to call them the same names as given in the listings, viz. "ANALYSE", "ANALMC" and "ANALBAS". Also in the case of saving to tape, they should be saved in that order. One further point is that when typing in ANALBAS, care should be taken not to insert too many additional spaces as it is vital that the program length does not exceed 511 bytes. To be on the safe side, you could omit the REM statements altogether.

To use the programs, first press BREAK to ensure that the computer's pointers and vectors are correctly initialised and then CHAIN "ANALYSE". A summary of the instructions is displayed while the programs are being loaded. Once loaded, pressing any key will clear the screen and return with the normal prompt ready for you to type in or load a Basic program as usual. The analysis programs will remain hidden until required. The only effect of these programs at this stage is that PAGE has been increased (by &500 - to &1300 on a cassette system or &1E00 on an Acorn disc system). You should not reduce PAGE below these values or the system is liable to crash.

Example Program Showing Use of * On and * Off

```
10 REM Program: EXAMPLE
20 REM Author: K.H.Craig
30 REM
40 *ON
50 FOR I=1 TO 1000: NEXT
60 FOR I=1 TO 3000: NEXT
70 FOR I=1 TO 6000: NEXT
80 *OFF
```

Two * commands are provided, *ON and *OFF (only the first two characters are actually checked) to start and stop the timing process. For simple programs it may only be necessary to insert a *ON at the beginning and a *OFF at the end of the program. Note that the end of the program means the last executable line and not the highest numbered line, which is different if, for example, procedures are used. It may be desirable to disable the analyser during INPUT statements by bracketing the relevant line with an *OFF/*ON pair, otherwise the times given will include the time it takes you to type in your reply. Note also that if the final *OFF is omitted, the results of the analysis may be misleading. Program "EXAMPLE" illustrates this.

LIMITATIONS

One restriction imposed, for reasons to be explained below, is that the line numbers used should be less than 256. At first sight this may seem a rather severe limitation, but in practice, few programs have more than this number of lines. It is then only necessary to renumber the program (e.g. with RENUMBER1,1) to reduce the line numbers below 256. What actually happens is that when the program looks to see which line to deal with, it looks only at the low byte and then increments a counter for a line number in the range 0 to 255. Provided this is remembered and understood, then you can use programs with line numbers greater than 255 by renumbering first.

DISPLAYING THE ANALYSIS

Having saved the three analysis programs to tape or disc, CHAIN "ANALYSE", and follow the instructions. You can then enter (or load if previously saved) the EXAMPLE program, and run it. Once finished, press f9 for the analysis. The results are displayed in 4 columns - line number, time spent (in seconds) executing that line, and percentage of the total time. The fourth column displays a string of asterisks corresponding to the percentage time spent on that line (each asterisk = 5%). This enables the time consuming lines to be picked out easily as the display scrolls up. Since the analyser works by taking snapshots of the Basic program every 20ms or so, infrequently executed lines may not be "caught in the act" and will not be displayed. This is actually an advantage as it leads to a less cluttered display.

The program can be re-run as often as required. Subsequent analyses will be aggregated, and this will help to improve the statistics and smooth out the results. The analyser can be reinitialised by pressing f8, and this will be necessary if the program is modified or a new program loaded in. The Break key is programmed to protect the analyser from a Break. To reset the computer to its initial state it is simplest to do a hard reset (Ctrl-Break), or *FX18, and then press Break.

PROGRAM NOTES - ANALYSE

ANALYSE simply displays the title page and increases PAGE by &500 before chaining in ANALMC. The displayed instructions could be completely omitted if not required.

PROGRAM NOTES - ANALMC

ANALMC contains the source code for the various assembler routines required. "init" redirects the event vector to point to the routine "intrpt", and also redirects the command line interpreter vector through our routine "cli".

"cli" looks for the first two letters of the command being "ON" or "OF". If present, the routine respectively enables or disables the start of vertical sync event. This event occurs 50 times a second and is a convenient way of obtaining regular interrupts at the right rate. Any other * command is vectored back to the operating system to be dealt with as usual.

When the event routine "intrpt" is entered, locations &0B and &0C are examined. These point to the Basic line that is currently being executed (in both Basic I and II). The line is then searched for the line number. This is easily found as each line of a Basic program begins with the 4 bytes - &0D, high byte of line number, low byte of line number, and length of line. The low byte of the line number is used as an index into the table of locations called "table%". The entry corresponding to the current line number is then incremented.

"table%" is a table of 512 bytes, and consists of 256 16-bit integers. It is the length of this table which restricts us to line numbers less than 256. Higher line numbers could easily be accommodated, but this would mean eating further into the space available to the user's program. An alternative method, which would use less memory, is to use a hashing technique. However, apart from being more complicated, we would run the risk of exceeding the recommended maximum time for an interrupt routine of between 1 and 2ms. (The worst case search time is already about 1.5 ms.) The use of 16-bit

integers was thought adequate as this allows a maximum of about 20 minutes execution time per line; in the unlikely event that this is exceeded, the program behaves sensibly and sets the entry to its maximum value.

"cleartable", as its name suggests, simply sets all entries of table% to 0.

The program "ANALMC" also sets up the keys f8 - f10 to call the relevant routines. (f7 is used temporarily, but may be reprogrammed once the programs have loaded.) It then loads in the final program "ANALBAS". Care has been taken throughout to ensure that the programs are relocatable and therefore correctly work whatever the initial value of PAGE (which is different for the tape filing system and the various DFSs around). This is the reason for the rather long-winded method of loading "ANALBAS", where we pass a *LOAD command via the command line interpreter. Note the (undocumented) use of STR$~ to produce a hexadecimal string corresponding to the load address. The version of ANALMC on the magazine cassette/disc is a fully commented version of the program listed here.

PROGRAM NOTES - ANALBAS

"ANALBAS" itself simply examines "table%", looking for non-zero entries and works out the times and percentages for these lines. A text window is defined to improve the display, and before returning, the display, print format and PAGE are restored.

For those who may wish to develop the routines further, the following information on memory layout will be useful (especially if the present page boundaries are crossed):

```
PAGE          - PAGE+&1FF   program ANALBAS
PAGE+&200     - PAGE+&3FF   table%
PAGE+&400     - PAGE+&4FF   machine code
                            routines.
PAGE+&500                   new value of
                            PAGE.
```

```
10 REM Program ANALYSE
20 REM Author K.H.Craig
30 REM Version B1.0
40 REM BEEBUG October 1984
50 REM Program Subject to Copyright
60 :
100 MODE7
110 PROCinstructions
120 VDU28,5,23,39,21
130 PAGE=PAGE+&500
140 CHAIN "ANALMC"
150 END
160 :
1000 DEF PROCinstructions
1010 FOR I=1 TO 2
1020 PRINT CHR$(141);CHR$(131);SPC(6);
"BASIC RUN-TIME ANALYSER"
1030 NEXT I
1040 PRINT'"The BASIC program to be an
alysed is"
1050 PRINT"typed in or loaded as usual
. The"
1060 PRINT"following commands should b
e entered"
1070 PRINT"in the program to switch th
e monitoring"
1080 PRINT"on or off:"
1090 PRINTCHR$(131);"*ON";CHR$(135);"
- monitoring on"
1100 PRINTCHR$(131);"*OFF";CHR$(135);"
- monitoring off"
1110 PRINT"The program should be renum
bered with"
1120 PRINTCHR$(131);"RENUMBER1,1";CHR$
(135);"(if necessary)."
1130 PRINT"Now";CHR$(131);"RUN";CHR$(1
35);"the program. When finished,"
1140 PRINT"press function key";CHR$(13
1);"f9";CHR$(135);"for the analysis."
1150 PRINT"Function key";CHR$(131);"f8
";CHR$(135);"re-initialises the"
1160 PRINT"analyser (otherwise subsequ
ent runs are"
1170 PRINT"aggregated)."
1180 PRINT'"Wait while the programs lo
ad..."
1190 ENDPROC
```

```
10 REM Program ANALMC
20 REM Author K.H.Craig
30 REM Version B1.0
40 REM BEEBUG October 1984
50 REM Program Subject to Copyright
60 :
100 REM Set up page pointers
110 page%=PAGE
120 MC%=page%-&100
130 table%=page%-&300
140 basic%=page%-&500
150 EVNTV=&220: CLIVEC=&208
160 OLDVEC=&70: ptr=&72: tabptr=&74
170 CLIOLD=&76: strptr=&78
```

```
180 FOR I%=0 TO 2 STEP 2
190 P%=MC%+&10
200 [OPT I%
210 .init
220 SEI
230 LDA EVNTV
240 STA OLDVEC
250 LDA EVNTV+1
260 STA OLDVEC+1
270 LDA #intrpt MOD 256
280 STA EVNTV
290 LDA #intrpt DIV 256
300 STA EVNTV+1
310 LDA CLIVEC
320 STA CLIOLD
330 LDA CLIVEC+1
340 STA CLIOLD+1
350 LDA #cli MOD 256
360 STA CLIVEC
370 LDA #cli DIV 256
380 STA CLIVEC+1
390 CLI
400 JMP cleartable
410 ]
420 P%=MC%
430 [OPT I%
440 .cleartable
450 LDA #0
460 LDY #0
470 .yloop
480 STA table%,Y
490 STA table%+256,Y
500 INY
510 BNE yloop
520 RTS
530 ]
540 P%=MC%+&40
550 [OPT I%
560 .intrpt
570 CMP #4
580 BNE exit
590 PHP
600 PHA
610 TXA
620 PHA
630 TYA
640 PHA
650 LDA &0B
660 STA ptr
670 LDA &0C
680 STA ptr+1
690 DEC ptr+1
700 LDA #&0D
710 LDY #&FF
720 .searchloop
730 CMP (ptr),Y
740 BEQ found
750 DEY
760 BNE searchloop
770 LDA #13
780 LDX #4
790 LDY #0
800 JSR &FFF4
810 JMP restore
820 .found
830 INY
840 INY
850 LDA (ptr),Y
860 ASL A
870 TAY
880 LDA #table% DIV 256
890 ADC #0
900 STA tabptr+1
910 LDA #table% MOD 256
920 STA tabptr
930 CLC
940 LDA (tabptr),Y
950 ADC #1
960 STA (tabptr),Y
970 INY
980 LDA (tabptr),Y
990 ADC #0
1000 STA (tabptr),Y
1010 BCC restore
1020 LDA #&FF
1030 STA (tabptr),Y
1040 DEY
1050 STA (tabptr),Y
1060 .restore
1070 PLA
1080 TAY
1090 PLA
1100 TAX
1110 PLA
1120 PLP
1130 .exit
1140 JMP (OLDVEC)
1150 ]
1160 P%=MC%+&A0
1170 [OPT I%
1180 .cli
1190 STX strptr
1200 STY strptr+1
1210 LDY #1
1220 LDA (strptr),Y
1230 CMP #ASC("O")
1240 BEQ gotO
1250 LDY strptr+1
1260 JMP (CLIOLD)
1270 .gotO
1280 INY
1290 LDA (strptr),Y
1300 CMP #ASC("N")
1310 BNE notON
1320 LDA #14
1330 BNE clifx
1340 .notON
1350 CMP #ASC("F")
1360 BEQ gotOF
1370 LDY strptr+1
1380 JMP (CLIOLD)
1390 .gotOF
1400 LDA #13
1410 .clifx
1420 LDX #4
1430 LDY #0
1440 JMP &FFF4
1450 ]
1460 NEXT I%

 1470 REM Initialise interrupt routine
and clear table%
 1480 CALL init
 1490 REM Set up function keys 7 to 10
 1500 REM KEY 7  - To "NEW" this program
 1510 REM KEY 8  - To re-initialise the
 analyser
 1520 REM KEY 9  - To run the analyser
 1530 REM KEY 10 - To reinitialise on B
REAK
 1540 *KEY7 NEW|MCLS|M
 1550 *KEY8 CALL PAGE-&100|M
 1560 *KEY9 PAGE=PAGE-&500|MRUN|M
 1570 *KEY10 OLD|MCALL PAGE+&410|MPAGE=
PAGE+&500|MCLS|M
 1580 REM Load in analysis program belo
w this program
 1590 REM Pass the *LOAD command throug
h OSCLI to
 1600 REM  allow us to pass a variable
load address
 1610 DIM cmd 20
 1620 $cmd="LOAD ANALBAS "+STR$~(basic%)
 1630 X%=cmd MOD 256
 1640 Y%=cmd DIV 256
```

→

```
 1650 CALL &FFF7
 1660 REM Output prompt and wait for re
sponse
 1670 VDU12,7
 1680 PRINTCHR$(134);"Press any key to
continue..."
 1690 *FX15,1
 1700 A=GET
 1710 VDU26,12
 1720 REM "NEW" this program
 1730 *FX138,0,135
 1740 END
```

```
 10 REM ANALBAS V1.0
 20 REM October 1984
 30 :
100 *FX13,4
110 f%=@%:ONERROR GOTO230
 120 t%=PAGE+&200
 130 CLS:PRINT" ANALYSIS RESULTS"''"
Line  Time  Percent"'
 140 VDU28,0,24,39,4:@%=&0606
 150 s%=0:FORi%=0 TO 510 STEP 2:s%=s%+
(t%!i% AND &FFFF):NEXTi%
 160 FORi%=0 TO 255
 170 c%=t%!(2*i%) AND &FFFF
 180 IFc%>0 pc%=INT(100*c%/s%+0.5):PRIN
T i%,c%*0.02,pc%;" ";STRING$(pc%/5,"*")
 190 NEXTi%
 200 PRINT'"Number of snapshots = "s%'
'"Press any key to exit..."
 210 *FX15,1
 220 a=GET
 230 ONERROR OFF:VDU26:@%=f%:PAGE=PAGE
+&500
 240 END
```

IF YOU WRITE TO US

BACK ISSUES (Members only)
All back issues are kept in print (from April 1982). Send 90p per issue PLUS an A5 SAE to the subscriptions address. This offer is for members only, so it is ESSENTIAL to quote your membership number with your order. Please note that the BEEBUG Reference Card and BEEBUG supplements are not supplied with back issues.

SUBSCRIPTIONS
Send all applications for membership, subscription renewals, and subscription queries to the subscriptions address.

MEMBERSHIP COSTS:

U.K.
£5.40 for 6 months (5 issues)
£9.90 for 1 year (10 issues)
Eire and Europe
Membership £16 for 1 year.
Middle East £19
Americas and Africa £21
Elsewhere £23
Payments in Sterling preferred.

Subscriptions & Software Address	Subscriptions and Software Help Line
BEEBUG	St.Albans
PO BOX 109	(0727) 60263
High Wycombe	Manned Mon-Fri
Bucks	1pm-4pm

PROGRAMS AND ARTICLES

All programs and articles used are paid for at around £25 per page, but please give us warning of anything substantial that you intend to write. In the case of material longer than a page, we would prefer this to be submitted on cassette or disc in machine readable form using "Wordwise", "Minitext Editor" or other means. If you use cassette, please include a backup copy at 300 baud.

HINTS

There are prizes of £5 and £10 for the best hints each month.

Please send all editorial material to the editorial address below. If you require a reply it is essential to quote your membership number and enclose an SAE.

Editorial Address: BEEBUG, PO Box 50, St Albans, Herts

BEEBUG MAGAZINE is produced by BEEBUG Publications Ltd.
Editor: Mike Williams. Production Editor: Phyllida Vanstone.
Technical Assistants: Alan Webster, John Waller and David Fell.
Managing Editor: Sheridan Williams.
Thanks are due to Lee Calcraft, Adrian Calcraft, John Yale, Tim Powys-Lybbe and Matthew Rapier for assistance with this issue.

NEW BEEBUG BINDERS

We have produced a new hard-backed binder for the BEEBUG magazine. These binders are dark blue in colour with "BEEBUG" in gold lettering on the spine and allow for the whole of one volume of the magazine to be stored as a single reference book.

The new binders now have a slightly larger capacity to accommodate 10 thicker BEEBUG magazines, and are supplied with 12 wires. This enables the index and the latest copy of the supplement to be included within the binder if required. Individual issues may still be easily added and removed, allowing for the lastest volume to be filed as it arrives.

The price of the new BEEBUG binder is £3.90 including VAT. Please add 50p post and packing for delivery within the UK. Overseas members please send the same amount, this will cover the extra postage but not VAT.

BEEBUGSOFT, PO BOX 109, High Wycombe, Bucks, HP10 8HQ.

BEEBUG NEW ROM OFFER

1.2 OPERATING SYSTEM

A special arrangement has been agreed between Acorn and BEEBUG whereby BEEBUG members may obtain the 1.2 operating system in ROM at the price of £5.85 including VAT and post and packing. The ROM will be supplied with fitting instructions to enable members to install it in their machine. If the computer does not subsequently operate correctly, members may take their machine to an Acorn dealer for the upgrade to be tested, which will be done at a charge of £6.00 plus VAT. This charge will be waived if the ROM is found to have been defective. If the computer has been damaged during the installation processs, the dealer will make a repair charge.

ADDRESS FOR 1.2 OS:
ROM Offer, BEEBUG. PO Box 109, High Wycombe, Bucks, HP10 8HQ

THE BEEBUG MAGAZINE ON DISC AND CASSETTE

The programs featured each month in the BEEBUG magazine are now available to members on disc and cassette.

Each month we will produce a disc and cassette containing all of the programs included in that month's issue of BEEBUG. Both the disc and the cassette will display a full menu allowing the selection of individual programs and the disc will incorporate a special program allowing it to be read by both 40 and 80 track disc drives. Details of the programs included in this month's magazine cassette and disc are given below.

Magazine cassettes are priced at £3.00 and discs at £4.75.
SEE BELOW FOR FULL ORDERING INFORMATION.

This Month's Programs Include:

Vol. 3 No. 5 cassette/disc. Matching Pairs game to keep you thinking, Fireworks Display, examples of animated graphics from our 'Beginners Start Here' article, Kitchen Chaos – an unusual action game, a program to test out your disc drives, A Dotty Grid for graphics and screen layout, a Flexible Menu Routine from this month's BEEBUG Workshop, a utility for analysing the performance of Basic programs, the winning entry in the Odd Factors Brainteaser competition, and as an extra bonus a colourful arcade game in machine code called Astro Wars.

MAGAZINE DISC/CASSETTE SUBSCRIPTION

Subscription to the magazine cassette and disc is also available to members and offers the added advantage of regularly receiving the programs at the same time as the magazine, but under separate cover.

Subcription is offered either for a period of 6 months (5 issues) or 1 year (10 issues) and may be backdated if required. (The first magazine cassette available is Vol. 1 No. 10; the first disc available is Vol. 3 No. 1.)

MAGAZINE CASSETTE SUBSCRIPTION RATES

6 MONTHS	(5 issues) UK £17.00 INC... Overseas £20.00 (No VAT payable)
1 YEAR	(10 issues) UK £33.00 INC... Overseas £39.00 (No VAT payable)

MAGAZINE DISC SUBSCRIPTION RATES

6 MONTHS	(5 discs) UK £25.50 INC... Overseas £30.00 (No VAT payable)
1 YEAR	(10 discs) UK £50.00 INC... Overseas £56.00 (No VAT payable)

CASSETTE TO DISC SUBSCRIPTION TRANSFER

If you are currently subscribing to the BEEBUG magazine cassette and would prefer to receive the remainder of your subscription on disc, it is possible to transfer the subscription. Because of the difference between the cassette and disc prices, there will be an extra £1.70 to pay for each remaining issue of the subscription. Please calculate the amount due and enclose with your order.

ORDERING INFORMATION

Please send your order to the address below and include a sterling cheque. Postage is included in subscription rates but please add 50p for the first item and 30p for each subsequent item when ordering individual discs or cassettes in the UK. Overseas orders please send the same amount to include the extra post but not VAT.
SEND TO:

BEEBUGSOFT, PO BOX 109, HIGH WYCOMBE, BUCKS, HP10 8HQ

Printed in England by Staples Printers St Albans Limited at The Priory Press.

ISSN 0263-7561