

## INDEPENDENT NATIONAL USER GROUP



FOR THE BBC MICROCOMPUTER

**BEEBUG NEWSLETTER**

VOLUME 1

NUMBER 5

SEPT 1982

## CONTENTS

Editorial	2	<u>HINTS AND TIPS</u>	
Delivery dates	3	Integer Variables	9
New User Guide Errata	4	16k Test	10
Verifying Programs	5	Sound Buffer	17
Operating System Notes	6	Step Through Listings	17
BEEBUG Technical Editor	9	Printer Interfacing	19
Unexplained Errors-Explained	10	Trapping Errors	24
Multi Coloured Characters	11	REMs in DATA statements	27
Software Competition Results	14	Out of Sequence AUTO	27
String Search Program	15	Recall of Function Keys	27
Screenplay (16k)	16	Black on White	30
Postbag	17	Corrupt Tapes	30
Delivery Complaints Letters	18		
ROM Charge	19		
Multi Coloured Beeb	20		
Logic (Part I)	21		
Points Arising	25		
User Key Update	26		
Procedure/Function Library	28		
Government's Micro Electronics Project	30		
Higher/Lower (16k) (GAME)	31		
Hangman (16k) (GAME)	33		

**BRITAIN'S LARGEST SINGLE-MICRO USER GROUP**

## EDITORIAL

### Exit BL

As from 1st September the distribution of the BBC microcomputer will be handled by Vector Marketing, Denington Estate, Wellingborough, Northamptonshire, NN8 2RL (Tel: 0933 79300). This comes after a long period of admitted problems at BL Marketing, and we hope that this will provide the long awaited cure. The new company, a subsidiary of Acorn Computers, represents a regrouping rather than a totally new company, and the chief executive of BL, Peter Goater, is now the chief executive of Vector - but we are assured that the problem was caused by equipment and not by personnel.

As a token of their reorganisation, they have sent out another despatch schedule to all those people that they estimate have outstanding orders, together with a confirmation form which is to be returned. The dates schedule looks impressive, but it is only any good if it can be adhered to. If it is not adhered to in your particular case, we suggest that you write firstly to Vector about it, then to Acorn, and if you still meet with no success, to the BBC (BBC Enterprises, BBC Microcomputer, Broadcasting House, London, W1A 1AA). We have taken this attitude because if this schedule is not adhered to it just represents a further insult to those waiting for long overdue deliveries.

### ROM Replacement Charge

In the July issue we reported that the replacement charge for the 1.0 operating system ROM would be £10. This is the amount that the BBC currently quote when you write to them about the ROM. Acorn however, who in earlier conversations with us did not correct us when we asked them why they were charging £10 for the ROM, have sent an information sheet (ref: LH/MS dated 3/8/82) to their dealers which says that: "The cost of this upgrade will be in the region of £15 when the operating system 0.1 is returned, this price is however provisional and final costings may vary".

In our initial discussions with Chris Curry of Acorn, we were told that a "nominal" charge would be made for the replacement. We do not regard £10 or £15 as nominal, and have expressed our feelings both to the BBC, and to the directors of Acorn. Neither body responded very favourably to our comments. We take the view that aspects of the 0.1 system such as the cassette filing bugs take it below the standards that one would expect in any computer system. A good number of members also take this view and have written variously to us, to Acorn and to the BBC about it, and we have published extracts from a couple of their letters in our postbag. Some of those who wrote are considering the idea of taking the BBC to the small claims court for the cost of replacing the ROM. We are taking legal advice on this matter, and will report back next month.

In the meantime, if you feel strongly about this question you should write to Acorn and the BBC (address given above). We name both organisations because we have been unable to ascertain who is ultimately responsible on this issue.

### "Acorn User"

The "Acorn User" is published by Addison-Wesley for Acorn Computers Ltd. as the "official magazine for users of the Acorn Atom and the BBC Microcomputer system.....", and claims to give "Authoritative information on all new Acorn products". There must of course, be a slight implication here that BEEBUG is neither official nor authoritative, so it is nice if we can scoop them on an important issue, as we did with the "Cassette Bugs fix" in our July issue. This fix has now been approved by Acorn, and will probably be published in the "Acorn User" number 2, in essentially the same form (so you don't need to retype their version). Several

readers also wrote to us about the glowing reviews in the "Acorn User" of Acornsoft's programs. The "Acorn User" says on page 22 that "Defender" will work on a model A. Having ordered the program many were dismayed to find that it actually required a model B (or a model A with memory and 6522 VIA upgrade).

On the question of the memory upgrade, the "Acorn User" carries an interesting comment, in their "Acorn Answers Your Questions" page, they write: "more experienced users can upgrade the memory" - and then they proceed to explain how to do it. The statement is interesting because as an "official" statement from Acorn it sets a useful precedent that appears to conflict with Acorn's policy stated elsewhere. Information sheet LH/MS of 3/8/82 quoted earlier, states that: "The 1.0 operating system upgrade must be fitted by one of Acorn's approved service centres, this is in line with our general policy that ALL MODIFICATIONS SHALL BE MADE AND TESTED BY QUALIFIED PERSONNEL" (Our caps). This is not the same thing, and one wonders which is correct, especially in view of a report by a member that a dealer informed him that his guarantee was invalid because he had fitted a memory upgrade himself. We would be grateful for clarification on this matter from Acorn.

### Magazine Matters

As you may have noticed elsewhere in the magazine, we are looking for a full time technical editor to help with the running of the magazine. We are also now in a position to pay contributors for substantial articles - so if you have any ideas please contact us. Also if you write to us requiring a reply, it is essential to enclose a stamped addressed envelope and to quote your membership number (the number that appears on the label of our mailings eg Y39775). Finally, please do not try to telephone the editorial department - we are only geared up for postal communication.

BBC MICROCOMPUTER SYSTEM		
Code	Description	Despatch Schedule
ANA01	Model A	Available on 3 weeks delivery
ANA02	Model A + Econet Interface	Production to commence in September
ANB01	Model B	February orders despatched by 7 August March ..... 21 August April ..... 28 August May ..... 4 September June ..... 11 September July ..... 18 September August ..... 25 September Thereafter 4 week delivery
ANB02	Model B + Econet Interface	Production to commence September
ANB03	Model B + Disk Interface	Up to February orders despatched by 28 August Up to April ..... 11 September Up to June ..... 18 September Up to August ..... 25 September Thereafter 4 week delivery
ANB04	Model B + Disk + Econet Interface	Production to commence in September
ANC01	Second Processor - 6502	Production to commence in November (provisional)
ANC02	Second Processor - Z80	Production to commence in November (provisional)
ANC03	Second Processor - 16 bit	Production to commence in February 1983 (provisional)
AND01	Single Drive (100)	Available on 4 weeks delivery, existing orders to be in line with ANB03 despatches
AND02	Dual Disk Drive (800)	Production to commence in August
ANE01	Teletext Receiver	Production to commence in October
ANE02	Prestel Receiver	Production scheduled for first quarter 1983
ANF01	12" Monochrome Monitor	Available on 3 weeks delivery
ANF02	14" Colour Monitor	Available on 3 weeks delivery
ANF03	Cassette Recorder	Available on 4 weeks delivery, existing orders to be in line with computer despatches
ANG01	3" Video Lead UHF Screw to BNC	Available on 3 weeks delivery
ANH01	Games Paddles per pair	Production to commence in August
ANJ01	Extra Copies of User Guide	Despatch of New Guides in progress as deliveries from printer received

We are grateful to Acorn's press officer for permission to publish this.

## NEW USER GUIDE ERRATA

Everyone should have a copy of the New User Guide (N.U.G.) by now, and in our view this forms a reasonably well assembled user REFERENCE manual. Where it falls down is that it is not designed as a beginners GUIDE to the machine. This is a shame in a way, because of the supposed link between the BBC machine and the computer literacy project. However, you cannot please everybody; and the real problem arises because the BBC machine offers so many facilities to the user.

Returning to the N.U.G. itself, the one real point where it fails to be clear is on the limitations of operating system 0.1. Under the list of FX calls, there is an indication of which ones will not work on 0.1, but in the body of the manual the various features of 1.0 are discussed without any indication that they will not work on 0.1 - and of course 97% of all Beeb users are currently using a 0.1 system.

### ERRORS

The N.U.G. is relatively free from errors, but inevitably a few have been detected, and we bring these to you courtesy of Acorn/BBC.

- p 77: Trailing semi-colon missing in VDU 23;8202;0;0;0;
- p 403: In example of appending programs, >LOAD ONE should read >LOAD"ONE"
- p 439: &EA(228) should read &E2(226)
- p 489: Lines do NOT start in double height!

Character 255 is a solid graphics character, not "backspace & delete"

- p 497: Codes 8A and 8B on the cursor-keys should be swapped.

There are several other typographical errors and less important factual errors. A significant omission from the N.U.G. is that there is no mention in the section for keyword STR\$ that it will produce a hexadecimal string (rather than decimal) if immediately followed by a tilde (~). I.e. PRINT STR\$(100) will produce the 2-character string "64". This can be useful.

## BITS AND PIECES FOR A's AND B's CHEAP, FAST and EFFICIENT SERVICE from

**PEDAGOG COMPUTER SERVICES** 11 Fairbridge Road, London N19 3EW

### MONITORS

RGB the Rolls Royce of Colour Monitors for your B, 14" screen ... .. £250.00  
RGB lead for above ... .. 10.00

PAL combined signal Monitor for A or B, 14" screen ... .. 250.00  
PAL lead for above ... .. 8.00

Monochrome (Green Screen) Monitor for A or B, 12"screen ... .. 75.00  
Lead for above ... .. 8.00

PRINTERS We have standardised on this excellent printer:  
EPSON MX 80 F/T/3 (Tractor/Friction/Graphics) ... .. 340.00  
Interface Cable for printer ... .. 18.00

### USER PORT USE your User Port!

1 Metre lead with connector to 'B' at one end, Terminal Board at other end, plus Data Sheet with connection details & ideas ... .. 25.00

ALL PRICES EXCLUSIVE OF VAT @ 15% & P. & P. at cost

TEL 0485/40604 or ABOVE ADDRESS

\*FOR INCLUSION ON OUR MAILING LIST, please send us your name and address.

---

## VERIFYING PROGRAMS

---

One subject which keeps on recurring in letters is that of verifying programs when saved on cassette. There are several ways to verify a program but nearly all seem to have drawbacks. Since most people find one way and insist on sticking to it, I think it necessary to describe here just what is wrong with the various methods used.

Firstly I will deal with the most commonly used method of verifying, that supplied by the operating system:

(1) \*CAT

This command is one which I presume to be designed primarily for use with discs, where a user would obtain an almost instantaneous catalogue of all the programs on a disc (though not, unfortunately, a printout?). When used with cassette to verify programs, it will list each block of a program as it finds it, but it only checks individual blocks rather than the program as a single entity. It also fails to report the now infamous bug that occurs on block zero, which subsequently makes a program completely unreadable (though the bugs fix supplied in the July issue fixes this one).

(2) LOAD ""

Another favourite method is to try loading the program back again as soon as it has been saved. The theory being that if anything goes wrong then the user has only to type OLD to retrieve the original which is still in memory. This would be fine if the theory were correct! Once a program has started "loading" in, it is overwriting the original version. If anything goes wrong after the "loading" message has been given, then it will be necessary to use 'ESCAPE' or 'BREAK' to get back to the BASIC command mode. When this happens, the message "Bad program" usually appears and you CANNOT get the old version back...at all! So, although this method will detect block zero faults, it will not allow recovery when other faults occur.

(3) \*CAT followed by LOAD ""


Of course you could use the \*CAT method followed by the LOAD"". This will detect all possible errors and is safe to use without losing programs. It is safe because a missing block zero which is not detected by \*CAT would prevent the faulty program from even starting to overwrite the original with the LOAD"" command; but seeing that it would not load would indicate a badly recorded program. The trouble is that although this works, the cassette must be played twice, being rewound inbetween, thus taking rather a long time.

(4) \*LOAD"" 8000

No, that's not a printing error... the \*LOAD command is not the same as the LOAD command, it is designed for loading machine code programs starting at any given address in memory, but it still works perfectly for BASIC programs. The difference which makes it useful is the ability to store the incoming program at any given address. This is specified by the number (in Hex) at the end of the command. By using 8000 as the address, the incoming program will be loaded on top of the resident ROM. Of course, because this is Read Only Memory, nothing is actually overwritten! So that your program remains untouched where you left it. When loading in this way all the errors will be detected just as if performing a normal LOAD and failure to load means that the program is badly saved. In which case, simply 'ESCAPE' from the loading, with your program intact, re-save the program and try again.

CONCLUSION :- The last method shown appears to be without fault and by far the best method to ensure your program will load back after saving it. NOTE: there MUST be an asterisk in front of the LOAD command.

Many thanks to all those members who have written in with their suggestions and especially to those who suggested \*LOAD"" 8000.

Rob Pickering. 

---

## OPERATING SYSTEM NOTES

---

Operating system 1.0 which we have mentioned on a number of occasions in BEEBUG has already been superseded, and according to sources within the BBC, O.S. 1.0 will now not be ROMmed. Some copies of 1.0 have gone out in EPROM form with machines supplied with a disc interface, but the order with Hitachi for the 1.0 operating system has been halted. The system to be ROMmed will now be 1.1 or higher, and of course this will considerably delay the date of supply of the new system.

At BEEBUG we now have access to a 1.1 operating system, and we are using it to validate all our software - both for publication in the magazine, and for putting into the software library. We will include with each program in the magazine a note saying which systems it has been checked on. To date the two problems that we have encountered are:

1. The use of \*FX4,2 which has different effects in the two operating systems - see July p30 for details.
2. The use of the TAB command after a VDU5 call. A 'feature' of O.S. 0.1 was that you could use the TAB command after executing VDU5 (which joins the text and graphic cursors), even though the documentation said otherwise. Many people discovered this (or more likely did not notice that TAB was not supposed to operate under these circumstances), and made full use of it. The way to make programs 'portable' from one operating system to another in this respect is either to replace TAB(x,y) by MOVE(x,y) (remembering to change the values of x & y accordingly) or to cancel VDU5 with VDU4 before using TAB (don't forget to re-instate VDU5 afterwards).
3. One other respect in which the new operating system differs from the old is in the use of VDU1. This call, as we mentioned in the July issue, can be used to send control characters to the printer. In O.S. 0.1 this could be done even when the printer was NOT enabled with VDU2 (or control-B). In O.S. 1.0 and later versions, VDU1 will apparently only function once the printer is enabled.



# Happy Memories

4816 100ns memory upgrade IC's

1 off	25-99	100 up
2.45	2.10	1.95

1 off price less 10% for Beebug members

Please add VAT. No Postage on sets of 8

Happy Memories, Gladestry, Kinston,  
Herefordshire. HR5 3NY

Tel: (054 422) 618/628

Access and Visa taken over the 'phone.

# IDEAS ON COLOURING AND SHADING

by John Yale

We show here how to extend the range of colours produced by the Beeb, and also how to create shading and stripe effects using a previously undocumented feature of the GCOL command. To make the most of these effects you will need a colour display, though useful effects can also be created in black and white (see especially part 2).

## Part 1— GENERATING COLOURS BY MIXING

It is possible to display more colours than would at first appear possible in any mode, by a colour mixing process known as "dithering". If an area of the screen is filled by a pattern of dots of two colours, then the eye will mix these colours, forming an intermediate one; [though you may need to step back from the screen a few paces for a really good dither. Ed.]

The program below demonstrates the principles. Mode 2 is used in order that all eight colours can be shown, for model A computers use mode 5 and add the lines

```
112 VDU 19,0,2;0;
114 VDU 19,3,4;0;
```

to replace black and white with green and blue. Other colours may be tried by changing the 2 and 4.

Line 120 defines a new character which has alternate dots in the foreground and background. This is because &55=01010101 and &AA=10101010 in binary. Each binary bit represents one point in the character matrix '1' for foreground and '0' for background.

The loop at line 140 selects eight foreground colours whilst the loop at line 160 selects eight background colours. Line 150 makes the pattern threetimes bigger by printing each character three times vertically.

Line 190 uses VDU rather than the equivalent PRINT CHR\$(224) because it is shorter. Two characters are printed to make the pattern wider. By changing the user defined character to have more or less foreground bits set, different shades of colour may be achieved.

```
100 REM Colour dithering demonstration
110 MODE 2
120 VDU 23,224,&55,&AA,&55,&AA,&55,&AA,&55,&AA
130 CLS
140 FOR foreground=0 TO 7
150   FOR line=1 TO 3
160     FOR background=128 TO 135
170       COLOUR foreground
180       COLOUR background
190       VDU 224,224
200     NEXT background
210   PRINT
220   NEXT line
230 NEXT foreground
240 VDU 20
```

## Part 2— NEW COLOURS, NEW SHADES AND STRIPE EFFECTS USING THE GCOL COMMAND

J.A.Knaggs, Alan Pemberton, L.H.Sellick and others have noted a very interesting previously undocumented colour facility on the Beeb. J A Knaggs writes: The first parameter of the GCOL command determines how subsequent graphics colours are to be handled. According to the User Guide it can take values of 0 to 4. In fact all numbers from 0 to 255 give different effects which vary with the mode of operation and the colour used. 'New' colours can be produced (e.g. grey) which

actually consist of alternate vertical stripes of different colours. The stripes can be of different widths, and one or both of the colours can be flashing. In a two-colour mode patterns of black and white stripes can be obtained. The program below demonstrates some of the effects that can be produced.

```

5 REM Undocumented GCOL effects
6 REM   by J.A.Knaggs
10 M%=2
20 MODE M%
30 PRINT TAB(1,1);"MODE ";M%
40 FOR Y%=6 TO 0 STEP -1
50   FOR X%=0 TO Y%
60     READ A%,K%
70     GCOL A%,K%
80     I%=120+X%*144+(6-Y%)*72
90     J%=1016-Y%*144
100    MOVE I%,J%
110    MOVE I%+144,J%
120    PLOT 85,I%+72,J%--144
130 NEXT: NEXT
140 DELAY=INKEY(500)
150 RESTORE
160 IF M%=2 THEN M%=4 ELSE M%=2
170 GOTO 20
1000 DATA 35,9,15,8,19,0,14,15,13,15,20,6,9,0
1010 DATA 21,10,15,9,14,14,5,11,9,1,20,7
1020 DATA 43,5,14,13,9,2,5,12,59,7
1030 DATA 9,3,15,13,43,1,5,9
1040 DATA 14,10,14,9,14,8
1050 DATA 61,14,20,1
1060 DATA 46,4

```

### Full GCOL Repertoire

The effects that have been discovered seem so interesting, and appear to have so much potential for creating unusual visual effects, that we have written a further program to work systematically through the repertoire of the GCOL command.

Solid vertical bars are plotted in colours 0 to 7 and then crossed horizontally with bars of the same colour but with a different GCOL parameter. Pressing return will increment the first GCOL parameter by one, entering a number (0 to 255) will set the parameter to that value.

```

10 REM Extra GCOL modes.
20 REM Investigation by
30 REM J.Yale 12-Aug-1982
40 REM based on an idea
50 REM by J.A.Knaggs
60 ON ERROR GOTO 590
70 MODE 2
80 PROCinit_screen
90 REPEAT
100  FOR G1 = 0 TO 255
110    PROCnew G1
120    PROCvert_boxes
130    PROCchoriz_boxes(G1)
140    NEXT G1
150 UNTIL FALSE
160 END
170
180 DEF PROCinit_screen
190 LOCAL I
200 VDU 28,0,0,19,0
210 VDU 24,0,0;1279;960;
220 REM Map colours 8-15 to 0-7
230 REM prevents flashing when inverting
240 FOR I = 0 TO 7
250   VDU 19,I+8,I;0;
260 NEXT I
270 ENDPROC
280
290 DEF PROCvert_boxes
300 LOCAL I
310 FOR I = 1 TO 7
320   GCOL 0,I
330   PROCbox(128*I,0,128*I+64,960)
340 NEXT I

```



```

350 ENDPROC
360
370 DEF PROCchoriz_boxes(G1)
380 LOCAL G2
390 FOR G2 = 0 TO 7
400   GCOL G1,G2
410   PROCbox(0,128*G2,1279,128*G2+64)
420 NEXT G2
430 ENDPROC
440
450 DEF PROCnew G1
460 INPUT TAB(10,0),"GCOL ",A$
470 IF A$<>" " THEN G1 = VAL(A$)
480 CLG
490 PRINT TAB(0,0),"GCOL ";G1;

```

```

500 ENDPROC
510
520 DEF PROCbox(X0,Y0,X1,Y1)
530 MOVE X0,Y0
540 MOVE X0,Y1
550 PLOT 85,X1,Y0
560 PLOT 85,X1,Y1
570 ENDPROC
580
590 REM Escape routine
600 MODE 7
610 REPORT
620 PRINT" @ line ";ERL
630 END

```



## HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

### Integer Variables

Peter Drysdale writes to ask the significance of the % added to the end of variable names in many programs. Any variables ending in % can only be used to hold integers or whole numbers. The result of this is that arithmetic may be performed very much faster. For example a FOR NEXT loop executes about THREE TIMES AS FAST with an integer index, as shown by the following program.

```

100 TIME=0
110 FOR I=1 TO 1000
120 NEXT I
130 PRINT TIME/100

```

```

140 TIME =0
150 FOR I%=1 TO 1000
160 NEXT I%
170 PRINT TIME/100

```

The range of an integer variable is +/- 2147483647. Exceeding this range in an addition or subtraction operation is not detected and incorrect answers will result. Multiplication does result in the 'Too big' error message if this range is exceeded.

## BEEBUG TECHNICAL EDITOR

BEEBUG requires a full-time technical editor to work on BEEBUG magazine, to start as soon as possible.

Applicants should have a good familiarity with the BBC machine and be able to demonstrate an ability in technical writing.

The work will be varied, and include both writing and editing articles for the magazine; it will carry a large degree of responsibility for the content of the magazine, under the direction of the managing editors.

The successful candidate will be based in St Albans, and will, at least in the first instance, be employed on a freelance basis, albeit at a full-time rate (actual rates of pay depending on experience). Applications should be made in writing to the editorial address, and should include a curriculum vitae as well as the names of two referees. Please also give a telephone number if possible.

Applications close on 10th October.

---

## UNEXPLAINED ERRORS — EXPLAINED

---

In this, the first of two articles on debugging, Rob Pickering explains one of the most common errors encountered when typing programs in from printed listings.

One of the most common subjects in letters received from members, is that of the programs listed in BEEBUG magazine. Generally the content of a letter is something along the lines of:

"...having spent hours typing in the MOONLANDER program from issue-1, I cannot seem to get anything except "No such variable..." error messages. When I correct one, I get one somewhere else. Can you tell me if there are bugs in the program listing? If there aren't then what am I doing wrong ?..."

I have worked through many such queries and now have a pretty good idea of what goes wrong. Firstly, there are no errors in the listings published, they were listed from the working programs. About 90% of the errors people encounter are due to just one cause:- missing out vital spaces. Because of the continuing large number of letters on this subject I have decided to give a full description of the problem here.

The BBC micro allows variable names to be of any length and may include embedded keywords, although they cannot start with a keyword. A keyword is one of the BASIC commands such as GET, THEN, RUN, etc. Because they may be embedded within variable names it is possible on BBC BASIC to have a variable name "NAMELIST" or "BRAND" where "LIST" and "AND" are not read as commands but simply part of the variable names. BUT, this is not true of other micros such as PET, Apple, etc. where the command words would be picked out and executed.

The most common place where this causes problems is with the "THEN" keyword. The following lines will help to illustrate:

```
10 IF A=BTHEN GOTO 100
10 IF A=B THEN GOTO 100
10 IF A=B%THEN GOTO 100
```


Although all of these forms would be acceptable on most micros, BBC BASIC interprets the first of the lines as having a variable called "BTHEN" where the "THEN" is embedded in the overall variable name. The second line is interpreted as would be expected with the space after variable "B" indicating that this is the end of the variable name. However, if an integer variable is used, the percentage sign will indicate the end of the name so that a further space is NOT required. Both the second and third forms are acceptable.

My personal preference is to use variable names in lower case characters, but this can make the errors look even less like errors. Consider the following:

```
10 IF abcd=efghTHEN PRINT"yes"
```

This is still an error since it is allowable to mix upper and lower case characters in variable names, such that the above is still interpreted to have a variable called "efghTHEN", even if it looks clear to you, it isn't to the computer!

To conclude I would like to point out that I am certainly not knocking the variable names on the BBC micro; I think they are the best of any version of BASIC. However, to the unwary user these problems are not at all clear and I don't think that they have been previously discussed.

Rob Pickering. 

---

### HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

---

#### 16k TEST

Terry Bromilow sent in a useful tip to find if a program will run in 16k. Add the line: DIM DUMM%(4088) which takes exactly 16k of memory.

Terry also notes that the current mode is stored at &367 in OS 0.1, so to find out which mode you are in: mode = ?&367

---

Program tested on 0.1 and 1.1 O.S. **MULTI COLOURED CHARACTERS** Program tested on 0.1 and 1.1 O.S.

---

John Yale and David Graham explain how to create and move multicoloured characters with the aid of the VDU 5 command. This technique has obvious uses if you are viewing in colour, but even with black and white it can allow different shades of grey within a character, and the provision of flashing parts. The first demonstration program is written for Mode 2, and therefore requires a 32k machine. The second uses Mode 5 and will operate in 16k.

### THE PRINCIPLES INVOLVED

Normally a user defined character can only be plotted in two colours, the current background and foreground. Any attempt to print characters on top of each other to obtain more colours results in the new character totally replacing the old. However after VDU 5 is used to join the graphics and text cursors together, the rules change as follows.

- \* Characters are plotted at the graphics cursor position and the text cursor is turned off. This enables text or user defined characters to be placed anywhere on the screen by the MOVE command. Contrary to the specification in operating system 0.1, the TAB command still works when VDU 5 is called. On OS 1.0 however, the TAB command does NOT function, and use must be made of the MOVE command to move the text cursor to a given position.
- \* Text and graphics are only produced within the graphics window.
- \* The colour used is obtained from the last GCOL command, not the COLOUR command.
- \* Only the foreground colour is plotted, not the background, allowing characters to be overprinted. This last point allows the plotting of multi-coloured user-defined characters as shown by the following program.

```
10 REM Multi colour characters
20 REM J.Yale  11 Aug 1982
30
40 ON ERROR GOTO 310
50 REM Define characters
60 VDU 23,224,&FF,&81,&81,&81,&81,&81,&81,&81,&81,&81,&81,&FF
70 VDU 23,225,&00,&7E,&42,&42,&42,&42,&7E,&00
80 VDU 23,226,&00,&00,&3C,&24,&24,&3C,&00,&00
90 VDU 23,227,&00,&00,&00,&18,&18,&00,&00,&00
100
110 MODE 2
120 VDU 5
130 MOVE 0,900
140 FOR I = 224 TO 227
150   GCOL 0,I-223
160   PRINT"Character ";I;" - ";CHR$(I)
170 NEXT
180 PRINT"SPC(10);
190 FOR I = 1 TO 4
200   GCOL 0,I
210   VDU 223+I,8
220 NEXT
230 PRINT
240 REPEAT
250   FOR I = 1 TO 4
260     VDU 19,I,8+I;0;
270     WAIT=INKEY(500)
280     VDU 19,I,I;0;
290   NEXT I
300 UNTIL FALSE
310 VDU4
320 END
```

Four characters are defined, which are non-overlapping squares of four different sizes. Users without colour monitors may find the following definitions easier to see:

```
VDU 19,224,&C0,&C0,&C0,&C0,&C0,&C0,&C0,&C0
VDU 19,225,&30,&30,&30,&30,&30,&30,&30,&30
VDU 19,226,&0C,&0C,&0C,&0C,&0C,&0C,&0C,&0C
VDU 19,227,&03,&03,&03,&03,&03,&03,&03,&03
```

The characters are plotted individually and then on top of each other giving the four colour character. Lines 50-90 define the four characters; lines 140-170 display them separately; and lines 180-220 overlay them to produce a single four-colour character. Lines 240 onwards are optional and just have the effect of repeatedly changing the colour of each segment by using the VDU 19 command. See below for further details. Note the use of the VDU 8 (cursor left) code in line 210; MOVE could have been used instead but would have required an extra command.

### MOVING ALIENS WITH FLASHING EYES

Because only the foreground (and NOT the background) is printed when characters are printed under VDU5, you cannot erase a character by overprinting it with a space (since a space is all background, and no foreground). There is however, a simple solution if you wish to erase a two-colour character. You define a character as a solid block, using the VDU23 command, and then print it in colour zero (ie black) over anything to be erased.

The program below uses this technique to move a two-coloured character across the screen. The character concerned is a red alien with yellow eyes, and the details below describe firstly how the character may be incorporated into a string variable with the appropriate colour commands, and finally how to erase it so as to create the impression of movement across the screen.

Line 50 defines the alien's shape, and line 60 its eyes - (these were drawn out using the BEEBUGSOFT "Redefine" program). The character defined in line 70 is the block to be used for erasing. Lines 110 and 120 use the VDU18 command to set the graphics colour. CHR\$18 has the same effect as VDU18 (except that it can be put into a string variable as we have done here). Then CHR\$0 followed by CHR\$1 has the effect of sending VDU 18,0,1 or GCOL 0,1. This just sets the foreground to red. The whole point of line 110 is that it allows us to incorporate the colour change command into a character string, so that we can then define a single character (CH\$) which contains the whole 2-colour alien. Line 120 does the same with the 'yellow' command, and line 130 defines the whole character as follows: RED\$ means print whatever follows in red, CHR\$224 is the alien's body - so that it comes out in red. CHR\$8 means backspace by one (so that the eyes appear in, and not adjacent to the body), and then the last 2 elements in the line cause the eyes to be printed in yellow.

If you run the program below you will see the two-colour alien produced by line 190 - which is simply MOVE 640,864:PRINT CH\$ - note that the MOVE command has been used here in place of TAB, since as mentioned above, the TAB command is disabled by VDU 5 in versions of the operating system later than 0.1. If you now press the space-bar, you will see an alien move from left to right. Leaving the space-bar depressed will make it move faster. The routine for moving it appears between lines 230-270, and in line 260 the alien is printed with a blank space to its left (from BLANK\$), and this blank space erases the previously printed alien. BLANK\$ is just a solid block, printed in colour zero (which is the background colour). The variable BLANK\$ is defined in line 140.

To get the alien's eyes to flash blue/yellow (so making them effectively piercing even in black and white), insert the line 185 VDU 19,2,12,0,0,0. This makes logical colour 2 produce colour number 12 which is flashing blue/yellow. See under COLOUR and VDU19 in the new user guide for details.



```

10 REM TO PRODUCE MOVING TWO-COLOUR ALIEN
20 :
30 REM DEFINE CHARACTERS
40 :
50 VDU23,224,24,60,126,219,90,126,66,129
60 VDU23,225,0,0,0,36,36,0,0,0
70 VDU23,255,255,255,255,255,255,255,255,255
80 :
90 REM PRODUCE CHARACTER STRINGS
100 :
110 RED$=CHR$18+CHR$0+CHR$1
120 YELLOW$=CHR$18+CHR$0+CHR$2
130 CH$=RED$+CHR$224+CHR$8+YELLOW$+CHR$225
140 BLANK$=CHR$8+CHR$18+CHR$0+CHR$0+CHR$255
150 :
160 REM DISPLAY CHARACTERS
170 MODE5
180 VDU5
190 MOVE640,864:PRINTCH$
200 GCOL0,3:PRINT "" PRESS SPACE BAR"
210 REPEAT
220 REPEAT UNTIL GET$=""
230 MOVE0,512:PRINTCH$;
240 FORC=0TO18
250 X=INKEY(10)
260 PRINTBLANK$;CH$;
270 NEXT
280 UNTIL TRUE = FALSE

```

# micro aids

SYSTEM CARE PRODUCTS

at home with the computer

SPECIAL INTRODUCTORY OFFER TO BEEBUG MEMBERS !!!

'micro cover' - protect your machine with this well designed, quality, polyester/cotton cover.

£2.95

inc.

perfect fit  
washable  
fawn colour

mc 100 BBC  
mc 105 Atom  
state which model

'cable care' - improved performance, increased safety, enhanced appearance with this range of cable management products.

£2.95

inc.

easy to fit  
neat & tidy  
reusable

sample pack  
exclusive to  
BEEBUG members

Flowchart stencil FREE with orders over £5.00

MICRO AIDS 2 Boston Close, Culcheth, Warrington. WA3 4LW

## SOFTWARE COMPETITION RESULTS

We had over 50 entries for our first software competition, and the standard of entry was generally good, though if anything, not enough use was made of the Beeb's special features such as functions, procedures, repeat loops and long variable names. We have awarded prizes worth over £550 in all, and the winners names are given below. You should be seeing some of these entries in the magazine or software library, and some of the better entrants have been offered royalties for writing further programs.

It is always invidious to judge competitions such as this, and judging certainly was not easy. We used five criteria for the purpose, though of course there is subjectivity even in this. They were as follows:

1. Presentation. This included the program's visual appeal, and the ease with which it could be used.
2. Documentation. This could have been included in the program itself, or on paper. There are two aspects to this - firstly does it enable the user to use the program; secondly is there sufficient information on variables and structure to enable enhancements or corrections to be made.
3. Programming. Use of procedures, functions, meaningful variable names, and sensible program structure are all important here.
4. Ingenuity & Algorithm. There are many ways in which a program can be coded. Some of the ways are appallingly inefficient. Some programmers can be exceedingly ingenious in solving problems, and this category was intended to reward them.
5. Errors. All sorts of errors were considered here, not just syntax errors. For example when the program finishes, does it restore all the normal functions such as 'cursor on', 'auto key repeat', 'windows' etc. Was input data validated so as to reject nonsense input. When a numeric response is required are the alphabetic keys disabled. (See INPUT function in July issue for how to do this). Was ON ERROR used.

Due allowance was, of course, made when we judged certain criteria to be inappropriate to a given program.

★★★★★	<u>PRIZEWINNERS</u>	★★★★★
<u>PRIZE</u>	<u>NAME</u>	<u>PROGRAM</u>
£100	M J Taylor	Starfire
£50	M Savva	Tutor
£50	Wg Cdr Carroll	Sums
£25	J R Buchanan	Snake
£25	I Sinclair	Text editor
£25	D J Coldicott	Cow Pat
£25	E F Christie	Manhole
£25	R Lowe	Shapematch
£25	R Whitehead	Invaders
£25	K Abdalla	Character
£25	R Bailey	Higher/Lower

£10 Prizes: R J Chiswell, J M Leach, N C Langley, P Upson, J P Thornley, R R Hull, J Herring, E Calvert, J Hirst, J G Peters, A J Rye, R Bailey, R Wilkinson, G Taylor, J R Walker, A Phillips, N Tingle.

We would like to thank all those who entered, and encourage entries for our next competition. The closing date for this has been extended to 31st October, and there are special categories both for programs largely in machine code, and for business applications, though all types of program are eligible and welcome.

---

# STRING SEARCH PROGRAM

by John Marriage

---

This program was written on a 32k machine, but there is no reason to suppose it will not run in a 16k machine. Its purpose is to assist with editing of Basic programs, particularly large ones. I wrote it to help with the development of a large Adventure-type program which uses a lot of procedures, some of which call other procedures. I needed to find every occasion where a procedure was used, and in some cases the places where a particular variable was mentioned, and I kept losing them!

The program is short, and resides in 2 pages (0.5k) of memory below the large program which you are developing. Whilst you are working on the large program, you can do all normal editing, etc., though a little care is needed before running. During editing, press f8 and you will be asked for a search string. This can be any string up to 12 characters, but without either BASIC keywords (which are stored in memory as tokens, and not spelt out). This restriction doesn't seem to be a problem in practice.

The search program then looks "above itself" in memory, and examines each line of the main program for the search string. If the target is found in any line, the line number is printed. All lines containing the target string will be so listed, and control then returns to the level of the main program, which is again edited as normal. The search process takes about 2 seconds per 1k of program checked, regardless of how often the target is found.

## OPERATING INSTRUCTIONS

- 1) Press "Break" or turn on the computer.
- 2) Load the string search program.
- 3) Run it and immediately ESCAPE (this assigns key f8).
- 4) Type in: PAGE=&1000 <Return>.
- 5) Either load or type in main program as normal.
- 6) To use the search facility, press f8 as described above.
- 7) After each use, control returns automatically to the main program
- 8) If you press "Break" then repeat step (4).

## TO RUN THE MAIN PROGRAM

To prevent the search program overwriting the main program, LOMEM (the bottom of variable storage) is shifted up high. If your main program needs more than 500 bytes of variable storage it won't run. If necessary, before running, shift LOMEM by: LOMEM=TOP

No line numbers below 10 in the main program please! (Remember that RUN will run my program not yours)

Why f8? Well I always load the first few 'f' keys with a standard set of useful editing commands when I am doing program development. It was the first empty key.

```

2 LOMEM=HIMEM-500:DIM T%12,P%250
3 *KEY 8"PAGE=&E00|M RUN|M"
4 B%=&1001:INPUTLINE"ENTER SEARCH STR
ING"$T%
5 REPEAT:$P%="":N% =256*?(B%)+?(B%+1):
K%=B%+3:L%=K%+(K%-1)-5
6 FOR B%=K% TO L%:$P%=$P%+CHR$(?B%):N
EXT:B%=L%+2
7 IF LEN($T%)>LEN($P%) THEN 9
8 IF INSTR($P%,$T%)<>0 THEN PRINT N%;
9 UNTIL ?(B%)=&FF:PAGE=&1000:END

```

## 16k SCREENPLAY

The short program entitled "SCREENPLAY" contributed by Ian Bell of Hatfield published on p10 of the June Newsletter needs a 32k machine. Listed below is a modified version which will run on a 16k (OS 0.1) machine. I doubt if it will work in 32K though... can't win 'em all I suppose. Many thanks to Stuart MacGreggor of Glasgow for the modifications.

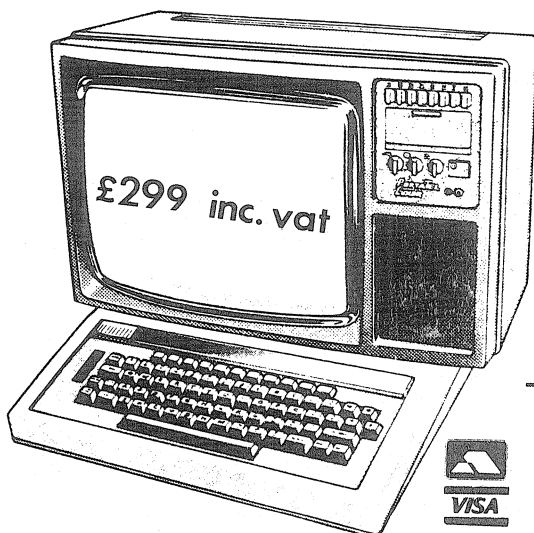
Screenplay shows the speed of the BBC micro's graphics, and the ease and speed of the assembler. It produces a screenful of fine resolution colour patterns; the effect is quite startling. Hold the space bar down to change the pattern, or just delete line 100 for an ever changing display.

```

20 MODE5:NC=4
30 DIM P%200
40 [OPT 0
50 .ST LDX#0:STX &70:LDX #HIMEM/256
   :STX &71:LDX #0
60 .HERE LDA &70:STA (&70,X):INC &70
   :BNE HERE:INC &71:LDY &71
   :CPY #&40:BNE HERE:RTS
70 ]
80 CALL ST
85 REPEAT
90 VDU19,RND(NC)-1,RND(8)-1,0,0,0
100 REPEAT UNTIL GET=32
110 UNTIL FALSE
120 REM PRESS THE SPACE BAR...
130 REM ..TO CHANGE THE PATTERN.
```

Thanks also to Geoff Smith of Worcester Park for an alternative model A version. 

### Timeshare your Colour Monitor with the Family



### COLOUR TV

*Plus*  
RGB  
*Plus*  
PAL VIDEO

BBC Micro lead included  
Excellent resolution, geometry

PortaTel LUXOR 14" Colour Monitor  
Model No. RGB3711 TV Receiver

**PortaTel** conversions limited  
television and electronic engineers  
25, sunbury cross centre, staines road west,  
sunbury-on-thames, middlesex. tw16 7bb  
telephone: sunbury-on-thames 88972



## POSTBAG

From: H.P.Craig, Oadby, Leicester.

There seems to be a defect in the INT function.

```
PRINT INT((123.45-123)*100)
```

gives 44 not 45 as it should.

### Reply

You have discovered for yourself that some decimal numbers cannot be converted into binary exactly. For example the decimal number 0.2 is 0.0011001100110011etc. Now however many bits you use for the binary equivalent the binary is ALWAYS WRONG.

If you try PRINT 123.45-123 you get 0.49999988 which shows that it is not the INT function that is wrong but the way numbers are represented. There is a whole field devoted to studying the accumulation and correction of such errors which can build up to mammoth proportions, try this:

```
10 A=1000000001
20 B=10000000000          (Make sure that you get the number
30 C=0.0000001          of zeroes correct in lines 10-30)
40 PRINT A/C-B/C
50 PRINT (A-B)/C
```

You should get two identical answers of 10000000 but you don't, try it and see, one of the answers is 20% in error!

---

### SOUND BUFFER

David Stonebanks has a program that draws Union Jacks and plays the National Anthem. He is looking for a method of detecting when the sound buffer can take another note. This may be done by using ADVAL with a negative argument in the range -5 (Sound channel 0) to -8 (Sound channel 3).

```
10 PROCspace           80 DEF PROCspace
20 FOR I = 1 TO 10     90 FOR J = -5 TO -8 STEP -1
30 SOUND 1,-1,10*I,10 100 PRINT ADVAL(J);
40 PROCspace           110 NEXT
50 NEXT               120 ENDPROC
60 END
```

The above program illustrates the use of this command. The procedure 'space' prints the space left in the four sound buffers, and this is initially 15. However this decreases by 3 as each note is added giving a maximum of 5 notes queued up. This is for the Amplitude or Envelope, Pitch and Duration. The first note does not enter the queue but is sent directly to the sound generator circuit.

---

From: G.A.Rooker, Hendon, London

Mr McMinn's method for stepping through listings (ie using CTRL-shift, Issue 4, p4) also works in at least one other way. Try using it during the game 'Bomber' (Issue 2); it will freeze the plane's flight. However, it does not appear to work where an object is moved around the screen by MOVE statement (as opposed to the PRINT TAB).

---

From Alan Pemberton, Sheffield.

I have just received the three BEEBUG software cassettes which I ordered recently, and find them excellent value for money - the all loaded perfectly at 1200 baud, and the documentation is sufficient for running the programs. It may have been useful, however, for those of us without printers to have included listings and

---

programming notes to make them easier to experiment with.

There are a couple of bugs which I found on the Games 3 cassette. Firstly, in "Shapematch" the string in line 2220 is too short to overtype "PENELOPE'S GO" whenever Penny or her eight-lettered friend has won a round. I changed:  
2220 PROCdbl(5,22,S\$, "Another turn!")

Secondly in "Rat-Splat" CHR\$230 is called without having been defined. This is OK unless you have been playing "Starfire", for example, in which case your car leaves a trail behind it, which the computer thinks is a rat, and crossing your own tracks causes 'splatting'. I added:

125 VDU 23,230,255,255,255,255,255,255

### Reply

Thanks very much for your useful comments, we shall be incorporating these corrections into later cassettes, but in the meantime we have included a note with Rat-Splat advising a cold start before running the program. As to supplying listings, we will consider this, but it would increase administrative overheads.

---

### DELIVERY COMPLAINTS

Distribution of the BBC machine is currently transferring to "Vector Marketing" (see editorial) and this will hopefully improve all aspects of the operation, but in case you are one of the many who have fallen foul of BL and are still waiting for your machine, here are a couple of letters (sampled from many that we have received) to help you to feel that you are not totally alone in your frustration. We have passed copies of these and other letters directly to the BBC, and have asked to know exactly what went wrong in these, and a number of other disastrous cases, and will report back.

Dear Sir,

Since I placed my order on 10th Jan 82, I have received nothing except pre-printed acknowledgements telling me nothing about delivery timescale. They have ignored two letters, and attempts to telephone them are futile as their telephone seems permanently engaged.

I think that to keep a customer waiting for six months without any information, apart from being bad manners, is a very irresponsible attitude to business. There have been a lot of reports in the press which tend to increase frustration in the absence of official information.

Yours faithfully  
D.R.Hignett

Dear Sir,

On Jan 26th I sent off my order for a Model-B micro, a cassette recorder and suitable cassette leads. On Feb 8th I received a letter of acknowledgement.

As I knew there were supply difficulties I waited 'till April before writing to ask what was happening. I received a letter telling me they could find no trace of my order. I answered the questions set out in the letter and subsequently received another copy of the "lost order" letter with a note written on it asking me to quote the acknowledgement letter. I sent them a photocopy of the letter.

I then received another letter thanking me for the photocopy but asking me the same three questions again! This time I told them everything I knew right down to the number of the cheque.

Having heard nothing for three weeks, I wrote another letter asking if they had traced my order and what will happen if they can't.

I have now received my fourth version of the "lost order" letter with the same three questions which I have now answered three times.

---

What do I do to get through to them? There must be many others like me, as the "lost order" letter and even its signature is duplicated.

Do you know what will happen to us unfortunates whose orders to BL Marketing have got lost? I would be grateful for some advice, being a member of the user group who is wondering if he'll ever be a user!

Yours  
P.J.Martin

## ROM CHARGE

We have received a considerable postbag of complaints about Acorn's intention of charging £10 for their new operating system ROM. A number of readers have even said that they are considering taking the BBC to the small claims court over the issue, since, they argue, the machine as originally supplied, fell below the implied specifications put out by the BBC. See the editorial for further details.

Dear Sir,

Like many other proud owners of the BBC micro, I was horrified to hear of all the bugs in the 0.1 operating system. As my machine contains the 0.1 ROM, I am one of the unfortunates that are going to have to pay the £10 'nominal charge'.

This cannot be right and I am sure that the law would be on our side if it was taken to court. Surely, under the "trades descriptions" or "Sale of goods" act there must be some way out.

I would like to see BEEBUG taking legal advice on this matter, and I for one, am prepared to go to the small claims court to fight this matter out.

Dr Harris

Dr Harris encloses a letter that he has written to Acorn, arguing for a free replacement ROM. Towards the end of his letter he makes the observation "I have just sent off my guarantee card and cannot recall having seen anything excluding the ROM from the guarantee." This is well put, and we would ask Dr Harris to pass on a copy of Acorn's reply for publication in BEEBUG. It should I think be of considerable interest.

There have been too many letters on this theme to include them all here, though Mr J. Spilsbury included a good idea in his letter about the ROM:

"I would suggest", he wrote, "that in the interest of good consumer relations, that the money lost to the buyers in interest over long waiting periods be used to finance the issue, free of charge, of a 1.0 ROM to every owner, together with full installation instructions, and with an option of free installation by a local dealer."



## **HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS**

### PRINTER INTERFACING

In BEEBUG Issue 1 p20 we gave details of printer interfacing, and mentioned an anomaly in the ACK and BUSY lines on current model A machines. It would seem that this has now been rectified, and the following data (taken from a current model B now applies.

- \* Pin 19 is connected to ACK, not as shown in the provisional manual.
- \* Both ACK and STROBE are negative logic i.e. zero volts represents the TRUE state.
- \* Pin 26 is left open circuit as it has a special function on some printers. (It resets a Seikosha GP-80A for example.)
- \* Viewed from the front of the computer, pin 1 is at the top right of the connector.

Program tested on  
0.1 and 1.1 O.S.

## MULTI COLOURED BEEB

Program tested on  
0.1 and 1.1 O.S.

This program by John Marsden prints all screen output in different colours, with sound as well!

The program illustrates an important facility of the BBC computer, vectoring of the output channel via the address &20E. Line 50 picks up the contents of the vector which is then used throughout the program to output characters.

Line 380 puts the start address of the assembler portion of the program into the vector, so that all character output from the computer is intercepted by this program.

Once this program has been RUN it can be erased, leaving the machine free for Basic. The machine code routine that performs the work is stored in the RS423 receive buffer at &0A00.

10 REM Multi-coloured Beeb Machine	140[OPTZ	270 PLA:TAX
20 REM by John Marsden	150 STA &74	280 PLA
30 REM	160 PHA	290 JSR oswrch
40 MODE 5	170 TXA:PHA	300 INC&80
50 oswrch=!&20E AND &FFFF	180 TYA:PHA	310 LDA&80
60 osword=&FFF1	190 LDA#17	320 CMP#4
70 ?&70=1: ?&71=0	200 JSR oswrch	330 BNE rts
80 ?&72=&F1: ?&73=&FF	210 LDA&80	340 LDA#1:STA&80
90 ?&75=0	220 JSR oswrch	350 .rts RTS
100 ?&76=1: ?&77=0	230 LDX#&70:LDY#0	360]
110 ?&80=1	240 LDA#7	370 NEXT Z
120 FOR Z=0 TO 3 STEP 3	250 JSR osword	380 ?&020E=0: ?&020F=&0A
130 P%=&0A00	260 PLA:TAY	

# TECHNOMATIC LTD.

Official *BBC* Stockists

Model A to Model B upgrade Kit £60.00

16K RAM 8 X 4816AP-3 100nS £21.60

FULL RANGE OF CONNECTORS & LEADS AVAILABLE EX-STOCK

### PRINTERS

SEIKOSHA GP100A £189

EPSON MX80F/T3 £330

NEC PC8023 £340

Carriage £8 per printer

### MONITORS

14" BMC Colour 18MHz Bandwidth £240

14" Microvitec Colour Monitor £269

Carriage £8 per Monitor

Sanyo Cassette Recorder £24.50 + £1 p&p

Please phone for our *BBC* leaflet for full details on software, books & hardware  
We also stock a large range of CPUs, Memories, TTLs, CMOS & Connectors  
Please add 40p P&P and 15% VAT to the order value

# TECHNOMATIC LTD.

17 Burnley Road, London NW10 1ED. Tel. 01- 452 1500/450 6597  
Retail Shops: 15 Burnley Road, NW10. 305 Edgware Road, W2

## LOGIC (PART I)

There are whole areas of programming which seem to remain unused by many people. The probable reason is that of 'fear of the unknown'. In this article we will deal with one such subject, that of LOGIC.

You probably will have noticed two special variables in BBC Basic - TRUE and FALSE. These have a variety of uses and it was a good bit of thinking on the part of the writers of BBC Basic to incorporate them. They do, in fact, have numeric values of -1 and 0 respectively, ie entering X=TRUE has the same effect as the statement X=-1. Representing TRUE as -1 may seem strange, but you should see why in a while. You would expect that something that was 'not true' would be 'false', and something that is 'not false' would be 'true'. So try this:

```
PRINT FALSE      gives 0
PRINT TRUE       gives -1
PRINT NOT FALSE  gives -1
PRINT NOT TRUE   gives 0
PRINT NOT 0      gives -1
PRINT NOT -1     gives 0
```

From the above examples you can see that the word NOT is also allowed. Other words (called "operators") that are allowed are AND, OR and EOR the latter standing for 'Exclusive OR' these will be explained later.

In order to really understand how logical statements operate you need to understand binary. This is not as frightening as it may sound.

### Binary

Binary is a number system comprising only the digits 0 and 1. Computers work in binary because they are made of logical elements and two-state devices. If present day computers worked in our ordinary (denary) number system they would be many times larger and slower.

Most registers within the 6502 microprocessor itself comprise 8 bits (bit means Binary digIT). A collection of 8 bits is called a byte for example 00001101 is a byte. This could be an instruction or a number, but in this article we are mainly concerned with numbers, in which case it represents 13. The numbers 0, 1, and 2 would be 00000000, 00000001, 00000010 in binary. The following table should help a little:

Denary	Binary	Denary	Binary
0	0	6	110
1	1	7	111
2	10	8	1000
3	11	9	1001
4	100	10	1010
5	101	11	1011

Negative numbers are represented by the leftmost digit being a 1 so -1 should be 10000001, but it isn't. It isn't, because if we add 1 to -1 we would expect to get zero so 00000001+10000001=10000010 which certainly isn't zero. Therefore -1 is stored as 11111111 because 11111111+00000001=00000000 because if you are only using 8-bits the leftmost digit gets lost in the carrying process.

A simple rule to find the negative representation of any number is to write down the positive representation and then, working from the right, copy down all digits up to and including the first 1, then reverse all the other digits.

Note that BBC Basic stores integers as 4 bytes so that:

```
TRUE is really 11111111 11111111 11111111 11111111
and FALSE is  00000000 00000000 00000000 00000000
```

### The NOT operator (p 306 of the User Guide)

The NOT operator is a 'unary' operator which reverses each binary digit, so that if it was a 0 previously it will become a 1, and vice versa. (A unary operator is one that only has one number on which to operate, 'minus' is another example because you can have -9 on its own, whereas you can't have a multiplication operator without two numbers). So NOT 00011101 will become 11100010. Hence a positive number will become negative and vice versa when NOT is applied to it. Practice a few examples because you can always try them on the computer to see if you are correct. Eg NOT 17 gives -18. (Test this with PRINT NOT 17)

### The OR operator (p 316 of the User Guide)

OR is a binary operator and requires two numbers on which to work. OR 4 doesn't make sense, 3 OR 7 does however.

PRINT 3 OR 7 gives 7                      PRINT 17 OR 5 gives 21.

In order to make sense of this we must again look at the binary equivalents:

3 is 0000011	17 is 00010001
7 is 00000111	5 is 00000101
7 is 00000111	21 is 00010101

Look at each bit in the above examples and note how the result is a 1 if either of the two bits above it is a 1. But a zero if both of the bits are zero.

Below is what is called a "Truth Table" it shows all the combinations for the OR operation. For example, if you are performing an 'OR' on two bits, and one bit is a 0 and the other bit is a 1, then the second row of the table shows you that the resulting bit will be a 1.

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

### The AND and EOR operators (pp 205, 250 of the User Guide)

These are also 'binary' operators and their truth tables are given below, together with the one for OR:

A	B	A OR B	A AND B	A EOR B
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	0

Various tasks can be performed using logical operations, one such task is called "masking" (see below).

### Operator Precedence

In the same way that multiplication is performed before addition in an arithmetic expression, the logical operators have rules of precedence too. The User Guide lists them on page 144. Without these rules note how the following statement would be ambiguous:

IF sex\$="M" OR sex\$="F" AND age>59 THEN.....

Also note that without brackets PRINT NOT "M"="F" gives a 'Type mismatch' error because the NOT has a higher precedence than =. The statement must be written PRINT NOT ("M"="F")

Uses for this are manifold - see for example p23 of issue 3. One use in particular that can be readily explained is to provide a mask so that all lower case and upper case letters are accepted as upper case only. For example, try the following program

which will always return upper case characters regardless of whether the 'caps lock' is on or not:

```
100 REPEAT
110 PRINT CHR$(GET AND 223);
120 UNTIL FALSE
```

This works because the value 223 (which is 11011111 in binary) is a suitable mask to mask out the third bit from the left.

```
A is CHR$ 65 is 01000001
a      97 is 01100001
```

This shows that ASCII codes (far from being random or haphazardly thought out), are quite cleverly arranged. Upper and lower case letters only differ in the third bit from the left. This bit can thus be masked out if not required.

The reverse process, to make all letters into lower case can be achieved by forcing the third bit to be a 1. To do this we OR the byte with 00100000.

```
100 REPEAT
110 PRINT CHR$(GET OR 32);
120 UNTIL FALSE
```

### Exclusive OR

Exclusive OR is nearly the same as OR, but the meaning is slightly different from the interpretation that we make in everyday language. For instance if I said "You will be imprisoned if you murder a man OR if you murder a woman" then you would still expect to be imprisoned if you murdered BOTH a man and a woman. The 'exclusive or' says that you would not. For example the phrase "You are a man OR you are a woman" is an example of an 'exclusive or' used in everyday language because you cannot be both (at least it's not very likely!).

Logically 'exclusive or' can be derived from AND and OR:

A EOR B is the same as (A OR B) AND NOT(A AND B)

One very useful feature of EOR is that it is reversible and you can easily return to the original value by performing another EOR. For example

```
10 code=22
20 REPEAT
30 char$=CHR$(GET EOR code)
40 PRINT char$;
50 UNTIL FALSE
```

To stop the program press 'escape'.

This is a simple encryption program, whatever you type is encoded using the 'secret' code initialised in line 10 (here the secret code is 22). As you type, the coded characters appear. The interesting thing about this program is that if you run the program again, but this time enter the coded characters then you will get back the original!

If you want to send secret messages to one another, then you both need the same program and code number. To make it more difficult to crack you could begin the message with the code, then this could be inserted into line 10 by the recipient, before the program is run. (Some weird things may happen if you use a code between 32 and 63 inclusive see if you can think why.)

Because you can perform another identical EOR to recover what was there before, this has very useful advantages in graphics. You can plot something on top of something else, and when you plot it again the original will re-appear. The following program uses 'exclusive or' plotting via the GCOL 3,? option:

```
5 MODE 5
10 GCOL 0,1
20 PROCfill box(100,200,800,1000)
30 GCOL 3,2
40 MOVE 0,0:DRAW 1279,1023
50 INPUT"Press 'return' to remove";Q$
60 MOVE 0,0:DRAW 1279,1023
999 END
1000 DEF PROCfill box(X1,Y1,X2,Y2)
1010 MOVE X2,Y1:MOVEX1,Y1
1020 PLOT 85,X2,Y2:PLOT 85,X1,Y2
1030 ENDPROC
```

Here a box is drawn in one colour, and then a line drawn through it. After you have pressed 'return' the same line is drawn over the previous one, and the original screen is reinstated totally. It may only be clear what is happening on a colour set I'm afraid.

In a similar manner a man can be made to walk in front of a house without erasing it. For an example of this see pages 162-166 in the book by Cryer which was reviewed in the May '82 issue of the BEEBUG newsletter.

[This article will be completed next month.]

S.W. 

## HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

### TRAPPING ERRORS

Many programs trap the escape key to allow exit from the program, or some other function. When debugging such a program, syntax errors are trapped and not reported to the screen. A simple change to the error routine overcomes this problem:

```
100 ON ERROR GOTO 1000
110 REM Rest of program
999 END
1000 REM Error routine
1010 IF ERR<>17 THEN
    REPORT:PRINT" @ LINE ";ERL : STOP
1020 REM Routine to handle ESCAPE follows
```

[Note:The error code for ESCAPE is 17.]



MICROWARE (LONDON ) LTD PRESENT THE "ZL"  
RANGE OF DISK DRIVE SUBSYSTEMS  
FOR THE BBC MICRO.

BARE DRIVES FROM ONLY £ 105.00

IN PLASTIC ENCLOSURES £ 135.00

DUAL UNITS WITH OWN P.S.U. £ 295.00

INCLUDES 12 MONTHS WARRANTY ON CASED SUBSYSTEMS  
EPSON PRINTERS

MX 80T/3.....£ 275.00

MX 80FT/3.....£ 325.00

MX 100/3.....£ 429.00

PRINTER CABLE.....£ 15.00

Full range of printers carried in stock. Come and see us for a free demonstration.

PRICES DO NOT INCLUDE POSTAGE AND PACKING OR VAT.

MICROWARE (LONDON ) LTD. 637 HOLLOWAY RD. LONDON N19.

PHONE 01 272 6398 FOR FURTHER DETAILS.

Microware

Microware



---

## POINTS ARISING

---

### Key Define (July p5)

For an update on last month's user key article, see "User Keys Revisited" in this issue.

### User Port (July p17)

We gave the number of the driver/buffer I.C. in the user port article variously as 74LS16 and 74LS17. Both will in fact work, and have the same pinout. The difference is that the 74LS16 is an INVERTING driver, and so will turn relays ON (rather than off) when it is connected to a high output from the port, this keeps the logic simpler.

### Chunky Invaders (July p31)

Several people wrote to report that they could not get the program to run. After some tracing we discovered that they had entered a one(1) in place of a lower case L (ie l) in lines 2190 and 2200. This caused the error message "Bad Argument at line 590" to appear. In fact, all our major programs (anything above a couple of lines or so) have always been listed directly from the BBC machine itself, though each month there are always several people who swear that our listings contain typographical errors. As you will notice in this issue, we have changed our daisy-wheel to give crossed zeros (0) so as to further clarify listings.

In practice, typing a program from a printed listing is not as easy as it may seem, and there are a number of traps for the unwary. In this issue we look at one of these traps in some detail - see "Unexplained errors explained". In later issues we intend to go into the whole question of debugging programs, which will help in tracking down any wrongly typed lines.

### VDU Syntax

BEEBUG issue 3 page 26 contained a slight error in the syntax of the VDU command. A semicolon signifies that the previous number (not the next number) should be sent as two bytes, least significant first. The three zeros at the end of the VDU19 command may be sent as: VDU19,0,1;0; The first semicolon sends '1' as two bytes i.e '1' then '0' and the second sends '0' as two zeros making three in all and thus equivalent to: VDU19,0,1,0,0,0

### String Handling Tip

David Nichols of Bishops Stortford has written in claiming responsibility for the string handling tip in the July issue. If you remember we lost his name. Many thanks to David for such a useful tip.

---

### USEFUL ARTICLES UPDATE

#### Personal Computer World

Sept 1982 Beeb Colour Hi-Res by Jeff Aughton (pp 180-181). Looks at memory mapping of graphics in mode 2.

---

---

## USER KEY UPDATE

---

The article on user defined keys carried in our July issue created a good deal of interest, and what follows is a summary of the suggestions received.

### NEW ERROR

Firstly, there was an error in the final line of the program. Richard Russell, Jeremy Taylor and others spotted this. The problem is that you cannot put NEW in a program line. Different solutions to the problem were suggested. One solution offered was to use the following two program lines:

```
240 ON ERROR NEW
```

```
250 ERROR
```

Another was to incorporate NEW into one of the user defined keys.

### CTRL V

Secondly, to save space in the key buffer, we used control-V (|V) as a shorthand for changing mode. This worked well in all our tests, but there is a bug in this command that the new user guide refers to quite casually - changing mode by this method does not reset the memory allocations for the screen display - technically speaking it does not reset HIMEM. Our advice is to play safe and replace |V7 with MO.7|M (MO. is the minimum abbreviation for MODE). For more on this, see the 'Control key' article in this issue.

### BUG FIX COMBO

Brian Carroll has reported that he has included the bugfixer routine (the program given in the July issue to fix the OS 0.1 cassette bugs) into a modified version of the key set routine, so that each time the machine is turned on, the keys can be set and the bugs fixed using a single program. If he would be kind enough to send us a cassette of his program, we will publish it next month.

### DISCRETE SAVING AND LOADING

The strings for the user defined keys are stored in a buffer at &0B00 (hex), and can therefore, as Jeremy Taylor points out, be directly saved and loaded using \*SAVE and \*LOAD. To do this, first define the keys that you require, then to save this on tape, type:

```
*SAVE"keys" 0B00+00FF
```

The tape can then be reloaded using: \*LOAD""

The advantage of all this is that you can perform the load operation without disturbing the Basic program already resident in the machine. Jeremy makes use of this routine to load in alternative sets of key settings depending on what work he is doing. Here are his two sets:

Key	Normal usage	Graphics
0	MODE 0	MODE
1	OLD	MOVE
2	LIST	DRAW
3	NEW	PLOT
4	AUTO	CLS
5	SAVE	CLG
6	LOAD	COLOUR
7	MODE 7	GCOL
8	RUN	POINT
9	*KEY	*KEY

He writes that key 9 is used to speed redefinition of itself for frequently used variable names.

One further suggestion that we received was to place a 'Return' character (|M) before each command, so as to prevent errors caused by material already written in the current screen line (ie data in the input buffer).

---

# HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

## REMs in DATA statements

Mrs L.H.L.Sellick says:

It is not possible to end a DATA line with a REM statement if the REM falls at a point where the program is trying to read a string. Thus:

```
100 DATA A,B,C,D :REM Cabbages
```

will produce "E :REM Cabbages" as the fifth datum string. If there is a comma in the REM statement then this will be treated as the end of a datum. Thus:

```
100 DATA A,B,C,D,E :REM Cabbage,Kings
```

will produce "Kings" as the sixth datum string.

## Out of sequence AUTO

David Moncur of London states that using AUTO you can insert a line out of sequence. For example, suppose you had two lines 60 and 80, and you were to then type AUTO 70 and arrange for:

```
70 5 STOP
```

to appear, you would then have a line out of sequence.

## Recall of function keys

David also contributes:

It is sometimes useful to recall the information associated with a function key and make it appear on the screen. This can be difficult if it is a long string that includes CLS. However it can be done neatly using AUTO: Choose a range of line numbers outside your current program, for example AUTO 9000. Pressing each function key followed by return causes the list to be displayed as program lines. Remember to delete them afterwards.



## MIDWICH HAS MOVED! OUR PRICES HAVE TOO — DOWN!

### BBC MICRO UPGRADE KITS

*BBC 1	8 x 4816 AP11 (IC61-68)	21.50
BBC 2	Printer/user I/O kit	7.60
BBC 4	Analogue input kit	6.50
BBC 5	Serial I/O & RGB kit	10.25
BBC 6	Expansion bus & tube kit	5.95

*\*Full spec devices as recommended by Acorn.  
Beware of cheaper and inferior devices which do not work correctly.*

### BBC CONNECTOR KITS

BBC 21	Printer cable & plug	13.00
BBC 22	User port connector and 36" cable	2.00
BBC 44	Analogue input plug with cover	2.25
BBC 55	DIN plugs for Serial I/O and RGB input	0.99
BBC 66	Bus port connector and 36" cable	3.50



VISA

24 Hour Telephone order service for credit card holders.

All prices exclude VAT and carriage (0.75 on orders under £10 nett).  
Official orders from educational and government establishments, and public  
companies accepted. Credit accounts available to others (subject to status).

All orders despatched on day of receipt. Out of stock items will follow on automatically at our discretion  
or a refund will be given if requested.

NO SURCHARGE FOR CREDIT CARD ORDERS



## MIDWICH COMPUTER CO LTD

Rickingham House, Rickingham, Suffolk IP22 1 HH Telephone (0379) DISS 898751

*Please make a note of our new address & telephone number*

---

## PROCEDURE / FUNCTION LIBRARY

---

The BBC machine offers the useful facility of procedures and user defined functions in Basic. We aim to exploit these features to the full by building up a library of useful procedures and functions that may be stored separately on tape or disc, and then incorporated into programs under development. We published an appropriate method of merging programs in the June issue p16 though if you use it, remember to renumber your procedure and host program so that there is no clash of line numbers. We intend to add to our library with each issue - and contributions are very welcome.

### Procedure/Function Library Index

Ellipse	Apr 82	p18	Draws an ellipse (or circle).
Box	May 82	p8	Produces a filled rectangle.
Polygon	Jun 82	p4	Produces a centred n-sided polygon.
Centre	Jun 82	p7	Prints text centred on any given line.
HCF	Jul 82	p12	Evaluates highest common factor.
Strip	Jul 82	p12	Strips trailing edges from a string.
Input	Jul 82	p12	Full handling of keyboard input, with prompts, locking out unused keys validating etc.

This month we publish an index of functions & procedures from previous issues, together with two new functions: "Days Between" which calculates the number of days between any two dates (providing that they fall between 1/3/1900 and 28/2/2400). The second function "Yes/No" checks the keyboard for a Y/N answer in either upper or lower case.

#### FUNCTION: Days Between

This function (listed on lines 1000 to 1040) returns the number of days between two dates. This is very useful for many purposes, and has been kept as short as possible.

The function does not perform any error checking, though this can easily be built into either the function or the main program.

The function only gives the correct answer if the dates are between 1/3/1900 and 28/2/2400. (Because 1900 and 2400 are NOT leap years). However, I don't see that as being a real limitation.

```

5 REM "Days Between" Function demonstration
10 CLS
20 INPUT "From (D,M,Y) ",d1,m1,y1
30 INPUT " To (D,M,Y) ",d2,m2,y2
40 PRINT "Number of days between:"
50 PRINT ;d1;"/";m1;"/";y1;" and ";d2;"/";m2;"/";y2;"
   is:"FNdays_between(d1,m1,y1,d2,m2,y2)
60 GOTO 20
1000 DEF FNdays_between(d1,m1,y1,d2,m2,y2)
1010 LOCAL t1,t2
1020 t2=INT(365.25*(y2+(m2<3)))+INT(30.6*(m2+1-(m2<3)*12))+d2
1030 t1=INT(365.25*(y1+(m1<3)))+INT(30.6*(m1+1-(m1<3)*12))+d1
1040 =ABS(t2-t1)

```

#### Yes/No Function

In any large program many lines of code may be used asking the user for a Yes/No reply. The Function listed below (lines 200 to 270) simplifies this and removes superfluous code from the main program, retaining only the Function name and prompt string. Two examples are shown of use, one to select from two choices (line 40), and the other to terminate a loop (line 80). Run the program to see how this works.


---

The Function returns TRUE for 'Y' or 'y' and FALSE for 'N' or 'n'. Line 240 of the procedure requires some explanation. The GET gets a single character from the keyboard whilst 'AND &DF' converts lower case characters to Upper. Line 250 then only needs to test for upper case characters even if a lower case answer was supplied.

If a 'RETURN' is required after the Y/N then insert the line:

```
263 REPEAT UNTIL GET=13
```

```
10 REM Yes/No Function Demonstration      150
20 REM J.Yale 12 Aug 1982                 160 DEF PROCblack_white
30                                         170 PRINT"Setting up for black & white"
40 IF FNyes_no("Do you have colour")     180 ENDPROC
    THEN PROCcolour ELSE PROCblack white  190
50                                         200 DEF FNyes_no(A$)
60 REPEAT                                  210 LOCAL reply$
70   REM Any program                     220 PRINT A$;" (Y/N) ? ";
80 UNTIL NOT FNyes_no("Another game")    230 REPEAT
90 PRINT"Bye"                              240   reply$ = CHR$(GET AND &DF)
100 END                                    250   UNTIL reply$="Y" OR reply$="N"
110                                        260   PRINT reply$;
120 DEF PROCcolour                         265   PRINT
130 PRINT"Setting up for colour"          270   =(reply$="Y")
140 ENDPROC
```



MAIL ORDER ONLY

**ELECTRONICS APPLIED,**  
**4, DROMORE ROAD,**  
**CARRICKFERGUS, Co. ANTRIM BT38 7PJ**  
 Please add 55p/order P & P

**COMPATIBLE**      **software**

MAIL ORDER LIST SEND LARGES.A.E.

BEEBUG MEMBERS **5%**  
discount

*Envelope and Character*  
*Definer. (32/k)*

Together these utility programs offer a complete character and sound envelope defining package for the BBC Micro. Even if you fully understand both the envelope and character defining commands, these programs will make their definition quicker and more accurate.

<i>side 1</i>	<i>side 2</i>
<p style="text-align: center;"><b>ENVELOPE DEFINER</b></p> <ul style="list-style-type: none"> <li>* Excellent use made of graphics windows and colour to display pitch and volume.</li> <li>* Database containing 20 predefined envelopes of everything from phasers to explosions.</li> <li>* Example graphs and step by step prompts allow easy defining of your own sound envelopes.</li> </ul>	<p style="text-align: center;"><b>CHARACTER DEFINER</b></p> <ul style="list-style-type: none"> <li>* 'A very nice VDU23 character definition program' — <i>Computer Users Club (GB)</i>.</li> <li>* Shows both magnified and true size characters as they are defined.</li> <li>* All other definable characters can be called up for display.</li> <li>* Easy to use and check performed on all input.</li> </ul>
<p>Recorded on quality cassettes, sent by first class post</p>	

Introductory Price - £4.95

Dealer Enquiries Welcome

## GOVERNMENTS MICRO ELECTRONICS PROGRAMME

We have a lot of members who are in the teaching profession in one way or another. They may be unaware of the existence of the M.E.P. and members interested in using the BBC machine for educational purposes could do well to contact the M.E.P.

All queries regarding the use of the BBC Micro in connection with education should go to your nearest M.E.P. regional centre. Unfortunately there is not space here to list all the 14 regional centres, but an A5 SAE to M.E.P. Headquarters, Cheviot House, Coach Lane Campus, Newcastle upon Tyne, NE7 7XA will get you a pamphlet. Please state that you are following the lead given in the BEEBUG MAG.

Here is a brief description of the M.E.P.'s activities to date supplied by the M.E.P. themselves:

The MEP began laying its foundations in March, 1980. Since then a Regional Information Centre has been set up in fourteen regions throughout England, Northern Ireland and Wales. This has enabled all 109 LEAs to link up cooperatively to ensure all schools benefit from the Programme.

The MEP works closely with the DoI, and as the 'Micros in Schools' scheme has been taken up, so the DES has encouraged the development of educationally sound materials to be produced for use with computers in the classroom. These materials, plus expertise on the various machines used in schools, are available from the centres. The Centres act as reference points for courses, curriculum development proposals and as links with industry.

The MEP has recently established four Special Education Centres who visit special schools with a mobile display of devices and peripherals which incorporate a micro in one form or another.

The BBC machine has played a key role in the development of computer literacy in schools. Many computer familiarity courses now feature the BBC machine and its ease of use and impressive capability has led to a positive change of attitude amongst teachers as more gain 'hands on' experience. The MEP is funding more exciting software for the BBC machine. It will be some time before we have the kind of software which matches the power of the hardware, but we are working toward this at a rapid rate. As new programs become available, the Regional Information Centres will help in the evaluation process, receiving feedback from teachers in the area and using their Newsletters for comments, advice, tit-bits of useful information etc.

### HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

#### Black on White

George Foot of Rotherfield Sussex tells us that he uses 80 column text display on an ordinary TV set and it is virtually illegible. However by reversing it so that it is black on white it makes it easier to read.

[Ed: This can be done using MODE 0:VDU 19,0,7,0,0,0:VDU 19,1,0,0,0,0]

#### CORRUPT TAPES

We have received many requests for a method of recovering a program containing a corrupt block on tape. S.P. Brooke writes that he has an 8k program with the first block damaged. Does anyone know a method to recover this?

Program tested on  
0.1 and 1.1 0.S.

## HIGHER/LOWER

by R Bailey

Program tested on  
0.1 and 1.1 0.S.

This is a prize winning program from our software competition. It makes quite good use of the Beeb's graphics, and yet still fits in a 16k machine.

The program is based on the popular TV game of "Play your cards right", where you have to predict whether the next card in a sequence will be higher or lower than the previous one.

To fit the program into a 16k machine you will need to save every possible space. So don't insert any extra spaces when typing from the printed listing.

### INSTRUCTIONS

You begin with a credit of £250. Your target is £2500. You can bet between £50 and whatever you have to your credit at each turn.

At the first, fourth and seventh card, you are offered a choice of changing the card. If you reach the target at the ninth card, you will begin a new round with your credit added to your total score.

If your credit runs out or you don't reach the target you are offered a new game.

### Program Structure

Roy has made it easier for you to tailor his program to your own needs by providing the following program structure. The only thing missing is a list of variables. He could not use meaningful variable names, because he wanted to keep the program to 16k.

<u>LINES</u>	<u>ACTION</u>
10	Randomise the random number generator.
20	Set up the array. Reset screen print field to 8 characters.
30	Best score, set this to your best score, and save the program.
40	Defines text area 16x20. Changes background colour to blue, colour 2 to black.
50-60	Sets the variables.
70-80	Prints the text. Shuffles and deals the cards.
90	Prints card value and checks for end of game.
100	Checks for enough credit.
110	Checks for change-card procedure.
140-260	End of game comments and another game option.
270-280	Data holding position of each card.
290-300	Function for use in PROCC
310-400	PROCC. Draws card in white and if F=0 then draws design to simulate back of card.
410-510	PROCS. Shuffles cards and sets array S to suits. This could be used on a 32k machine to prints suits on cards.
520-610	PROCCH. Changes card if requested.
620-700	PROCV. Produces the values that will be used when printing the cards.
710-740	PROCSH. Prints value in red or black in middle of card.
750-790	PROCT. Prints text.
800-839	PROCCQ. Accepts your bet and subtracts it from your credit.
840-880	PROCHL. Determines whether your prediction is correct and alters credit if required.
890	PROCD. Deletes messages.
910	PROCU. Updates credit on the screen.
940	PROCBS. Prints best score and total score.

```

10 X=RND(-TIME)
20 MODE5:DIMA(12),B$(9),S(12):@%=&000
00908
30 BS=0
40 VDU28,4,16,19,0:VDU19,128,4,0,0,0,
19,2,0,0,0,0
50 RT=0:CLS
60 C=250:MB=50:AIM=2500
70 F=0:PROCT:RESTORE:FORJ=1TO9:READX,
Y:PROCC:NEXT
80 PROCS:PROCV:RESTORE:F=1
90 FORJ=1TO9:READX,Y:PROCC:PROCSH:IFJ
=9THEN130
100 IFC<MB THEN140
110 IFJ=1ORJ=4ORJ=7THENPROCCH
120 PROCQ:PROCU:PROCHL
130 NEXT
140 CLS:RT=RT+C:IFC>=AIM THEN260
150 IFRIT>BS THENBS=RT
160 IFC<MB THEN240
170 PRINTTAB(0,2)"You have failed"
180 PRINTTAB(1,4)"to reach the"
190 PRINTTAB(4,6)"TARGET"
200 PROCBS
210 PRINTTAB(0,8)"ANOTHER GAME?"
220 PRINTTAB(4,10)"Y/N":A$=GET$
230 IFA$="Y"THEN50ELSE CLG:PRINTTAB(0,
10)"SEE YOU SOON":TIME=0:REPEAT:UNTILTIM
E=500:MODE7:END
240 PRINTTAB(0,2)"You have failed"
250 PRINTTAB(2,4)"miserably":GOTO200
260 PRINTTAB(0,9)"CONGRATULATIONS":GOT
060
270 DATA50,10,350,10,650,10,950,10
280 DATA50,260,350,260,650,260,950,260
,50,560
290 DEFFNX=150-Z*30
300 DEFFNY=200-Z*40
310 DEFPROCC
320 GCOL0,3
330 MOVEX,Y:DRAWX,Y+200:DRAWX+150,Y+20
0:PLOT85,X,Y
340 DRAWX+150,Y:DRAWX,Y:PLOT85,X+150,Y
+200:IFF=1THENENDPROC
350 GCOL0,1:FORZ=0TO4
360 MOVEX,Y+Z*40:PLOT1,FNX,FNY
370 MOVEX+Z*30,Y:PLOT1,FNX,FNY
380 MOVEX,Y+FNY:PLOT1,FNX,-FNY
390 MOVEX+Z*30,Y+200:PLOT1,FNX,-FNY
400 NEXT:ENDPROC
410 DEFPROCS
420 FORJ=1TO12
430 A(J)=RND(52):F=0
440 IFJ=1THEN470
450 FORK=1TOJ-1:IFA(J)=A(K)THENF=1
460 NEXT
470 IFF=1THEN430
480 NEXT
490 FORJ=1TO12:FORL=1TO4
500 IFA(J)>13THENA(J)=A(J)-13:S(J)=L
510 NEXT:NEXT:ENDPROC
520 DEFPROCCH
530 PROC:PRINTTAB(2,8)"Change card?"
540 PRINTTAB(6,10)"Y/N":REPEAT:A$=GET$
550 UNTILA$="Y"ORAS$="N"
560 IFA$="N"THEN610
570 IFJ=1THENA(J)=A(10):S(J)=S(10)
580 IFJ=4THENA(J)=A(11):S(J)=S(11)
590 IFJ=7THENA(J)=A(12):S(J)=S(12)
600 PROCV:PROCC:PROCSH
610 PROC:ENDPROC
620 DEFPROCV
630 FORL=1TO9
640 B$(L)=CHR$(A(L)+49)
650 IFA(L)=9THENB$(L)="T"
660 IFA(L)=10THENB$(L)="J"
670 IFA(L)=11THENB$(L)="Q"
680 IFA(L)=12THENB$(L)="K"
690 IFA(L)=13THENB$(L)="A"
700 NEXT:ENDPROC
710 DEFPROCSH
720 VDU5:IFS(J)=1ORS(J)=3THENGCOL0,1EL
SEGCOL0,2
730 MOVEX+50,Y+100
740 PRINTB$(J):VDU4:ENDPROC
750 DEFPROCT
760 PRINTTAB(0,1)"Target=",AIM
770 PRINTTAB(0,3)"Min.Bet=",MB
780 PRINTTAB(0,5)"Credit=",C
790 PROCBS:ENDPROC
800 DEFPROCQ
810 REPEAT:PRINTTAB(0,8)"What is your bet"
820 INPUTTAB(2,10)"Bet="B:PROC:UNTIL
B>=MB ANDB<=C
830 C=C-B:PROC:ENDPROC
840 DEFPROCHL
850 PRINTTAB(0,8)"Higher or lower"
860 REPEAT:PRINTTAB(5,10)"H/L...":A$=G
ET$:UNTILA$="H"ORAS$="L"
870 IF(A$="L"ANDA(J+1)<A(J))OR(A$="H"AN
DA(J+1)>A(J))THENC=C+B*2
880 PROC:PROCU:ENDPROC
890 DEFPROCDD
900 PRINTTAB(0,7)SPC(80):ENDPROC
910 DEFPROCDC
920 PRINTTAB(8,5)SPC(8):PRINTTAB(8,5),C
930 ENDPROC
940 DEFPROCCBS
950 PRINTTAB(1,13)"Best Total"
960 PRINTTAB(1,14)"Score Score"
970 PRINTTAB(0,15),BS,RT;
980 ENDPROC

```

☆☆☆☆☆☆☆☆☆☆

ENTREPRENEURIAL PROGRAMMER to convert a unique application which has been developed on a Rockwell AIM 65 in 6502 mini assembler to BBC with discs and printer for sharing sales proceeds. Well documented. Apply to:  
364 Chase Green Ave, Enfield, Middx.



Program tested on  
0.1 and 1.1 O.S.

## HANGMAN (16k)

Program tested on  
0.1 and 1.1 O.S.

One of the first programs people often write for themselves is Hangman. This version is one of our competition winners. It is well thought out and uses B & W teletext graphics. If I can be critical for a few moments, the program fell down on the fact that it was not structured, and did not use meaningful variable names, also it did not restore the auto repeat on the keys after you said that you didn't want another game. The latter problem has been cured. (N.B. I disagree with the proper nouns in line 830, but have left them in).

If you want to change the words, or add your own then simply change the number of words in the DATA statement in line 730, and place your data anywhere, as long as it is after line 730, and is not using an existing line number. I estimate, that at an average of 6 letters per word, you can enter another 900 words even in a 16k machine. (Such is the efficiency in terms of memory requirement of teletext graphics). Example: If you want to add 2 words 'egg' and 'jam' then enter the lines:

```
730 DATA 234
1341 DATA EGG,JAM
```

Once again, this program, in common with all BEEBUG's programs, is listed DIRECTLY from the BBC Micro. See the article "Unexplained Errors Explained" in this issue if you have difficulty in getting it to work.

```
10 REM Hangman
20 REM by
30 REM John Peters & Tina Iles
40 REM
50 REM FX11,0 is Auto Repeat off
60 MODE7:*FX11,0
70 ON ERROR GOTO 1520
80 J=RND(-TIME)
90 PROCTITLE
100 PROCINST
110 FORP=1TO9:VDU7:PROCDRAW:A$=INKE
Y$(100):NEXT:P=0:REM VDU7=BEEP
120 PROCRAVE:VDU7:PRINTTAB(5,21)"C
an you save the innocent man."
130 REM INKEY(X) Used as delay
140 A$=INKEY$(400):CLS:PROCTITLE
150 PROCWORD:PRINTTAB(0,21)CHR$(141;
TAB(0,22)CHR$(141
160 FORJ=1TOLEN(A$):PRINTTAB(4+J*2,
21)" ";TAB(4+J*2,22)" ";:NEXT
170 PRINTTAB(2,19);"ABCDEFGHIJKLMNO
PQRSTUVWXYZ"
180 IFF=9THENPROCEND:CLS:PROCTITLE:
GOTO150
190 PRINTTAB(24,24)"ENTER LETTER ";
:L$=GET$:PRINTCHR$(7);TAB(37,24);L$;
200 REM CHR$(7) IN LINE 150 = BEEP
210 IFL$<"A"ORL$>"Z"THEN190
220 REM INSTR find L$ in A$ or retu
rns 0
230 L=INSTR(A$,L$):PRINTTAB(ASC(L$)
-63,19);" ";:REMOVE LETTER FROM ALPHA
BET
240 IFL=0THENP=P+1:PROCRAW:GOTO180
250 L1=L1+1:PRINTTAB(L*2+4,21)L$;TA
B(L*2+4,22)L$
260 A$=LEFT$(A$,L-1)+"#"+RIGHT$(A$,
(LEN(A$)-L)):REMOVE FOUND LETTER PUT IN #
270 L=INSTR(A$,L$)
280 IFL=0THENGOTO290 ELSE 250
290 IFL1=LEN(A$)THENPROCFIN:CLS:PRO
CTITLE:GOTO150
300 GOTO190
310 GOTO310
320 DEFPROCTITLE
330 LOCAL J%
340 FORJ%=2TO3:PRINTTAB(14,J%)CHR$(
8D;"HANGMAN":NEXT:REM CHR$(8D) DOUBLE
HEIGHT
350 ENDPROC
360 DEFPROCRAW:REM HANG MAN
370 ON P GOTO 380,400,420,440,460,4
80,490,500,510
380 PRINTTAB(0,18)CHR$(97);STRING$(3
9,"f"):REM CHR$(97) PRODUCES WHITE GRA
PHICS
390 ENDPROC
400 FORJ=17TO6STEP-1:PRINTTAB(9,J)C
HR$(97;"j");:NEXT
410 ENDPROC
420 PRINTTAB(9,5)CHR$(97;"_";STRING
$(14,"p")
430 ENDPROC
440 FORJ=1TO5:PRINTTAB(10+J,11-J);"
">:NEXT
450 ENDPROC
460 PRINTTAB(25,5)"0":FORJ=6TO7:PRI
NTTAB(25,J);"5";:NEXT
470 ENDPROC
480 PRINTTAB(23,8)"hcci";TAB(23,9)"
j(,j"TAB(24,10)"£7!":ENDPROC
490 FORJ=11TO13:PRINTTAB(24,J)"j";C
HR$(FF):NEXT:ENDPROC
500 PRINTTAB(23,11)"jk"+CHR$(FF)+"k"
;TAB(23,12)"*j"+CHR$(FF)+"*":ENDPROC
510 PRINTTAB(24,14)"jj";TAB(24,15)"
```

```

zj0":ENDPROC
520 DEFPROCINST
530 PRINTTAB(0,5)"In this game you
must try to save the""innocent man f
rom hanging by guessing"
540 PRINT"the hidden word."
550 PRINT"You must try to guess a l
etter that is in the word if you gue
ss wrong a piece"
560 PRINT"of the picture will be ad
ded."
570 PRINT"When the the picture is f
inished the manwill be hanged and you
lose."
580 PRINT"Each time you guess a let
ter it will be removed from the alpha
bet"
590 PRINT"Good luck, Justice rules
OK."
600 PRINTTAB(3,24)"PRESS ANY KEY TO
CONTINUE":A$=GET$:CLS
610 ENDPROC
620 DEFPROCGRAVE
630 CLS:PROCTITLE
640 FORJ=5TO19:PRINTTAB(15,J)CHR$&9
7;STRING$(5,CHR$255):NEXT
650 FORJ=9TO11:PRINTTAB(9,J)CHR$&97
STRING$(17,CHR$255):NEXT
660 PRINTTAB(15,10)" R I P "
670 ENDPROC
680 DEFPROCWORD
690 RESTORE
700 READ nwords
710 W=RND(nwords)
720 FORJ=1TOW:READ A$:NEXT:A1$=A$
730 DATA 232
740 DATA BOAT,APPLE,ELEPHANT,SOFA
750 DATA HOUSE,FLAT,GARDEN,BLUE,FISH
760 DATA WATER,MILK,MOTHER,FATHER
770 DATA TELEVISION,TELEPHONE,LION
780 DATA BOOK,GILL,CHAIR,CREAM
790 DATA CAR,DOG,MOUSE,CASSETTE
800 DATA PAPER,WOOD,BATHROOM,KITCHEN
810 DATA CHILD,ADULT,SPEAKER,HEAD
820 DATA TOMORROW,YESTERDAY,WEEK
830 DATA POLICEMAN,VENUS,SUN,MOON
840 DATA WINDOW,DOOR,GLASS,GLARE
850 DATA WOMAN,WIFE,HANDLE,PENCIL
860 DATA RADIO,SHOE,SHEEP,LAMB,LAMP
870 DATA SPACE,CLOCK,
880 DATA METAL,COLOUR,RULER,QUEEN
890 DATA RING,DIAMOND,RUBY,BAG
900 DATA KEYBOARD,MATCH,BELL,TEAPOT
910 DATA FRAME,DEER,BISCUIT,DONKEY
920 DATA DAIRY,TIME,GOLDEN,BREAD
930 DATA GOVERNMENT,MONEY,ANT,GREAT
940 DATA BOTTLE,CREAM,HOLIDAY,CURTAIN
950 DATA VIDEO,EGGS,MAN,LIGHT,NIGHT
960 DATA APPROVE,RESIST,RESCUE,SAFETY
970 DATA SYMBOL,SWALLOW,WATCH,WAX,WAY
980 DATA WEAPON,WRIGGLE,XYLOPHONE
990 DATA YAWN,YELLOW,VET,VICE,VALVE
1000 DATA UNIVERSAL,TYCOON,TUNNEL,NUT
1010 DATA TRACK,TONGUE,ROOT,ROMANTIC
1020 DATA TOLD,MORE,MYSELF,LAST,HERE,
1030 DATA AGAIN,APART,SONG,KNEW,DRESS
1040 DATA RETURN,CHEESE,HELL,STORY
1050 DATA BATTLE,BELIEVE,BELOVED,BERRY
1060 DATA BLAZE,BLUSH,BUCKET,BUSINESS
1070 DATA CALCULATE,CAMERA,CRYSTAL
1080 DATA CASSEROLE,CASUAL,DOWN,DOOM
1090 DATA DYNAMO,ELLIPSE,ENGINEER,END
1100 DATA EQUAL,EQUATOR,ESTABLISHMENT
1110 DATA EXPRESS,EXTENT,FASHION,FARM
1120 DATA FIRING,FLEECE,FLY,FLUX,FOG
1130 DATA FOLLOW,FRICTION,FRUIT,GAP
1140 DATA GALAXY,GALLOWS,GAZE,GATHER
1150 DATA GUESS,GUIDE,GULP,HABIT,HANG
1160 DATA HARP,HARD,SOFT,HATCH,ZOO
1170 DATA LIFE,COMPUTER,EFFECT,MISSION
1180 DATA PERMISSION,POSITION,FILE
1190 DATA OPTION,CHANNEL,ADDRESS,MODE
1200 DATA DISPLAY,CONTROL,OPERATING
1210 DATA GRATEFUL,FUNCTION,CHARACTER
1220 DATA DIFFERENT,SUPPLIED,CONSIDER
1230 DATA COORDINATES,INCLUDED,COINCIDE
1240 DATA INDEPENDENT,MNEMONIC
1250 DATA PROVISIONAL,BEGINNER,ADVANCE
1260 DATA HAPPY,CLEAR,ALTHOUGH,KEYBOARD
1270 DATA SPOKESMAN,OAK,UNREASONABLE
1280 DATA CAREFUL,CORRECT,EQUIVALENT
1290 DATA UNDERNEATH,ESPECIALLY,USEFUL
1300 DATA RESPECTIVE,CONTRIBUTION,DISC
1310 DATA IMPORTANT,RECOMMEND,TERMINATE
1320 DATA IMPLEMENT,PARAMETERS,CHANGE
1330 DATA PERIODIC,FREQUENCY,POSSIBLE
1340 DATA FACETIOUS,UNCOMPLIMENTARY
1350 ENDPROC
1360 DEFPROCEND
1370 PROCTITLE:PROCGRAVE
1380 PRINTTAB(3,22)"You've let the p
oor man die, FOOL!"
1390 PRINTTAB(4,23)"Press any key to
reveal the word."
1400 L$=GET$:PRINT"The word was ";A
1$
1410 PRINT""WOULD YOU LIKE ANOTHER
GAME?";
1420 L$=GET$:IFL$="N"THEN1510
1430 IFL$<"Y"THEN1420
1440 P=0:L1=0
1450 ENDPROC
1460 DEFPROCFIN
1470 J=INKEY(200):CLS:PRINT"
1480 FORJ=8TO9:PRINTTAB(5,J);CHR$141
+"WELL DONE":NEXT
1490 PRINT""YOU SAVED THE INNOCENT
MAN"
1500 GOTO1410
1510 REM Auto Repeat back on
1520 PRINT
1530 *FX11,50

```

# BEEBUGSOFT



## BEEBUG SOFTWARE LIBRARY

Games 1 (£3.00) Starfire (32k)  
Games 2 (£3.00) Moon Lander (16k): 3D Noughts & Crosses (32k)  
Games 3 (£3.00) Shape Match (16k): Mindbender (16k): Rat-Splat (16k)  
Utilities 1 (£3.50) Disassembler (16k): Redefine (16k): Mini Text Editor (32k)

The following programs have been tested on operating systems from 0.1 to 1.0 and are available from 1st September 1982:

Games 4 (£3.00) Magic Eel (32k) Fast moving arcade-type game. Guide the Eel around the screen gobbling up everything it comes across, it gets longer and faster with every bite. When you have eaten through the first frame, this is replaced with harder ones - faster with more obstacles.

Applications 1 (£3.50) SuperPlot (32k) Produces tailored screen representations of (£4.50 from 15th Oct) any function entered. This can be achieved in any of the three major coordinate systems: Cartesian, Polar, or Parametric. SuperPlot comes complete with a 7-page instruction booklet. Explore the world of graphic representation.

Please add 50p postage and packing to all orders regardless of size, then add 15% VAT, and post to BEEBUG Software, 374 Wandsworth Road, London SW8 4TE.

## SOFTWARE COMPETITION ENTRY FORM

Closing date 31th October 1982

Programs of all types (eg games, educational, business etc) are eligible. They should be submitted on a cassette with explanation, instructions, and documentation on paper (typed if possible); and accompanied by the application form below (or a copy of it). Members may submit more than one program, but each entry must be sent under separate cover. If you require the tape to be returned, please enclose a suitable stamped addressed package. Prizes range from £10 to £50 with a £100 reserved prize for programs of special merit.

### Entry Form

Program Title:..... Membership no (essential):.....

Programmer's name:..... Address:.....

Category:.....  
(Games, Educational etc) .....

Will run on Model A.... B....

The program submitted here is my own work, and has not been submitted to another organisation.

I understand that if I am a prize winner my program may be used by BEEBUG for its program library or for publication. In either case this would be with acknowledgement but without further payment. Judging will be carried out by the editors, and their decision is final.

Signed:.....

Date:.....

## IF YOU WRITE TO US

### Back Issues (Members only)

All back issues are kept in print (from April 1982). Send 80p 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.

### Subscriptions Address

BEEBUG  
Dept 1  
374 Wandsworth Rd  
London  
SW8 4TE

### Subscriptions

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

Membership costs: £4.90 for 6 months (5 issues)  
: £8.90 for 1 year (10 issues)  
(Europe - £15 for 1 year)

### Software (Members only)

See details overleaf. Available from the subscriptions address.

### Contributions and Technical Enquiries

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

The Editor  
BEEBUG  
PO Box 50  
St Albans  
Herts  
AL1 2AR

### Ideas, Hints & Tips, Programs, and Longer Articles

All contributions to the magazine are welcome, especially substantial articles, we will pay approximately £30 per page for these. But in this case, please give us warning of anything that you intend to write. In the case of material longer than a page, we would prefer this to be submitted on cassette in a format similar to that produced by our Mini Text editor (June 82 newsletter or Utilities 1 cassette) or as a Beeb datafile providing that you use the bugs fix on page 21 of the July issue, or you have the 1.0 (or later) operating system.

#### NEXT MONTH

New series for newcomers to the BEEB. Full list of BEEBUG discounts. Procedure/Function library. Memory display program - look at the contents of EPROM, ROM and RAM, with tabulated readouts in decimal, hex, binary and ASCII. Music generation ideas. Plus more games programs by winners of our competition. Notes on debugging programs in BBC Basic. Logic pt 2.

BEEBUG NEWSLETTER is edited and produced by Sheridan Williams and Dr David Graham, and its contents are subject to copyright.

Thanks are due to Phyllida Vanstone, Rob Pickering, and John Yale for assistance with this issue.