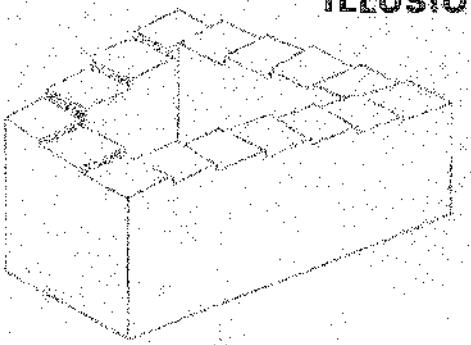
£1.00

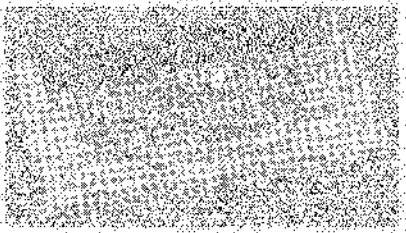
BEEBUG

FOR THE BBC MICRO

Vol 2 No 5 October 1983

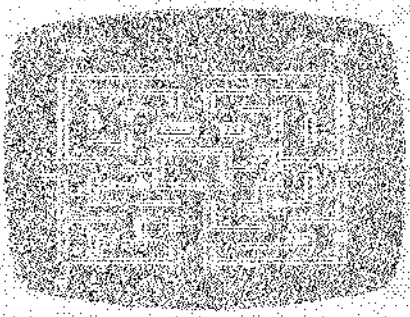
ILLUSIONS





ACORN ELECTRON REVIEWED

DRAWING 3D SURFACES



MUNCHMAN

PLUS

* DISC STRING SEARCH
* TELETEXT MODE
* CUSTOMISING JOYSTICKS
* COMPILERS & INTERPRETERS
* SPEECH SYNTHISER
* GAMES REVIEWED
* PRINTERS REVIEWED
* TAPE RECORDERS REVIEWED

BRITAIN'S LARGEST COMPUTER USER GROUP
MEMBERSHIP EXCEEDS 20,000

EDITORIAL

ELECTRON ORBIT

The new Acorn Electron will soon be appearing in the shops, and we have taken the opportunity of reviewing it in this issue, though we will not be carrying items for the Electron in future issues. As you will be aware, the Electron is essentially a whittled down Model B. It is however a first class machine in its own right, and we have launched a separate user group and magazine called ORBIT to cover this micro. One advantage of this, as we see it, is that we will not be forced to fill the magazine with explanations of how to modify each program to run on the Electron, and we will not be tempted to let mode 7 fall into disuse just for the sake of downward compatibility (the Electron has no mode 7).

PCW SHOW TICKETS

We have been threatened with a delivery of 20000 "£1 off" tickets for the PCW show, allowing entry for £2.00 instead of £3.00; and if they arrive, you will receive one with this magazine. The show is to be held in the Barbican centre from 29 Sept to 2 Oct.

MAGAZINE CASSETTES

This month's magazine cassette contains all the programs from this issue plus two extras: a 3 part harmonisation of Bach's Prelude XII and a machine code screen dump for the NEC 80 printer. One or two members have written regretting our inclusion of additional items on the magazine cassette. These items are things which we have not had room for in the magazine, and often they particularly lend themselves to cassette distribution. In this latest case for example the Bach Prelude is an exremely long listing, but can be placed on the magazine cassette at no extra cost.

However, for those members who require a particular "extra" item on the magazine cassette, but who do not wish to purchase the whole tape, we are instigating a trial photocopy service. Send 50p per item required, plus sae to: Photocopies, BEEBUG, PO Box 50, St Albans, Herts. Please give your membership number.

David Graham.

NEWS AND NOTES FROM THE BEEBUG NETWORK

HINT WINNERS

This month's hint winners are N.Kelly & B.Knott who share the £10 prize and J.P.Carnell who wins the £5 prize.

ORBIT FOR THE ACORN ELECTRON

For further details of our new monthly magazine for the ACORN ELECTRON, see this month's supplement.

MAGAZINE CASSETTE - EXTRA ITEMS

This month's magazine cassette contains two extra items:

PRELUDE XII by J.S. Bach encoded by D.R. Piercy - A three minute Bach prelude in 3 part harmony with visual representation.

NEC 80 SCREEN DUMP by Simon Ainsworth - A useful machine code screen dump (OS 0.1 and 1.2) for all graphics modes of the Beeb, and with two alternative size printouts.

BEEBUG POST

BEEBUG requires a young technical assistant with software expertise on the Beeb. See the supplement for details.

BEEBUG MAGAZINE

GENERAL CONTENTS

PAGE	CONTENTS
2	Editorial
4	Product News
6	The Teletext Mode
8	Fabric Patterns
9	A Versatile Renumber Program
12	The Acorn Electron Reviewed
15	Customising Your Joystick
16	Compilers and Interpreters Explained
17	The Ruston Compiler Reviewed
18	Using the Speech Synthesiser
20	Six Computer Games Reviewed
22	Invisible Alarm
22	Answers to the BEEBUG Crossword
23	Three Printers Reviewed
27	Two Tape Recorders Reviewed
28	Points Arising
29	Drawing 3D Surfaces
31	Disc Sector String Search
34	Postbag
36	Illusions
38	Munch-Man
41	Brain Teaser

HINTS, TIPS & INFO

PAGE	CONTENTS
8	Recovering Poorly Recorded Wordwise Files
8	Local Array Elements
11	Auto Version Numbering
14	Character Set Definitions
16	Single Key Program Save
19	A Visual Verify
21	Count
26	Using an Oki 80 with Wordwise
26	NMI and &D00
26	Coloured View Display
33	Locking Files within a Program
33	VDU 1
33	ASCII Codes in Function-Keys
41	Simplifying Character Definitions
41	Abbreviation for Colour

PROGRAMS

PAGE	CONTENTS
7	Teletext Programs
8	Fabric Patterns
10	Renumber Program
15	Joystick
19	Counting Aloud
22	Invisible Alarm
30	3D Surfaces
32	Disc Sector Search
36	Illusions
38	Munch-Man

ADDITIONAL ITEMS ON MAGAZINE CASSETTE

- Prelude XII by J.S.Bach
- NEC 80 Screen Dump

PRODUCT NEWS NEWS PRODUCT NEWS

ELECTRON LAUNCH AT ACORN USER SHOW

The official launch of the Electron occurred on 23rd August as a prelude to the Acorn User Show. For the occasion, Quentin Bell (Acorn's new Advertising Agency) put on a splendid show fronted by TV personalties, Wendy Craig and Cliff Michelmore. To complete the event, there was a green Electron Fizz cocktail, and rock with the words "Acorn Electron" written all through it. There were also some Electrons on show! These had a variety of Acornsoft games - some of which like Snapper and Monsters were running noticably slower (see Electron review elsewhere). The interesting thing was that you could use SHIFT-BREAK to load the display machines through an undisclosed and hidden 'file-server', joined through the rear connector.

Unlike the Beeb, the Electron will not be sold on mail order, and dealers are expecting supplies any day now. W.H.Smith, for example, are to sell the Electron in some 200 of their retail outlets. The retail price of the Electron will be £199 inc VAT, though it is suggested that there is room for a downward movement in price to meet competition.

WINCHESTER HARD DISKS FOR THE BEEB

Both GSL and Pace have announced 5.25 inch Winchester hard disc systems for the Beeb. These will allow up to 20 Megabytes and more, of storage on a single disc. Demonstration units were on show at the Acorn User Exhibition at the end of August, and the units will soon be on sale. The price tag is around £2000, which puts it well outside the budget of most home users!

Further details from:

Pace Supplies Ltd.
0274-729306

GSL Ltd.
0264-58744

ADVANCED USER GUIDE

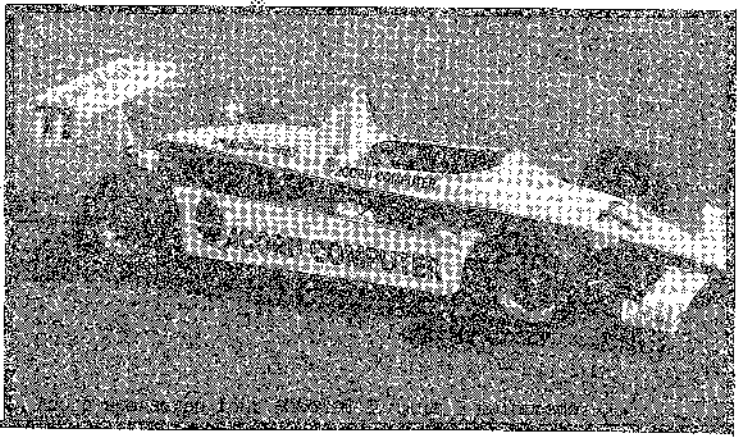
This 500 page manual is published in a similar format to the BBC User guide, and costs £12.95. It has been independently produced by the Cambridge Microcomputer Centre, though it has Acorn's approval even to the extent of containing an authorised circuit diagram for the Beeb. From our early assessments the book appears to be an invaluable guide covering mainly the operating system, the programmable hardware (ie 6845, video ULA etc) and the assembler. It also contains the most complete list of FX calls that we have seen.

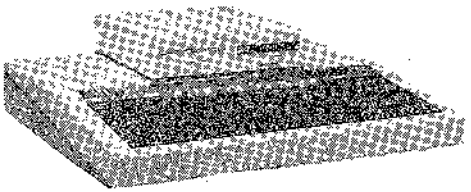
Further details on 0223-355404

DOUBLE DENSITY DISCS

LVL Microsystems and Microware Ltd have both announced double density disc controller boards. These boards plug into the socket used by the 8271 disc controller chip, to give a true double density facility. This effectively doubles the on-line capacity of most disc drives, by doubling the number of sectors in each track of a disc. One further advantage is that the scarce 8271 chip, which does not support double density, is replaced by more readily available controller chips.

Further details from:
LVL on 0602-394000
Microware on 01-272-6237





NEW 3 INCH DISC UNIT

One of the most interesting developments for some time is the new 3 inch disc unit from AMS. The new unit is very small and compact as might be expected. Each disc is contained in a rigid plastic case, and a sprung metal cover over the read/write window provides complete protection from mishandling. The discs should certainly be robust and will be easy to store in this form. Although the disc drives are single sided units providing the standard 100k bytes of online disc storage, the discs themselves are double sided, with A and B sides just like a cassette. Each side can store 100k bytes and is inserted with the A or B side uppermost as required. This gives a total storage capacity per disc of 200k bytes. Initial tests by us would indicate that the units are reliable in operation and that they are comparable with other disc units for the BBC Micro.

The one problem that we foresee with the AMS drives is that of compatibility. It is likely to be some considerable time before software houses offer their products on three inch disks, and until then AMS users will have to purchase software on cassette, much of which may not be easily transferrable to disc.

The new disc units cost £225 for a single drive and £399 for a double drive, including VAT and delivery in both cases. The price also includes cables, manual and utilities. Discs cost £4.95 each or £22.50 for a pack of 5.

The disc units can be ordered direct from Advanced Memory Systems Ltd, Woodside Technology Centre, Green Lane, Appleton, Warrington, Cheshire WA4 5NG.

SECURITY DEVICE FOR YOUR BEEB

You can now fit an EPROM in your BBC Micro which will prevent unauthorised persons from tampering with the machine and will also identify it if it is lost or stolen. The customized EPROM is programmed with your name and address, and also your telephone number, and the machine's serial number if you wish. This information is displayed on the screen when you switch the micro on. Unless you can then type in your personal code the EPROM prevents any further use of the machine. There is of course nothing to prevent the EPROM from being unplugged from the circuit board, but that takes a certain level of know-how, and the time to remove the lid and keyboard. A security conscious owner could always solder in the EPROM.

The device costs £15 from Software Services, 65 South Mossley Hill Road, Allerton, Liverpool L19 9BG. You will need to send the full details to be included in your EPROM plus your choice of five character security code.

NEW OPERATING SYSTEM FROM KENDA

Kenda Software Services Ltd have produced an alternative disc filing system for the BBC Micro known as the Mighty Oak DMFS. This is quite different in structure from the Acorn DFS, being much more like CP/M, an operating system much used on other microcomputers. Disc storage is allocated dynamically so that both data files and directory can grow as and when required. There are no problems in extending files, as additional disc blocks are allocated from wherever they are available. A further plus point with the Kenda system is that it contains its own 2k bytes of internal RAM and does not require any extra memory space within the micro, unlike the Acorn DFS. We hope to review this in a future issue. The Mighty Oak DMFS is available from Kenda at Nutsey Lane, Totton, Southampton SO4 3NB and costs £79.95 plus VAT.

THE TELETEXT MODE

by Mike Williams

Teletext mode or mode 7 on the Beeb is often overlooked by newcomers, and indeed, by many more expert users. But what other mode offers text and graphics, all sixteen colours, single and double height text and uses only 1k of memory to support the screen image? We begin here a series on how to get the most from mode 7. This first article starts from square one, and covers some of the ground covered in our brief article in Vol.2 No.1. The series will soon progress beyond that point.

When you switch on the BBC micro you are by default in mode 7. It is different from the other 7 modes in many respects. It does not have high resolution graphics capability, or user defined graphics characters, and instructions like MOVE, DRAW, PLOT, COLOUR and GCOL do not apply. However, full colour is available, as is a limited set of pre-defined graphics characters. It may also surprise you to know that the text window facility (see User Guide p.387) functions in mode 7.

COLOURED FOREGROUND

A mode 7 screen consists of 25 lines of 40 characters, numbered 0 to 39 across the screen, and 0 to 24 from top to bottom. If you enter mode 7 (either by switching on the machine, or pressing Break, or typing MODE 7 <return>) any text entered from the keyboard will appear in white on a black background.

Generating coloured text is easy. Type the following instruction into the machine:

```
PRINT CHR$129 "BEEBUG"   <return>
```

If you do this you will see the word BEEBUG displayed on the screen in red. Other numbers instead of 129 will produce BEEBUG in different colours as follows:

Value	Colour
129	red
130	green
131	yellow
132	blue
133	magenta
134	cyan
135	white

Try the same instruction with different colours. Now just type in

```
PRINT "BEEBUG"
```

The word will now appear in white, the default colour for text, but there is a further and important difference between them. The coloured version starts one character in from the left hand side of the screen compared with the last version. This is because the first character position contains a Teletext control character (CHR$129 etc) which controls the colour of the following text. All the effects that can be produced in mode 7 are defined by these control characters. They each occupy one position on the screen, and can be copied like any other character using the COPY key. Try an experiment. Use the cursor and copy keys to copy one of the coloured versions of the word "BEEBUG". If you start copying from the first letter "B" the word will be copied in white, because you will not have copied the control character. If you move the cursor to the apparently blank position preceding that letter and copy that as well as "BEEBUG", then "BEEBUG" will appear in the colour you are copying. You can, of course copy the control character from one position and the letters from another. Experimenting like this will help you to understand how the control characters work. Once a control character has been placed on a line of the screen, it will affect all further text until either another control character or the end of the line is reached. Try typing in the following after clearing the screen with CLS:

```
PRINT CHR$129"BEEBUG"CHR$130"MAGAZINE"
```

On the 1.2 Operating System, you can generate the colour codes by simultaneously pressing Shift and one of the function keys.

Here is a list of effects:

```
Press Shift and
 f0  NOTHING
 f1  RED text
 f2  GREEN text
 f3  YELLOW text
 f4  BLUE text
 f5  MAGENTA text
 f6  CYAN text
 f7  WHITE text
 f8  FLASHING text ON
 f9  FLASHING text OFF
```

Mode 7 provides an easy way to create coloured backgrounds for strings of text. This is achieved using control code 157. The effect of this is to produce a background (one character high) in the current foreground text colour. To illustrate this try the following:

```
PRINT CHR$129;CHR$157;CHR$132"Blue text
 on a red background"
```

The first code generates the colour red, the second makes that the background colour, and the third makes the following text blue. To terminate the background colour after the printed text, add ;CHR$156. this returns the background to black.

COLOURED SCREEN

Here is a short program which produces a colourful result on the screen:

```
100 REM Program BEEBUG1
110 MODE 7
120 FOR X=0 TO 28 STEP 7
130 FOR Y=0 TO 6
140 PRINT TAB(X,Y)CHR$(129+Y)"BEEBUG"
150 NEXT Y
160 NEXT X
170 END
```

The (129+Y) in line 140 selects a different colour each time the loop is repeated. Change the step size in line 120 to 5 and rerun the program. Each successive column of words overlaps the previous one but because of the control character at the start of each word there is an apparent space between one column and the next.

Now we know how to produce coloured text, we will look at how we can produce double height text. The control code for this has value 141. Try typing the following line:

```
PRINT CHR$141 "BEEBUG" CHR$141 "BEEBUG"
```

We have "BEEBUG" displayed on the screen in double height characters. It is necessary to print the line of text twice, on both lines to produce double height text. In fact the first printing produces the top half of the double height characters, and the second printing, the bottom half. Double height text can of course be coloured.

```
PRINT CHR$141 CHR$129 "BEEBUG" CHR$141
CHR$129 "BEEBUG"
```

The top and bottom halves of the double height text can also be in different colours. Try changing the CHR$129 (in one position only) to a different colour value.

We will now rewrite our program BEEBUG1 to produce double height characters. Here is the new program called BEEBUG2:

```
100 REM Program BEEBUG2
110 MODE 7
120 FOR X=0 TO 32 STEP 8
130 FOR Y=0 TO 6
140 PRINT TAB(X,2*Y)CHR$141CHR$(129+Y
)"BEEBUG"
145 PRINT TAB(X,2*Y+1)CHR$141CHR$(129
+Y)"BEEBUG"
150 NEXT Y
160 NEXT X
170 END
```

Each time "BEEBUG" appears, it is preceded on the screen by two blank characters, because of the two control characters, one for double height and one for the colour. This is why line 120 has had to be changed, as the letters of "BEEBUG" plus the two control characters add up to 8. If you change the step size in line 130 from 6 to 5 and the 129 in line 145 to 130 and rerun the program, the top and bottom halves of the double height characters will now be in different colours.

Finally, here is a short program which allows you to list any program in coloured text on the screen.

```
100 REM Program BEEBUG3
110 MODE 7
120 FOR Y=0 TO 24
130 PRINT TAB(0,Y)CHR$129;
140 NEXT Y
150 VDU28,1,24,39,0,30
160 END
```

The 129 in line 130 could be replaced by any of the colour codes listed before. The colour control character is placed at the start of every line on the screen and then the text area of the screen is redefined in line 150 to protect the control codes (see user guide). Type the program, run it and then list the program. It will appear in whatever colour you have chosen in line 130. This can also be programmed into a function key.

```
100 REM Program BEEBUG4
110 *KEY0MO.7:F.Y=0TO24:P.TAB(0,Y)CHR
$129;:N.:V.28,1,24,39,0,30|M
120 END
```

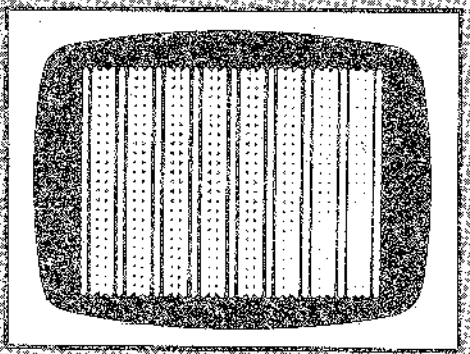
In this program, if you replace the 129 in line 110 by (129+Y MOD 6) you will get rainbow listings.

Next month we will look at Teletext graphics. Meanwhile you may like to try codes 136 and 137. These initiate and cancel flashing text (not background).

FABRIC PATTERNS (16k/32k)

By D.J.Allom

This short program builds up interesting fabric patterns on the screen. As written, it runs in mode 2 and requires a 32k machine, although by changing to MODE 5 in line 150 and changing the start of line 170 to GCOL3,RND(3), it can be made to run on a 16k machine. Pressing any key while a new pattern is being produced will cause the program to pause when that pattern is complete. Pressing any key a second time will cause the program to continue, changing the pattern yet again.



```
100 REM Program FABRICS Version 1A
110 REM Author D.J.Allom
120 REM BEEBUG October 1983
130 REM PRESS ANY KEY TO HOLD PATTERN
140 REM THEN ANY KEY TO CONTINUE
150 MODE2:VDU23;8202;0;0;0;
160 REPEAT
170 GCOL3,RND(7):A%=RND(4)*4
180 FORB%=256 TO 1024 STEP RND(4)*8
190 FORC%=22A TO 800STEP4%
200 PLOT69,B%,C%
210 NEXT C%
220 NEXT B%
230 IF NOT INKEY(0) X=GET
240 UNTIL FALSE
```

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

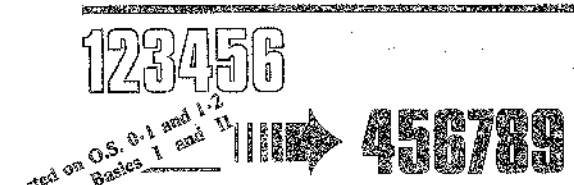
RECOVERING POORLY RECORDED WORDWISE FILES - P.Ells

If you suffer from cassette problems then you should always save files twice. If this is done then you can recover a difficult program by loading most of it from one copy and the difficult blocks from the other one. This is rather tedious but more interesting than re-typing a big file.

Unfortunately Wordwise does not allow these 're-tries' with cassettes as it stops loading whenever there is an error. To get around this use *LOAD "" E00 to load the file (from Basic). Get the file in memory properly and then re-save it with *SAVE "name" E00 +length, length being given as a 4 digit hex number after the last block of the load. This gives another copy of the file to attempt to load into Wordwise.

LOCAL ARRAY ELEMENTS - N.Kelly and B.Knott

To use arrays locally in a procedure define the array globally and use a FOR NEXT loop to declare each individual element LOCAL.



A VERSATILE RENUMBER PROGRAM (16k/32k)

by G & L Pettit

G.& L.Pettit describe a useful utility program which allows selective renumbering of Basic programs. It will work on a model A or B, but on a model A the length of the program to be renumbered is resticted to around 4k.

The RENUMBER command in BBC Basic is limited in its usefulness when a program falls into one of several common categories - for instance, when using procedure or subroutine libraries as an addition to a master program. These master programs will inevitably be of different lengths and, if the RENUMBER command is used on the resulting total programs, the line numbers of the procedures will differ from one program to the next. This makes recognition of the procedures difficult; if subroutine libraries are used, when a subroutine is known only by a line number and not by name, recognition is impossible after renumbering. Also, many program standards require that each program include initial REMs containing dates, author's name, peripherals required, algorithms employed, etc. Since no two programs will have identical REMs, the main program will start at different line numbers if blanket renumbering is used.

It is to cater for these and other situations, where selective renumbering is to the programmer's and end-user's advantage, that the Selective Renumber program has been written. It allows renumbering of the leading statements only, of a portion of the middle of the coding, or of the end statements only. The new starting line number and the new increment (applied to the renumbered lines only) are specified by the user and the program will inform him if there is any overlap between existing line numbers and the new ones BEFORE renumbering takes place. The user may therefore withdraw from an overlap situation and re-specify the parameters, before any confusion occurs.

Although line numbers in REMs are not altered, line numbers in all other statements will be changed, including calls from the non-renumbered statements. Thus if subroutines are being renumbered, calls from the main program are changed to match, or if only a part of the main program is being renumbered, the GOTOs or RESTOREs in the rest of the main program will be changed if necessary.



PROGRAM OPERATION

First type in the selective renumber program and SAVE on disc or cassette in the usual way. To use the program, load the Basic program to be renumbered, as normal, then type

```
PAGE=TOP+&100
LOAD"RENUM"
RUN
```

This procedure will prevent the renumber program over-writing the user's program, assuming this is already resident in memory at &E00 (if using tape) or at &1900 (if using discs). The value TOP+&100 has been chosen so that the renumber program is loaded in at the next page in memory, thus giving the maximum space for work.



When Selective Renumber is run, the program will first check that the user's program has been loaded. If not, an error message is displayed and the Renumber program will stop, to allow the user to reset PAGE to &E00 (or &1900), to load his program and to reset PAGE again to TOP+&100. When RUN is typed, a message requests the range of line numbers to be renumbered. This may be answered by

	, nnn	for all lines up to nnn,
mmm	, nnn	for all lines between mmm and nnn inclusive, or
mmm	,	for all lines from mmm onwards.

The program will next request the new initial line number and new increment, for the section to be renumbered. If the resulting line numbers would conflict with the unchanged numbers, a message will ask the user if he wants to abandon the renumbering. If he abandons, his program will be unaltered, otherwise the renumbering will be carried out without further messages. At the end of the run, the user must return PAGE to its original value by typing PAGE=&E00 (tape) or PAGE=&1900 (disc) before listing his program.

Finally, when running the program, please note that it takes an appreciable time to renumber even a small part of a long program. Don't be impatient, just be thankful you're not having to edit all the line numbers on the screen - and go and file your letters or read BEEBUG, while your micro does the job for you. You won't use this program every few minutes, but when you get in a tight corner with line numbers, and you can't use the BBC RENUMBER command, this may get you back into production again.



VARIABLES USED

The array E% contains the line numbers of the original program, and the array F% contains the equivalent line numbers in the renumbered portion of the program. Initially, all elements of F% are set to -1 and this is used as an indication of the renumbered range.

N% is PAGE for the program to be renumbered
L% is the total no. of lines in the program.
G% is the first line to be renumbered.
H% is the last line to be renumbered.
D% is the new starting line number of the renumbered part.
M% is the increment of the line numbers in the renumbered part.
P% counts the lines as they are renumbered.

```
  100 REM Program RENUM Version 1A
  110 REM Authors G & L Pettit
  120 REM BEEBUG October 1983
  170 N%=&E00:REM This is for tape. For
 disc, N%=&1900
  180 MODE7:PRINTTAB(6,2)"SELECTIVE REN
UMBER PROGRAM"
  190 PROClines
  200 DIME%(L%),F%(L%)
  210 PROColdlines
  220 PROCnewlines
  230 PROCtest
  240 PROCline_ren
  250 PROCinfill
  260 PRINT"Time = "TIME DIV100" secs"
  270 END
  280 :
  290 DEFPROClines
  300 finish=FALSE:L%=0:C%=N%
  310 REPEAT
  320 IF?(C%+1)=&FF finish=TRUE:GOTO340
  330 C%=?(C%+3)+C%:L%=L%+1
  340 UNTILfinish
  350 ENDPROC
  360 :
  370 DEFPROColdlines
  380 C%=N%
  390 FORI%=1TOL%
  400 E%(I%)=(?(C%+1))*256+?(C%+2)
  410 C%=C%+?(C%+3)
  420 NEXT
  430 ENDPROC
  440 :
  450 DEFPROCnewlines
  460 FORI%=1TOL%:F%(I%)=-1:NEXT
  470 N$="***Must be numbers***"
  480 PRINT'"Give line numbers of first
 and last"'"lines to be renumbered."'"
Use the format e.g.  100,250"
  490 INPUT" or   ,250 (renumbers up to
 line 250)"'" or 100,   (ditto from lin
e 250 to end)"',G$,H$
  500 IFASCG$=-1 G%=0:GOTO520
  510 G%=VALG$:IFG%=0ANDG$<>"0"PRINT'N$
:GOTO480
  520 IFASCH$=-1 H%=E%(L%):GOTO540
  530 H%=VALH$:IFH%=0PRINT'N$:GOTO480
  540 INPUT'"Give new line number for f
irst line and increments (e.g.200,10) "
D$,M$
  550 D%=VALD$:M%=VALM$:IFD%=0ORM%=0PRI
NT'N$:GOTO540
  560 P%=0:TIME=0
  570 PRINT'"Delay for processing"
  580 FORI%=1TOL%
  590 IFG%>E%(I%)THENY%=E%(I%):GOTO630
  600 IFH%<E%(I%)GOTO630
  610 Z%=I%
  620 F%(I%)=D%+M%*P%:P%=P%+1
  630 NEXT
  640 P%=D%+M%*(P%-1)
  650 ENDPROC
  660 :
  670 DEFPROCtest
```

```
  680 clash=FALSE
  690 FORI%=1TOL%
  700 Q%=F%(I%):IFQ%=-1GOTO750
  710 FORJ%=1TOL%
  720 IFQ%<E%(J%) J%=L%:GOTO740
  730 IFQ%=E%(J%) PROCclash
  740 NEXT
  750 NEXT
  760 IF(Z%+1)>L% Z%=L%-1
  770 IFD%<=Y% PRINT'"Your ranges overl
ap!":clash=TRUE
  780 IF(Z%+1)>L% OR D%>E%(L%) GOTO800
  790 IFP%>=E%(Z%+1) PRINT'"Your ranges
 overlap!":clash=TRUE
  800 IF clash INPUT'"Do you want to go
 on (Y or N) "A$:IFA$<>"Y"ANDA$<>"y" EN
D
  810 ENDPROC
  820 :
  830 DEFPROCclash
  840 IFQ%<G% OR Q%>H% PRINT"Clash in l
ine ";Q%:clash=TRUE
  850 ENDPROC
  860 :
  870 DEFPROCline_ren
  880 C%=N%
  890 FORI%=1TOL%:F%=F%(I%)
  900 IFF%=-1GOTO930
  910 ?(C%+1)=F%DIV256
  920 ?(C%+2)=F%MOD256
  930 C%=C%+?(C%+3)
  940 NEXT
  950 ENDPROC
  960 :
  970 DEFPROCinfill
  980 finish=FALSE
  990 C%=N%
 1000 REPEAT
 1010 B%=?C%:IFB%=&20 C%=C%+1:GOTO1010
 1020 IFB%=13GOTO1080
 1030 IFB%=&22REPEATC%=C%+1:B%=?C%:UNTI
LB%=&22:C%=C%+1:GOTO1010
 1040 IFB%<&8B GOTO1060
 1050 IFB%=&8B OR B%=&8C OR B%=&E4 OR B
%=&E5 OR B%=&F7 PROCsub
 1060 C%=C%+1:GOTO1100
 1070 PRINT~R%,~S%,~T%
 1080 IF ?(C%+1)=&FF finish=TRUE:GOTO11
00
 1090 C%=C%+4
 1100 UNTIL finish
 1110 ENDPROC
 1120 :
 1130 DEFPROCsub
 1140 C%=C%+1:B%=?C%:IFB%=&20 ORB%=&E5
GOTO1140
 1150 IFB%<>&8DGOTO1350
 1160 C%=C%+1:U%=0
 1170 R%=?C%:S%=?(C%+1):T%=?(C%+2)
 1180 S%=S%-&40
 1190 T%=(T%-&40)*256
 1200 IFR%MOD&10=0 U%=16384:R%=R%+&4
 1210 IFR%=&54 R%=0:GOTO1250
 1220 IFR%=&44 R%=64:GOTO1250
 1230 IFR%=&74 R%=128:GOTO1250
 1240 IFR%=&64 R%=192
 1250 O%=U%+R%+S%+T%
 1260 FORI%=1TOL%
 1270 IFE%(I%)<>O%GOTO1310
 1280 IFF%(I%)=-1:GOTO1300
 1290 PROCinsert
 1300 I%=L%
 1310 NEXT
 1320 C%=C%+2
 1330 B%=?(C%+1):IFB%=&20 C%=C%+1:GOTO1
330
 1340 IFB%=&2C C%=C%+1:GOTO1140
 1350 ENDPROC
 1360 :
 1370 DEFPROCinsert
 1380 U%=F%(I%)
 1390 V%=U% DIV&4000
 1400 W%=U% MOD&4000
 1410 S%=W% MOD64+&40
 1420 T%=W% DIV256+&40
 1430 U%=W% MOD256
 1440 W%=U% DIV64
 1450 IFW%=1 W%=-&10:GOTO1480
 1460 IFW%=2 W%=&20:GOTO1480
 1470 IFW%=3 W%=&10
 1480 R%=W%+&54-4*V%
 1490 ?C%=R%:?(C%+1)=S%:?(C%+2)=T%
 1500 ENDPROC
```

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

AUTO VERSION NUMBERING - J.P. Carnell

When developing a program it makes sense to save each new version of it under a new filename. This can be achieved automatically as follows.

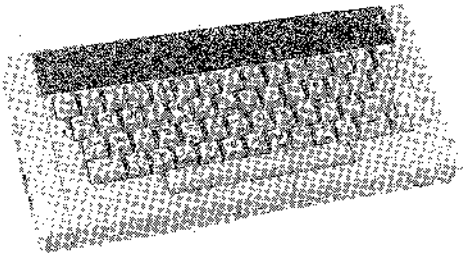
First, let Z%=0, then program f0: *KEY0,Z%=Z%+1|MSAVE"PROG"+STR$(Z%)|M

When you wish to save the new version press f0. This will save it as PROG1, PROG2 etc, with each subsequent depresssion of f0. The resident integer variables must be used for this purpose as they are retained during program editing. Take care also not to use the variable Z% anywhere in the program.

THE ACORN ELECTRON REVIEWED

by David Graham

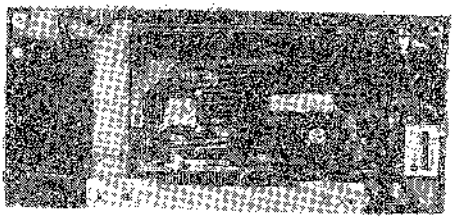
The Acorn Electron has been officially launched, and indications are that it will prove to be a popular machine. As we suggest in our editorial, we shall not be covering the Electron in BEEBUG Magazine - We have begun a separate user group for that machine, with its own dedicated magazine ORBIT. However, we feel that BBC users will be interested to hear how the Electron measures up to the Beeb.



First of all, it looks as though the supply difficulties encountered by early BBC users will not be repeated with the Electron. Our review machine was a production model, and this is a good sign, as it suggests that production lines have been gearing up since well before the launch. In other ways too, Acorn have learned from their experiences with the Beeb. The packaging has been well thought through, and the documentation is far better than that received by early Beeb owners.

ELECTRON PACK

Apart from the machine itself, the pack contains a large moulded plug with built-in transformer, a UHF TV lead, a 'Welcome' cassette, and two books. One of these is a manual, very similar in size to the Beeb manual. This is well planned, with a detailed list of contents. A strong minus point is its lack of index, and we would urge Acorn to append an index to the release version of the manual. The second book 'Start Programming with the Electron' by Masoud Yazdani (published by Addison-Wesley) is an admirable introductory text. What is so impressive about it is that it introduces fully structured programming right from the start. In the first chapter it introduces the PRINT statement alongside PROC and ENDPROC. Procedures are fundamental to structured Basic, and it is nice to see them being introduced in this way. My one serious reservation about this book is again its lack of index in the release version.



MACHINE HARDWARE

The machine is well constructed, and has a good feel to it. It has a compressed BBC keyboard (56 keys). There are no red function keys, but the user-defined keys are still implemented (press FUNC and one of the number keys simultaneously). When used with the alpha keys, the FUNC key allows direct entry of most Basic keywords. This is a useful facility implemented by the Electron's operating system.

One way in which the Electron contrasts sharply with the Beeb is that it has only six connections with the outside world, whereas the model B positively bristles with connectors of various kinds. Somewhat unexpectedly, three of the Electron's sockets are for

the visual display (UHF, video and RGB). This leaves the power input socket, a cassette socket (with motor control), and an edge connector to be used with external interfaces.

Acorn have a number of planned add-ons to upgrade the Electron to almost a model B and beyond, but these will not be available for several months.

INNARDS

Internally the Electron has a higher discrete component count than might be expected in a budget machine. There are two printed circuit boards - the power supply circuitry is separate from the main board, as on the BBC. The main board is dominated by a large square-format ULA chip, and alongside this a 6502A, a Basic ROM (the same as the Beeb's Basic II), a 16k Eprom containing the operating system, and four 64k bit dynamic memory chips. The Electron uses these as if they were eight 32 bit memories, though there is some loss of speed involved in the process. The Electron thus has 32k of RAM, and since PAGE is set to &E00 (as on the BBC micro) the user is left with 17.75k of free memory in mode 6. In mode 2 this falls to a very small 5.75k (as with the Beeb), leaving precious little memory for program storage, and no immediate hope of an add-on second processor to improve things.

MISSING PARTS

Severe economies have been made in whittling the model B down to an Electron, yet these economies have been intelligently made, with only limited loss of facilities. Thus although there is no 6845 video controller chip, many of its functions are carried out in software. What the software cannot achieve is the very fast hardware scrolling on the Beeb, and this together with the hardware sideways scroll have disappeared. Similarly with the SOUND and ENVELOPE commands; software has taken over some of the functions carried out in hardware, and only two sound channels and one envelope are available on the Electron, and the envelope lacks the last six parameters.

The most noticeable 'missing part' of the Beeb is the Teletext mode hardware. The Electron has no mode 7. Modes 0 to 6 are all implemented exactly as on the Beeb, and calling MODE 7 defaults to mode 6, the start-up mode. The loss of mode 7 is a great pity for a number of reasons. It is both an economical and interesting mode, and moreover it brings compatibility with viewdata systems. Its exclusion on financial grounds is fully appreciated of course. Without mode 7, the most economical mode is 6, but this takes 7k more memory than mode 7, and does not allow multicolour screens. We would add that one of the advantages of not covering the Electron in BEEBUG is that we can continue fully to support the use of mode 7 in our programs; without the disincentive of having to carry alternative versions each time that we carry a program using mode 7.

SPEED LOSS

Generally speaking the Electron makes up for missing hardware by incorporating extra software routines in the operating system to perform the same function. The inevitable consequence of this approach is loss of speed. This is most noticeable in the Electron's screen handling in the higher resolution graphics modes. And it is not only the act of changing the display which is time intensive, but even supporting the display takes extra time. For example, if the program TIMETEST is run on a BBC, the printed result is 476 (representing a time of 4.76 seconds to perform the calculation of the sine two hundred times). This timing is, as we would expect, quite independent of the screen mode in use.

```
10 REM TIME TEST
20 TIME=0
30 FOR A=1 TO 200
40 X=SIN(12)
50 NEXT
60 PRINT TIME
```

Running the same program on the Electron gives a result of 558 when the screen is in mode 6, and a massive 1176 in mode 0. This single test suggests that the Electron can approach the calculating speed of the Beeb providing that the central processor (6502) is

not given too much other work to do. Programs which make use of the higher modes will have to get around this in other ways. In some cases it may be possible to do the bulk of the processing while in mode 6, and then change to a higher mode to display the results. Machine code action games using mode 2 for example will need a certain amount of rewriting to speed them up.

THE COMPETITION

This review has been presented in terms of a comparison between the Electron and the BBC micro. Potential purchasers of the Electron will generally not be making such comparisons because the two machines are in quite different price brackets - even more so since the phasing out of the model A. The Electron's competition will be from machines such as the Spectrum, the Dragon and the VIC-20. Broadly speaking the Electron compares extremely favourably with these, offering a combination of excellent text and graphics, two-channel sound, and a full typewriter keyboard. The promised add-ons should prove to be a further incentive in favour of this well designed successor the the BBC micro.

The Electron lacks the following features of a model B:

- Printer port
- RS 423
- Analogue port
- User port
- 1 MHz bus
- Tube
- Paged ROM sockets
- Mode 7
- Full SOUND and ENVELOPE (reduced implementation only)

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

CHARACTER SET DEFINITIONS - P.R.Eggleton, Andrew Armstrong, Matthew Rapier

Redefining characters other than those above the normal character set has some very strange effects. There is only room to store 1 block of 32 characters as soft characters at any one time. The first character you define appears to set which block of 32 characters is to be the re-definable one, and this works well, copying the existing character definitions into the soft space and re-defining as required. The problems come when you define a character in a different block. This results in all the characters in that block being copied into the soft space and the character being re-defined, but the ASCII codes which generate these characters are the ones for the first definition. This is not really a bug as you are not supposed to try and define characters in different 32 byte blocks.

The correct method which does actually work is to use *FX20,n where 'n' is the number of 32 character blocks which you wish to use for soft character definitions. Thus *FX20,1 allows one any of the characters in one 32 character block to be re-defined, and *FX20,3 allows characters in any three 32 character blocks to be re-defined. For maximum flexibility set *FX20,7 which allows all printable characters to be re-defined.

Whenever *FX20 is issued all previously defined characters are re-set to their default values - recovering from wrong definitions. It is necessary to increase PAGE when using Basic programs to allow for the extra memory used by the soft character definitions. OSHWM is reset as needed.

A simple method for setting PAGE correctly is to type:

```
*FX20,n
*BASIC
```

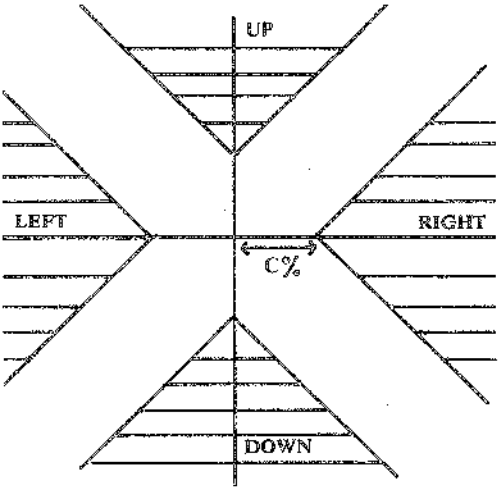
Tested on O.S. 0-1 and 1-2 and on Basics I and II

CUSTOMISING YOUR JOYSTICK (16k/32k)

by D H Moncur and D Earlam

Joysticks are not normally consistent in their responses, that is; the position where you suddenly change from level flight to a nose dive is generally not the same on every joystick.

This program allows you to choose where the positions left, right, up and down are recognised, and so control the sensitivity of movement in a particular direction. This can then compensate for any tendency to e.g. veer left, and also provides a means of using joysticks with any program requiring non-proportional directional control.



You can also, of course, disorientate other players by swapping the directions from their logical North, South, East, West = Up, Down, Right, Left convention.

RUNNING THE PROGRAM

Setting up the joystick entails first moving the lever to the centre position and signifying this with the FIRE button. You will then have to move the lever in each of the four prompted directions and press the FIRE button at the point from where you wish that direction to be sensed. When all four directions are set up, the program will print the direction of the joystick, as it is moved, until the space bar is hit. S% gives a perpendicular tolerance either side of each direction. This may be altered.

To use these routines in your own programs first call PROCsetupjoystick and then call FNjoystickposition each time you wish to determine the state of the joystick. If FNjoystickposition = "RIGHT" then it is to the right, and so on. Your program must of course contain those definitions listed at line 1000 onwards.

```
  10 REM CUSTOMIZE JOYSTICK
  20 REM version 1A
  30 REM D.H.Moncur and D.Earlam
  40 MODE 7
  50 VDU23;8202;0;0;0;
  60 PROCsetupjoystick
  70 REPEAT
  80 PRINTTAB(16,12)FNjoystickposition
  90 UNTIL INKEY-99
 100 END
 110 :
1000 DEF PROCsetupjoystick
1010 DIM X(5),Y(5)
1020 PRINT"CENTRE?"
1030 REPEAT:CENTX=ADVAL(1):CENTY=ADVAL
(2):UNTIL (ADVAL(0) AND 3)=1
1040 FOR DIR=1 TO 4
1050 FORQ=1 TO 1000:NEXT
1060 IF DIR=1 D$="UP"
1070 IF DIR=2 D$="DOWN"
1080 IF DIR=3 D$="LEFT"
1090 IF DIR=4 D$="RIGHT"
1100 PRINTD$"?"
1110 REPEAT:X(DIR)=ADVAL(1):Y(DIR)=ADV
AL(2):UNTIL(ADVAL(0) AND 3)=1
1120 NEXT:CLS
1130 ENDPROC
1140 :
1150 DEF FNjoystickposition
1160 JP$="CENTRE":S%=10000
1170 X=ADVAL(1):Y=ADVAL(2)
1180 IF X<=X(4) AND ABS(CENTY-Y)<S% JP
$="RIGHT "
1190 IF X>=X(3) AND ABS(CENTY-Y)<S% JP
$="LEFT  "
1200 IF Y>=Y(1) AND ABS(CENTX-X)<S% JP
$="UP    "
1210 IF Y<=Y(2) AND ABS(CENTX-X)<S% JP
$="DOWN  "
1220 =JP$
```

COMPILERS AND INTERPRETERS EXPLAINED

by Sheridan Williams

Why is machine code faster than Basic? What is the difference between a Basic Interpreter and a Basic Compiler? - Sheridan explains.

A program written on the Beeb in Basic runs very slowly compared with a program that performs an identical task written in machine code. This should be easier to understand once you realise that the computer's natural language is a succession of binary codes - or machine code. The computer needs to look up each Basic keyword in a table to find the address of the machine code to execute. Algebraic expressions need to be 'parsed' to determine the order in which each operation can be performed. For example, in 2*(3+4) the '+' must be executed before the '*'. A good analogy can be made by assuming that you (a person who only speaks English) are given a set of instructions written entirely in French. It will take you far longer to carry out those instructions because you have to find an interpreter to translate the instructions for you. You could perform the translation yourself if you had a French-English dictionary, and a book on French grammar. When the computer translates from Basic into machine code it too needs a dictionary and a set of rules. The computer's dictionary and rules come in two forms - an 'Interpreter' or a 'Compiler'. The Beeb's Basic Interpreter is already built in, and when you type RUN the majority of time that the computer is working, it is actually looking up the Basic keywords in a table and parsing expressions.

One of the time consuming things about an interpreter is that it interprets every statement each time it encounters it, so if the program is in a loop it may repeat the interpretation dozens of times. Returning to our analogy, if we need to use our instructions on several occasions, it will be much quicker to have them translated into English once and for all (like a compiler) rather than to use an interpreter each time. One of the advantages, however, is that it does not require much spare memory to carry out the process, as it only needs to remember the line currently being worked on. An interpreter also makes the language easy to use interactively, as changes to the program are easily made and tested.

A 'Compiler' on the other hand will take the whole program and translate it into machine code in one go before the program is run. This means that the compiling process can be performed separately from the running of the program, and need not detract from the running time. However, a compiler needs a relatively large amount of memory, or a disc system to perform the compilation, so that it can hold both the 'source' program (the original Basic program) and also the 'object' program (the equivalent machine code program). The great advantage of a compiler is the speed of execution of the resulting machine code program. This could be as much as 30 times faster than an interpreted program, though a program written directly in machine code might be twice as fast again, if well written. The source program is no longer needed for running purposes, and does not need to reside in the computer. This offers the advantage of protection for the author, as the machine code is unintelligible to most people, but the disadvantage is that changes require not only editing, but re-compilation as well.

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

SINGLE KEY PROGRAM SAVING - Anthony Stone

Use *KEY0,SAVE $(PAGE+5)|M

This requires the first line of the program to contain

10REMNAME where NAME is the program name.

Thus the function key will save the program under the name given as its first line.

THE RUSTON COMPILER REVIEWED

by Sheridan Williams

Product: Instant BBC Machine Code
By: Jeremy Ruston
Supplier: INTERFACE, 44-46 Earls Court Rd, London, W8 6EJ
Price: £34.95 inc VAT

This review is based on tests of the compiler carried out at BEEBUG, but also includes the comments sent in by several members who have purchased the compiler. In particular I would like to thank Philip Morris, Koenraad Rutgers and Trevor Baker.

This compiler is written in Basic, and is unusual, compared with others that I have used, in that it only supports a limited subset of BBC Basic. The instructions supported are: LET CLS CLG COLOUR MODE DRAW MOVE END GCOL GOTO GOSUB IF...THEN...[ELSE] OFF PLOT PRINT REM RETURN SOUND VDU FOR...NEXT REPEAT...UNTIL INPUT *FX CALL.
The functions supported are: ADVAL INKEY RND TAB(x,y).
Operators are: + * - AND, OR, # (meaning not equal to), and the square brackets.
Variables allowed are A%-Z% (or A-Z) and only integer arithmetic is supported. You should also be aware that some of the instructions are used in slightly different ways to normal.

Ruston also says that programs should be written with the compiler in mind, rather than developing them in ordinary BBC Basic and then converting them for compilation afterwards. This is sound advice in this case, but only because of the limitations of the compiler. It is true that most Basic compilers require a few changes to be made to the program that runs in interpreted mode, but using this compiler, programs require a major rewrite. This could probably be tolerated in a compiler costing less than £10.

Ruston's compiler, although missing many of BBC Basic's standard features, has a feature called "Sprites": to quote from the book:
"These new statements only work in MODE 4.... Sprites are objects defined on an 8x8 grid - exactly like the user-defined graphics you are used to. The difference being that sprites can be moved around the screen without disturbing what is on the screen already." This could be useful in a games context.

The compiler uses large amounts of memory, (which is understandable,) and makes compiling programs in memory impossible for all but the shortest. In reality compilation must be done from disc/cassette. The compiler is a "three pass" compiler, and requires the file to be read three times, on disc this is no problem, but from cassette it requires rewinding the cassette three times.

The manual is cheaply produced using print from a matrix printer, it is accurate and fairly easy to follow. There is also a listing of the compiler; using this it is possible to reduce the length of the compiler (by removing redundant lines, REMs, and unnecessary spaces). This makes it possible to compile far larger programs in memory.

CONCLUSION

The compiler works well and produces quite fast running object code, which typically runs between 5 and 8 times faster. On the whole it is a rather expensive piece of software, the price means that it is for specialists only; and my impression is that it is far too expensive for the limited subset of BBC Basic that it supports.

We should also point out that Interface require royalties on any program developed using this compiler.

[This sounds rather like a piano maker wanting a royalty on every piece of music composed on his pianos! - Ed.]



USING THE SPEECH SYNTHESISER (16k)

by David Graham

David Graham describes two routines for calling the speech synthesizer from Basic. The first is a simple method of generating a string of spoken words. The second is a set of procedures which can produce spoken numerical output in the range -999999 to +999999. This latter task, which has applications in speaking clocks, spoken games scores and in the educational field is not as easy as it might first be thought.

SPEAKING PHRASES

As suggested in our review of the Acorn speech upgrade (BEEBUG Vol.2 No.3 P.14), the speech synthesiser can be called up either from machine code (using OSBYTE and OSWORD calls) or direct from Basic using variants of the SOUND command. The simplest Basic call takes the form:

```
SOUND -1,N,0,0
```

The -1 indicates that the call is to the speech synthesiser, and that the vocabulary is in PHROM A (ie PHrase ROM A). The manual details this more fully.

If you want to produce a string of words - or a number of strings, then a routine of the following kind may prove to be useful. This one speaks the words: "Press any key to start".

```
100 READ T
110 FOR A = 1 TO T
120 READ N
130 SOUND -1,N,0,0
140 NEXT
150 DATA 5,246,168,212,50,260
```

The word codes for the words to be spoken (obtained from the speech system user guide) are held in the data statement at the end. The first item (read in as T) is the total number of words that are to be spoken together. Any combination of words and numbers can be strung together in this way, and you can insert a pause with Data of the value 127.

SPEAKING NUMBERS

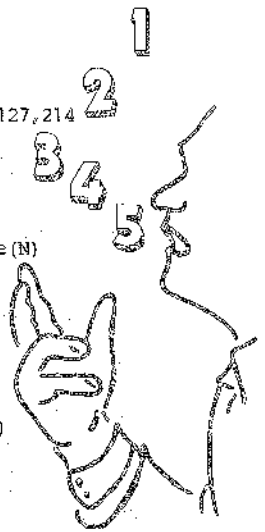
Making the speech unit count, or speak numbers chosen at will, is somewhat more difficult. The problem arises because a routine is required to recreate any given number in exactly the way in which it is spoken. Take for example the number 21119. It is fairly easy to make the machine give the number as Two One One One Nine. But this is somewhat uninspiring, and all the facilities exist to speak the number as we normally speak it. To achieve this a routine would be required to add the following pieces together: TWEN TY ONE THOUSAND ONE HUNDRED AND NINE TEEN.

The series of procedures listed below from line 1000 onward achieves this for numbers between -999999 and +999999. The main procedure PROCspeaknumber(N) is used in conjunction with a set of other procedures which handle the ranges of numbers indicated in their procedure names. If a number is out of range, you will be told so verbally. At present there is no facility for treating decimals, but this would be an easy addition, since these are spoken as a series of straight digits (eg - say to yourself 118.118).

As an example of its use the procedures are preceded by a 5 line program which repeatedly calls the procedure PROCspeaknumber to count from 0 to 999999. As you can see, PROCspeaknumber is called with a single parameter (N). This is the number to be spoken.

In more interesting applications, the Beeb's TIME function could be used to create a speaking clock, or PROCspeaknumber could be called to give scores in a game or to set and answer arithmetic problems in an educational program.

```
   10 REM COUNTING ALOUD
   20 REM Version 1.1A
   30 REM David Graham
   40 REM THE 5 LINES BELOW
   50 REM DEMONSTRATE THE
   60 REM PROCEDURE
   70 :
   80 FORN=1 TO 1000000
   90 PROCspeaknumber(N)
  100 PROCspeak(127)
  110 NEXT
  120 END
  130 :
 1000 DEFPROCspeaknumber(N)
 1010 IF N<0 PROCnegative(N):N=ABS(N)
 1020 IF N<10 PROCunits(N)
 1030 IF N>=10 AND N<13 PROCtentotwelve
(N)
 1040 IF N>=13 AND N<20 PROCteens(N)
 1050 IF N>=20 AND N<100 PROCtens(N)
 1060 IF N>=100 AND N<1000 PROChundreds
(N)
 1070 IF N>=1000 AND N<1E6 PROCthousand
s(N)
 1080 IF N>=1E6 PROCoutofrange
 1090 ENDPROC
 1100 :
 1110 DEFPROCspeak(P)
 1120 SOUND-1,P,0,0
 1130 ENDPROC
 1140 :
 1150 DEFPROCnegative(N)
 1160 PROCspeak(219)
 1170 PROCspeak(127)
 1180 ENDPROC
 1190 :
 1200 DEFPROCoutofrange
 1210 RESTORE:FORA=1 TO 5
 1220 READ word
 1230 PROCspeak(word)
 1240 NEXT
 1250 ENDPROC
 1260 :
 1270 DATA 229,127,143,127,214
 1280 :
 1290 DEFPROCunits(N)
 1300 PROCspeak(N+48)
 1310 ENDPROC
 1320 :
 1330 DEFPROCtentotwelve(N)
 1340 IF N=10 N=264
 1350 IF N=11 N=190
 1360 IF N=12 N=273
 1370 PROCspeak(N)
 1380 ENDPROC
 1390 :
 1400 DEFPROCteens(N)
 1410 PROCspeak(2*N+120)
 1420 PROCspeak(135)
 1430 ENDPROC
 1440 :
 1450 DEFPROCtens(N)
 1460  PROCspeak((N DIV 10)*2+140)
 1470 PROCspeak(137)
 1480 IF N MOD 10>0 PROCunits(N MOD 10)
 1490 ENDPROC
 1500 :
 1510 DEFPROChundreds(N)
 1520 PROCunits(N DIV 100)
 1530 PROCspeak(140)
 1540 IF N MOD 100<>0 THEN PROCspeak(97)
 1550 IF N MOD 100<>0 PROCspeaknumber(N
MOD 100)
 1560 ENDPROC
 1570 :
 1580 DEFPROCthousands(N)
 1590 PROCspeaknumber(N DIV 1000)
 1600 PROCspeak(44)
 1610 IF N MOD 1000>0 AND N MOD 1000<10
0 PROCspeak(97)
 1620 IF N MOD 1000<>0 PROCspeaknumber(
N MOD 1000)
 1630 ENDPROC
```



HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

A VISUAL VERIFY – R.D.Smith

Rather than use the *LOAD "" 8000 to verify a program or data file, an interesting alternative is

```
MODE 1
*LOAD "" 4400
```

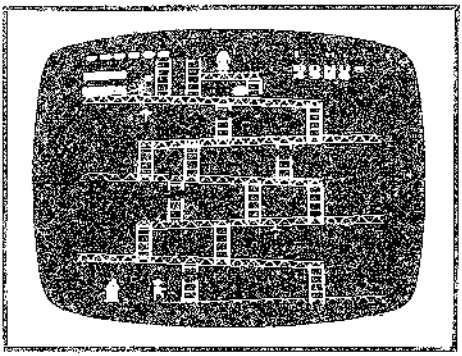
(Note that when using discs the quotes should contain the appropriate file name.) The result is colourful, and gives a visual indication of the file length.

SIX COMPUTER GAMES REVIEWED

by Alan Webster

Alan Webster reviews a new batch of games, and finds them all to be above average.

Name : Killer Gorilla
Supplier: Program Power
Price : £7.95 inc. VAT
Rating : *****



Killer Gorilla is a very well written version of the arcade game Krazy Kong. Having only watched the arcade game and not played it, I found the game both compelling and difficult.

The game revolves around trying to rescue a maiden from the reaches of a gorilla, and to do this you have to negotiate a series of ladders, conveyor belts and walkways whilst trying to avoid the fireballs, barrels and many other objects.

There are four different screens in all, and after completing the fourth screen the game restarts but is a little different than before.

As an avid games player I would strongly recommend this game to anyone (although the key layout requires 3 hands!) because at its low price it is certainly excellent value for money.

Name : Swoop
Supplier: Program Power
Price : £7.95 inc. VAT
Rating : ***

Swoop is yet another cross-version of Space Invaders / Galaxians, but better than most versions around. Rolling two games into one always seems a good idea, but very rarely pays off.

In Swoop you have to shoot the aliens who move across the screen Invaders style. Then one breaks off and 'Swoops' down to bombard your base. If the alien misses, it plants an 'egg' which restricts the movement of your base as you cannot cross over these eggs. After a short time the eggs explode.

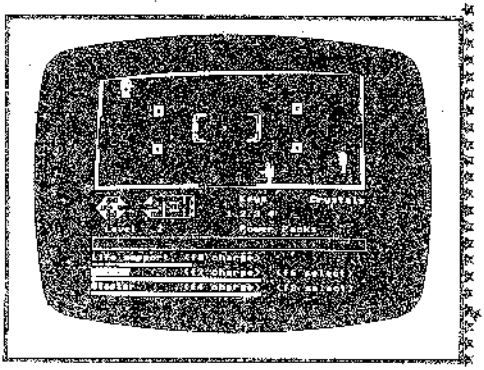
Although not one of the most exciting games, it is certainly hard, and good value for money.

Name : Android Attack
Supplier: Computer Concepts
Price : £8.95 inc. VAT
Rating : *****

Android Attack is an very well written game and is very difficult to play. It requires great dexterity and quick thought and must be ranked amongst the hardest games around, espacially at skill level 19. There are so many different features of this game that there is just not enough space to describe them all, and the number of different characters to be found in the game (I don't know if I even found them all) is quite astonishing.

The game involves the player's man (and it looks realistic) running around a maze shooting at different types of android, whose graphic representation is stunning. The player must shoot or blow up the androids to stand any chance of winning. A high quality game worthy of any games library.

Name : Space Adventure
Supplier: Virgin Games
Price : £7.95 inc. VAT
Rating : *****



This is an 87 room adventure with a difference. It is entirely in animated graphics. The game involves you trying to find four crystals inside a deserted space ship. These crystals are hidden behind four locked doors, the keys to which are strewn around the ship.

Your progress is repeatedly hampered by androids who shoot to kill you. If you are not careful your life support system will become drained and you will die. You can however shoot back at the androids with blasters and phasers, but these too have a limited supply of power.

Randomly positioned around the ship are power-packs which can be used to re-charge either your life support system, your blaster or your phaser.

Overall, an excellent game, with the smoothest graphics I have ever seen; and a very good cassette from this new software company, a subsidury of Virgin Records.

Name : Moonraider
Supplier: Program Power
Price : £7.95 inc. VAT
Rating : ****

At last, a comparable 'Scramble' game to that of Acornsoft's Rocket Raid. Moonraider has some good colourful graphics, and is a smooth, fast free-flowing game. Unlike Rocket Raid, you can choose which section of the game to start on, and you can also choose the difficulty. A nice touch is added at the end with the inclusion of a smiling face. Because the game is inexpensive and very good, I might be tempted to place this above Rocket Raid as the top Scramble game.

Name : Starship Command
Supplier: Acornsoft
Price : £9.95 inc. VAT
Rating : ****

Starship Command is a 'Star-Trek' type game with a difference. It is also a full graphics game in which you have to pilot a succession of spaceships through space in order to obtain points. You collect points by killing enemy ships which hover around you menacingly. The game is one of the best 'non-arcade' style games that Acornsoft produce, and is well worth buying, as the graphics, speed, and on-screen presentation are very near perfect.

The only drawback (there has to be one!) is that there are more controls than you have fingers.

SPECIAL OFFER

We have been able to arrange special offers for members on the best of the above games reviewed in this issue. Details of the offer will be found elsewhere in the magazine. We hope to make this a regular feature of software reviewed by BEEBUG.

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

COUNT - Peter Lusmore

The COUNT value is only incremented when Basic is used to print. If output is occurring from machine code through OSWRCH directly then the value of COUNT will be incorrect. To allow for this you can increment count directly. It is stored at &1E. Note that this location can be modified directly on a tube system as this is part of Basic workspace and so can be modified as normal memory in the tube memory space. This address is correct for Basic I and Basic II, though later releases may use a different location.

INVISIBLE ALARM

by Gordon Weston

The Alarm program listed below will sit invisibly in your machine and produce a beep after a predefined delay. This can be anything from a fraction of a second to several hundred years (though this latter has not been tested - yet). The alarm is not affected by the running of Basic or machine code programs provided Break is not pressed or the memory used is not overwritten.

When you run the program it asks for a time in seconds. Just enter this, and sit and wait. If you require delays greater than a year and a third, then you should alter the FF in line 80.

If you wish to execute your own section of code after this time delay, rather than generate a beep, then delete lines 50 and 60 and 130 to 200, and set lines 90 and 100 to point to the low and high bytes (currently 7D and 0) respectively of the start address of your code.

```
10 REM Invisible Alarm
20 INPUT "No. of seconds"n
30 n=n*100
40 *FX14,5                    130 P%=&7D
50 !&70=&FFFF0000             140 TOP%=
60 !&74=&0014000C8            150 LDX#&70
70 !&78=-n                    160 LDY#0
80 ?&7C=&FF                   170 LDA#&7
90 ?&220=&7D                  180 JSR&FFF1
100 ?&221=0                   190 RTS
110 A%=4:X%=&78:Y%=0          200 ]
120 CALL &FFF1                210 END
```

Explanation:

The time is set as 5 bytes in lines 70 and 80, the counter scanning upwards from the negative value given to zero. &7C contains the high byte and &78 the 4 low bytes.

Line 40 enables the interval timer crossing zero event. Lines 50 and 60 set up a control block in memory (&70 to &77). This control block defines the sound parameters to be used. Lines 90 and 100 set the indirect vector to &007D. Line 110 sets up the OSWORD parameters with A=4 (see User Guide page 460). Line 120 calls OSWORD. The assembly language at lines 140 to 200 set the pointers to the control block and call OSWORD, producing the beep.

ANSWERS TO THE BEEBUG CROSSWORD

Published in the August issue Vol.2 No.4

ACROSS

1.COMPUTER
4.ATN
7.HARDWARE
11.ADVAL
12.POS
13.CLS
15.OFF
17.DIV
18.ON
22.FA
23.LOMEM
25.ARGUMENTS
27.UNTIL
29.MOD
30.ERL
31.CMP
32.ULA
33.PLP
34.REM
36.NMI
37.EXT
38.OPT
40.COUNT
42.MODES
43.PRINT
44.CR
45.IF
47.NO
48.OR
50.PLING
52.LOCAL
54.JIM
56.CYAN
57.AUTO
58.ONE
59.CONTROL
62.EXP
64.ERROR
66.TAN
68.INPUT
70.INKEY
71.GOSUB
72.SOUND

DOWN

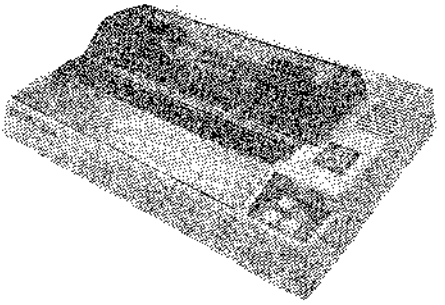
1.CAPSLOCK
2.MOS
3.UNCOMMITTED LOGIC ARRAY
4.ADD
5.TV
6.NAK
8.DEFAULT LOGICAL COLOURS
9.AND
10.ENVELOPE
14.LN
16.FF
19.SGN
20.AMPLITUDE
21.RND
24.MAP
25.ADDRESSING
26.SERIAL PORT
28.TOP
35.MACHINE
36.NETWORK
39.ESCAPE
41.OSFILE
46.TILDES
49.ECONET
51.LIST
53.INC
55.VAL
58.OSCLI
60.NOT
61.RUN
63.POUND
65.OLD
67.ACS
69.NEW

THREE PRINTERS REVIEWED

The Shinwa-CTI, Star DP840 and the NEC PC-8023B-C

Shinwa-CTI CP-80 Printer
Reviewed by Charles R.W. Lyne

Price: £332 inc VAT.
Supplier: Key Computer Centres Ltd.,
Enterprise House,
Terrace Road,
Walton-on-Thames.



The CP-80 is made by a Japanese firm called C.T.I. (Creative Technology International Inc.), and seems to have been designed as a drop in replacement for the MX-80 in almost all situations.

To this end, all the many software features are identical in operation to the Epson, making it very convenient for those who have programs and word processors already set up for the Epson.

The hardware is outwardly similar to the Epson, being 377(W) x 295(D) x 125(H) millimetres in size, and 5.3 Kg. in weight. The paper feed knob is on the left, but the switches/indicators are, familiarly on the front right corner. The controls are:-

On/Off Line Form Feed Line Feed

(The two feed switches are only operative when the printer is "Off Line").
Indicators:-

Power On On Line Ready Paper Out

Maximum print speed is the same as the Epson, at 80 c.p.s. (the new Epsons are faster), and it has the normal 80 character standard print width, though the characters are slightly smaller, so the eighty characters take less of the paper's width. This feature also allows the compressed character format to have up to 142 columns. The double width characters are still limited to 40 columns. Unlike the standard Epson, the paper feed mechanism has options for tractor feed (sprocketed paper) and friction feed (single sheets) at no extra cost.

Interface options are nominally the same as in the Epson, i.e. parallel (Centronics) built in, and a variety of RS232 interface boards by Epson and other manufacturers are supposed to be directly compatible. There does appear to be a problem at present though with the design of the printer electronics, which are not fully compatible with standard Epson interface boards. In fact, the printer will be damaged if a standard RS 232 interface is connected! This of course is less likely to affect BBC "B" users who have the parallel interface available as standard. It is worth noting that the effective baud rate of the parallel interface is up to about 40,000 baud, so the data transfer part of the printing operation is faster than on an unbuffered RS 232 interface.

Other facilities of note are, the italic characters, compressed

Printers previously reviewed

BEEBUG Vol.1 No.10
Seikosha GP 100 A
Epson MX80/100
Olivetti Praxis 35
Tandy CGP-115

BEEBUG Vol.2 No.3
Epson RX80 and FX80
Seikosha GP 250 X

characters, emphasized characters, double printed characters, and combinations of these options.

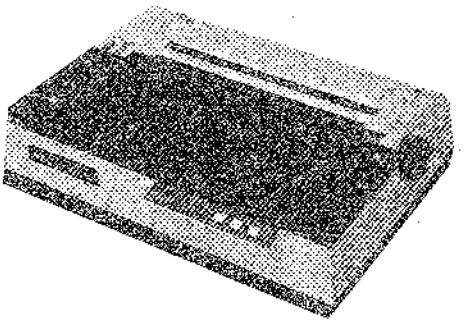
The CP-80 also has superscript, for mathematical formulae and subscript, for chemical formulae and other applications.

This printer does not have the international character sets, selectable under software control, as the Epson does, but the £ sign is character number 129, which is of course, easily defined from Wordwise using the "DP129" command, or in Basic as "CHR$(129)". There is also a command for continuous underlining and a useful range of graphics characters is provided. There is also a hardware option for producing slashed zeros.

The mechanical and electronic construction appears to be very sound, but the design of the case leaves a little to be desired concerning access to the internal switches. In conclusion, a good printer which compares favourably with the many other 80 column printers, but at the same price as the Epson printers, I personally would buy an Epson.

NEC PC-8023B-C Printer
Reviewed by Mike Williams

Price: £368 inc VAT
(see note after review).
Supplier: Technomatic Ltd
17 Burnley Road,
London NW10 1ED.



This NEC dot matrix printer is an attractive and compact model. The main on/off switch is at the side with the other main switches and warning lights at the front. Unlike some printers, when first switched on the printer is not on-line and has to be selected by means of the SEL switch. The front panel switches, when the printer is deselected, provide Line Feed and Top of Form functions. The printer has a standard parallel interface for connection to the BBC Micro.

The printer will take either friction fed or sprocket fed paper. The cover is in three parts which are removed for paper and ribbon changing and for access to the DIP switches. The hinged perspex paper cutter has a very coarse serrated edge and a lot of flexing occurred in the cover when refastening the paper cutter. Apart from this both friction and sprocket feed worked well.

The DIP switches are situated, unusually, inside and at the bottom of the machine. There are two switches covered by a clear plastic strip which clearly labels the two switches. However, access to the switches means reaching through the the wires which move the print head backwards and forwards, and this was quite awkward, to say the least. One interesting feature is that any of 5 different character sets can be selected including UK, German and Swedish. The European character sets also include the Greek alphabet as standard which could be useful for scientists and mathematicians.

Printing speed is normally 100 cps and the printer produces quite a variety of whining noises while working (indeed some ingenious programmer might even be tempted to make it play music!).

The printer is able to print Pica (10 cpi. 80 ch. per line) and Elite (12 cpi. 96 ch. per line) characters and to print both of these condensed and enlarged. In addition, proportional spacing can also be selected and characters can be enhanced and/or underlined. You can set up to 16 horizontal tab positions and up to 6 vertical tab positions, useful for printing standard formats. As well as the normal character set, the printer will produce dot mode graphics, though apparently not compatible with that of

Epson printers. The printer also has an internal graphics character set which includes the symbols for the card suits (hearts, clubs, diamonds and spades) and various other shapes. Teletext graphics characters are not available nor can characters be defined by the user.

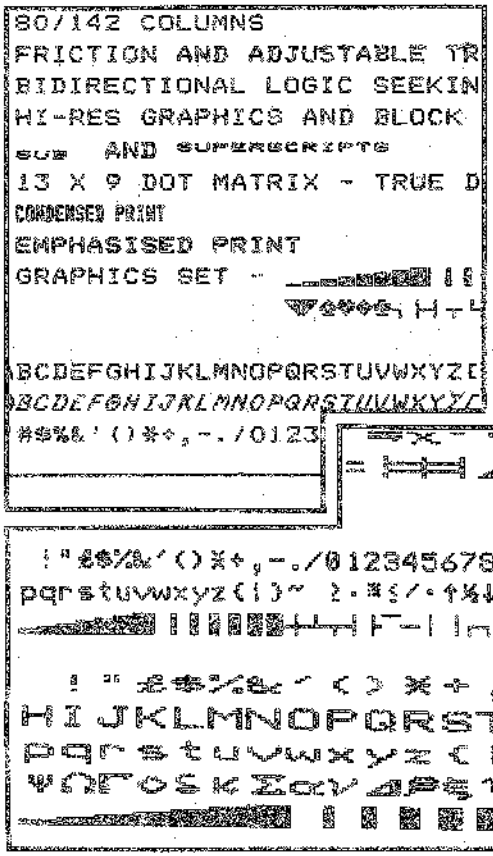
The handbook supplied with the printer is quite comprehensive but could be improved. Elementary things, like how to remove the top cover are missing and the general layout makes it difficult to locate specific pieces of information quickly.

All in all this is a very reasonable printer for the price. It looks well made and provides a full printing specification as one would expect at this price. There are just a few niggles as mentioned above that could have been avoided by slightly better design and construction.

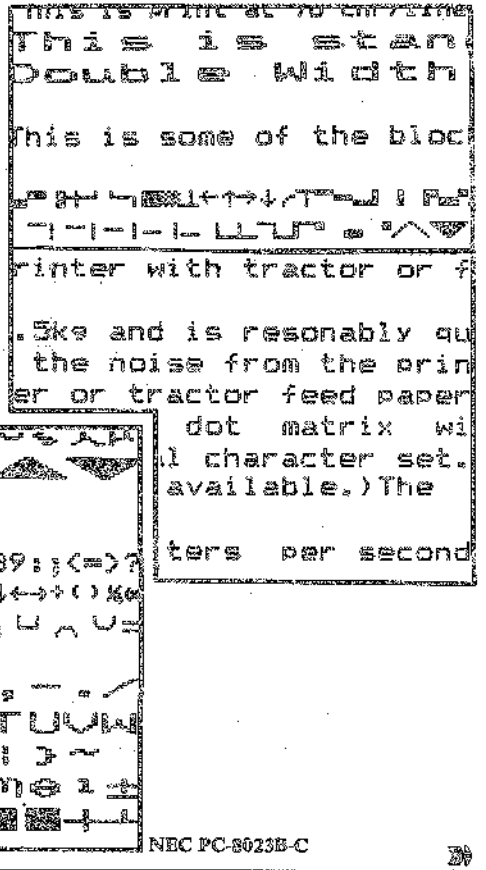
SPECIAL OFFER

We are very grateful to Technomatic for loaning us a printer for review and their good co-operation at other times. Technomatic will sell the NEC printer to BEEBUG members at the special price of £358 inc VAT plus £8 for carriage until the end of October. Send your order direct to Technomatic and quote your BEEBUG membership number, referring directly to this review. Technomatic do, of course, supply many other types of printer.

The Shinwa-CTI



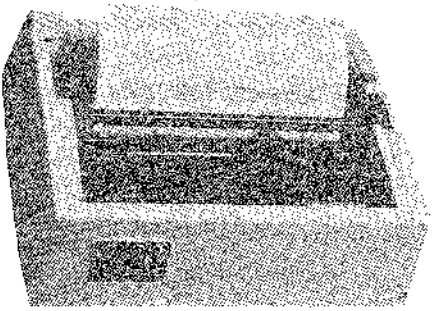
Star DP840



NEC PC-8023B-C

STAR DP840 Printer
Reviewed by Ian Gilbert

Cost: £276 inc VAT.
Supplier: CJE Microcomputers, Worthing.



This is a dot matrix printer with tractor and friction feed, weighing a sturdy 8.5kg and reasonably quiet in operation - the plastic cover reducing the noise from the print head. It is easy to use with A4 sheets of paper or tractor feed paper. (Sizes from 3" up to 10"). Printing is by a 9 by 7 dot matrix with all characters, block graphics and an international character set. (A high resolution graphics option for screen dumps is available). The printer does not have lower case decenders.

Printing speed is fast at 80 characters per second and is bi-directional with logical seeking. This combines to give rapid printing of large programs and documents, the characters being very clear and readble. The print ribbon is contained on two standard typewriter reels.

It prints at 80, 96 or 132 characters per line and has a double width printing option. Although the print is condensed at 132 chrs/line it is still clear and useful for producing large tables of information.

Centronics or RS232 interfaces are available for direct connection to the BBC Micro. The high resolution graphics are available by changing the printers character ROM. This ROM is produced by CJE Microcomputers of Worthing (Tel 0903 213900) who can also supply the printer with the ROM and lead for around £260 incl. delivery.

I have been using this printer for nine months and have found it an invaluable aid to programming. It produces clear and rapid print and although it does not have lower case decenders it is good value for money.

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

USING AN OKI 80 WITH WORDWISE - L.Schapira

When using this printer with Wordwise it is necessary to execute *FX6,13. This solves problems of generating spaces between parts of the text.

NMI and &D00 - C.Marshall

On O.S. 1.2 the NMI (non-maskable interrupt) routine is put at &D00. Because of this the O.S. puts an RTI instruction there as its first action on Break, and the DFS will write its own NMI routine there as well. Thus code put at &D00 will be corrupted when Break is pressed. Writing to anywhere in the &D00 block with any filing system ROM present could destroy memory or crash the filing system. Not only is the NMI routine in this block but also an extended vector space and pointers to the beginning of private work space areas.

COLOURED VIEW DISPLAY - S.Fagg

The usual way to change the colour of the display in modes other than 7 would be to use the VDU command, however VIEW does not support this. To get at the colours you must use the control characters more directly e.g.
*KEY0,|S|A|B|@|@|@|M etc.
This will give green text on a black background when f0 is pressed in command mode. While in text mode the function keys are disabled, they may be re-enabled with *FX225,1.

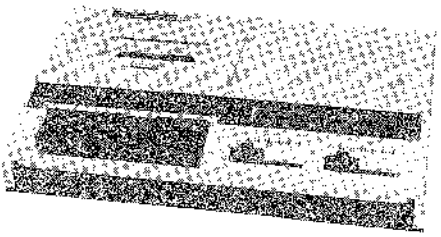
TWO TAPE RECORDERS REVIEWED

By Mathew Rapier

We reviewed four tape recorders for the BBC Micro in BEEBUG Vol.2 No.1. (The CCR800 model from W.H.Smith reviewed then is still available and remains a good buy.) We now review the new Acorn Data Recorder, which replaces the previous Ferguson model, and a rather more expensive recorder from Marantz.

THE BBC MICROCOMPUTER SYSTEM DATA RECORDER

SUPPLIER: Vector Marketing
COST : £29.90 inc VAT



This new recorder replaces the Ferguson 3T07 recorder supplied by Acorn. It is not, contrary to rumour, a digital recorder. The name DATA RECORDER does not appear to signify any radically new design. Externally the BBC recorder is coloured to match the computer and is reasonably well presented with all the controls at the front together with the easily readable VU meter. The tape counter, however, is behind the tape and is rather small and hard to read. On the review machine the finish is already coming off giving a rather battered look despite only a small amount of use. All the standard controls are present including pause. It is unfortunately not possible to "cue"; i.e. have play and fast forward down at the same time, a facility which would greatly speed up searching for programs.

Being designed for use with the BBC micro, this cassette recorder has only a single 7 pin din socket. The record and playback levels are fixed so that you cannot set them incorrectly. The result of this is that the user has no control whatsoever on the levels used by the recorder. The volume and tone controls only affect the speaker output so that you can listen to your program loading quietly. The machine has motor control, working through the 7 pin connector. Unfortunately there is no override on this so that you are forced to use the computer's motor control commands to rewind the tape. This can be very inconvenient. Most tape recorders have the motor control on a separate plug which can be easily removed.

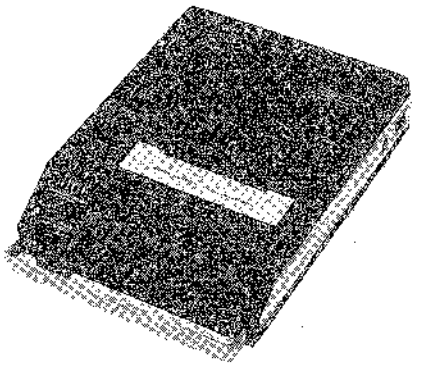
The Welcome tape supplied with the micro loaded first time, but a program recorded on this tape recorder was NOT reliable, one block needing a second attempt. One of my own tapes would also not load at all, having difficulty even getting the name of the file correct. I found this very surprising, but I could probably have got the tapes to load if there had been some control over the tape recorder's output. It is inevitable that there will be some variations between BBC micros and between the tape recorders themselves and, with no control at all, these variations cannot be compensated for. It is a pity that a tape recorder specially produced for use with the BBC micro is not more reliable in use. It could probably be used for music quite successfully as the sound quality is reasonable, though the lack of a microphone limits its use in other areas.

THE MARANTZ SUPERSCOPE C190

SUPPLIER: Comet
COST : £42.90 inc VAT

This tape recorder is the top of a range of Superscope machines. It is well built with a tough finish. My own Superscope is two and a half years old and is still in good condition. All the

standard controls including pause are there, and a "cue and review" facility. The latter is the ability to fast forward or rewind whilst in play mode, which considerably eases the finding of programs in the middle of a tape. A standard set of jack sockets and a DIN socket are present, and the machine works reliably with both. There are also recording level and variable playback speed controls. The tape counter is large but in the wrong place to be read easily.



The user has complete control over recording and playback levels. The volume and tone controls both affect the output level to both types of connector and these settings can be varied for different tapes, though they are not particularly critical. The manual recording level is very useful as automatic level controls are usually confused by computer output. A setting of +1 dB on the VU meter gives very reliable operation. When using the DIN output the noise also goes to the speaker. This requires the use of a BBC User Guide on top to keep it down to a reasonable volume. If the jack outputs are used then there is no noise output from the speaker, but the sound of the program loading can prove very useful when trying to find a program. On recording there is silence, just the movement of the meter showing that data is coming in.

This tape recorder appears to be able to load anything. With creative use of the volume, tone and variable playback speed even the most appalling tapes can be read. I have found the variable speed extremely useful on some commercial tapes. In use, for recording programs as well, complete reliability is the norm. I quote a letter from Mr. T.Bromilow - "I can report that after 12 months quite heavy use, I have never lost, to my knowledge, a single-bit - and I've never cleaned the heads". My own experience with the Superscope and that of people I know who use it with other computers is exactly the same. This machine can be strongly recommended, though it is fairly expensive at around £40. Unfortunately, it is absolutely useless for music reproduction as it sounds dreadful, but don't be put off. I have never encountered a file that could not be loaded with this machine.

POINTS ARISING

Cassette programs to disc - Vol.2 No.4

We must apologise for an editorial error in this article. In attempting to clarify things(!) we have confused the file length with the load address in two statements. To get it right, replace both occurrences of the expression:

```
FILENAME 05 0585 FFFF1111 FFFFeeee
```

with

```
FILENAME 05 1111 FFFFxxxx FFFFeeee
```

Utility Editor - Vol.2 No.1

There is a problem with this program when the line number falls between 3328 and 3583 (&D00 and &DFF). The problem is that one byte is equal to 13, the code for Return. To solve this problem type in the following line:

```
900 FORJ%=I%-1 TO I%-254 STEP-1:IF ?(J%
-2)=13 AND ?(J%-3)<>13 AND ?(J%-4)<>13
PRINT AST$;?J%+?(J%-1)*256,;:J%=I%-254:
IF DISP%X%=I%+K%:PRINT" ";:REPEAT:Y%=?X
%:PRINT CHR$(Y%);:X%=X%+1:UNTIL Y%=13 O
R Y%=58 OR Y%=61:PRINT
```

Thanks to M Wood for pointing this out.

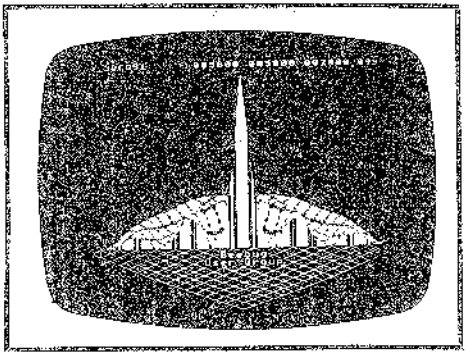
Tested on O.S. 0.1 and 1.2 and on Basics 1 and II

DRAWING 3D SURFACES (16k/32k)

by Derek Chown

Displaying 3D surfaces on the screen is interesting, but complicated to work out. Derek Chown has done all the hard work and presents you with a program for projecting a "picture" onto a 3D surface.

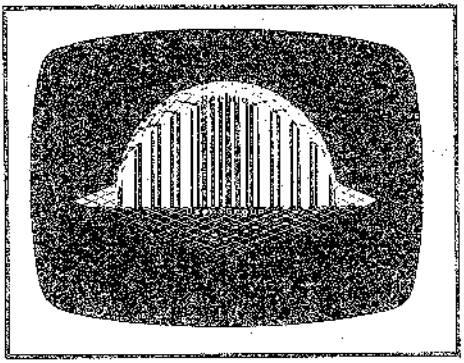
Essentially this program draws a completely flat (2D) surface with a grid superimposed on it. Text may then be written onto the grid. A second phase of the program creates a 3D projection of the grid, in such a way that any lettering is raised and shaped against the contours. The screen shots give some idea of what the program can do. The next three paragraphs give a slightly more technical explanation.



The 3D plot is achieved by placing the pixel of the "picture" with coordinates (x,y) at a height z, calculated from x and y.
That is z = f(x,y).

This particular method of plotting enables opaque surfaces to be represented, covered with a grid to give the picture depth. The method is a little cunning. First an isometric view of a grid is drawn to represent the plane of x and y, and at each pixel within the grid a "match-stick" is erected whose length is z. Match-sticks which stand on the grid lines are drawn in black, and matches that stand between the grid lines are drawn in white. In this way the grid lines are effectively drawn on the surface by the visible tips of the black match-stick.

Drawing of the match-sticks begins at the point furthest from view, and progresses towards the foreground. The foreground match-sticks therefore obscure all or part of the match-sticks behind them. This is what makes the completed surface opaque.



A call to a suitable screen dump could be inserted at line 610 where indicated by the REM statement. Remember that if you use a screen dump routine the printer image will be the reverse of that on the screen (black changes to white, and vice-versa).

It is amusing to insert into the program a statement or two to place words in the grid before the plotting begins. The match-sticks erected in the letters of the words will be in black, and will produce distorted printing as if scraped on the surface. (Rather in the nature of the Cern Giant drawn in chalk on the hillside!). Strictly, though, these are not drawn accurately, because the originals were not drawn in the same isometric projection as the grid lines.

Unfortunately, this program takes a long time to run, especially in the high resolution modes, so I have provided a means of producing a very crude impression of the final outcome for scaling purposes, etc. This is done by plotting z at, say, every fifth pixel. The "speed factor" (sf%), as I

have called it, controls this feature. A speed factor of 1 gives the most detailed (and slowest) picture; higher values giving progressively cruder but faster images.

It is quite possible that some of the trigonometrical calculations in the program are inefficient, so perhaps there are rewards of speed for those willing to tinker with the various projections. For my own part, it is the fact that these mathematical pictures are so easily created with such startling precision that is so fascinating.

The 3D surface itself is specified in lines 750 and 760. The program as listed specifies a hemisphere. Here are two other surfaces to try:

Surface 2: 1/cos^2

```
750 S=SQR((X%-M% DIV 2)^2 + (Y%-M% DIV
2)^2)*4*PI/M%
760 S=(COS(S)/(S+2))^2*3000
```

Surface 3: Discontinuity

```
750 S=((((M%-X%)*Y%)MOD
160000)*(M%-Y%)*(M%-X%))
760 S=S/300000000
```

```
  100 REM Program 3DPLOT Version 1A
  110 REM Author Derek Chown
  120 REM BEEBUG October 1983
  130 ON ERROR GOTO 150
  140 GOTO 190
  150 ON ERROR OFF
  160 VDU3:MODE7:REPORT
  170 PRINT" at line ";ERL:END
  180
  190 MODE7
  200 PRINT':INPUT"Enter speed factor: "
sf%
  210 MODE4:VDU23;8202;0;0;0;
  220 D%=500
  230 W%=1280
  240 C%=(W%+1)DIV2
  250 B%=(D%+1)DIV2
  260 CS=W%/32
  270 SN=D%/32
  280 FOR L1%=0 TO 16:REM DRAW GRID
  290 GX%=CS*L1%+.5
  300 X%=CS*(16-L1%)+.5
  310 GY%=SN*L1%+.5
  320 Y%=SN*(16-L1%)+.5
  330 PLOT 4,C%+GX%-1,B%+Y%
  340 PLOT 5,C%-X%,B%-GY%
  350 PLOT 4,C%-GX%,B%+Y%
  360 PLOT 5,C%+X%-1,B%-GY%
  370 NEXT L1%
  380 PRINTTAB(17,24)"Beebug"TAB(15,25)
  390 TN=D%/W%
  400 SN=ATN(TN)
  410 CS=COS(SN)
  420 SN=SIN(SN)
  430 M%=C%*SQR(2)+.5
  440 FOR BT%=D% TO 0 STEP -4*sf%:REM MO
DES 0,1,2,4,5
  450 T=ABS(B%-BT%)/TN
  460 L1%=T-3.5
  470 L2%=W%-T+3.5
  480 FOR CT%=L1% TO L2% STEP 4*sf%:REM
MODES 1,4
  490 REM FOR CT%=L1% TO L2% STEP 8*sf%:
REM MODES 2,5
  500 REM FOR CT%=L1% TO L2% STEP 2*sf%:
REM MODE 0
  510 PROCXY(CT%,BT%):REM CONVERT PIXEL
ADDRESS TO (X%,Y%)
  520 PROCPT(X%,Y%):REM CALC Z%
  530 GX%=(X%+4)DIV8
  540 GY%=(Y%+4)DIV8
  550 GCOL0,POINT(CT%,BT%)EOR1
  560 PLOT69,CT%,BT%
  570 PLOT5,CT%,SY%
  580 NEXT CT%
  590 NEXT BT%
  600
  610 REM Call screen dump here
  620
  630 END
  640
  650 DEFPROCPT(X%,Y%)
  660 LOCAL P,R,S
  670
  680 REM**************************
  690
  700
  710 REM RE-CODE THIS SECTION FOR ANY O
THER FUNCTIONS YOU CAN THINK OF
  720
  730 REM   HEMISPHERE
  740
  750 R%=(X%-M%/2)^2+(Y%-M%/2)^2:S%=M%/2
  760 IF R%>=S%*S% S=0 ELSE S=SQR(M%*(X%
+Y%)-S%*S%-X%*X%-Y%*Y%)
  770
  780
  790 REM**************************
  800
  810 Z%=S+.5:REM THIS VALUE IS PLOTTED
  820 P=.7854-ATN((Y%+1E-10)/(X%+1E-20))
  830 R=SQR(X%^2+Y%^2)
  840 SX%=C%-R*SIN(P)+.5
  850 SY%=D%-R*COS(P)*SN+S*CS+.5
  860 ENDPROC
  870 DEF PROCXY(SX%,SY%)
  880 LOCAL P,R
  890 P=TAN(.7854-ATN((C%-SX%)/(D%-SY%+1
E-20)*SN))
  900 R=(C%-SX%)^2+((D%-SY%)/SN)^2
  910 R=SQR(R/(1+P^2))
  920 X%=R+.5
  930 Y%=P*R+.5
  940 ENDPROC
```

Tested on Basics I and II

DISC SECTOR STRING SEARCH (32k)

by Colin Opie

Colin describes a program that will help you find a file on disc after it has been deleted from the disc directory.

In the July issue, Vol.2 No.3, we described an extremely useful general purpose 'Disc Sector Editor' which, amongst other things, could be used to restore files which have been accidently deleted from a disc. Clearly the task of file restoration is relatively easy if a current '*INFO' listing exists of the disc in question, but what if this isn't available? With an 80 track disc you could end up searching 800 sectors to find the beginning of your file! We now describe a search program which can be used to search for a string of characters on a specified disc. Using this program approximately 2.5 sectors can be checked every second giving a scan rate of around 1.5 minutes for a 40 track disc. This is an immense saving in time compared to a manual check.
[It is interesting to note that the program takes twice as long to run on our Torch Disc Pack due to the 'head & motor'/'head & select' access mechanism outlined in our Torch review in BEEBUG Vol.2 No.1 and in 'Postbag' in BEEBUG Vol.2 No.3. Ed.]

DESCRIPTION OF PROGRAM

The program is listed at the end of this article and consists of a mixture of Basic and Assembler. The assembler part is the string search procedure, hence providing a good search speed. (Using a Basic string search procedure, this takes three times as long).

It is possible to obtain a listing of the assembled code by setting the appropriate boolean variable 'listasm' in line 150. If this variable is set to 'TRUE' then a listing will be given each time the program is run. Setting it to 'FALSE' stops the listing from occurring. Usually the listing will only be required for debugging any typing-in errors or for obtaining a printed copy, (set VDU2 first). During normal use of the utility the listing would sensibly be switched out.

Checks are constantly made for disc read errors and a display of the current track and sector being searched is maintained continuously. Results are displayed in the same order and manner as would be required for entry in the Disc Sector Editor (i.e. as 'track/sector' in decimal, and as 'relative sector' in hexadecimal). In this way it is very easy to take the results and use them in conjunction with the editor in order to locate the lost program.

USING THE PROGRAM

On running the utility a series of questions are presented:

1) Drive: - Enter the drive number (0 to 3) of the disc which you want to be searched,

2) String: - Give the string (maximum 80 characters - though only 8 will be sensibly displayed) which is to be searched for. Basic tokens cannot be included in this string.

3) Printer: - Typing 'Y' in reply to this will enable the results, showing the location(s) of the specified string, to be sent to a printer as well as to the screen. The printer must previously have been initialised using the normal *FX commands as necessary. Typing 'N' will send the results to the screen only.

All three replies require a <return> to terminate them. Once they have been entered the search program will check the specified disc to see if it is formatted. If it is, then the number of tracks on that disc will be displayed in the top right-hand corner of the screen, otherwise an error report will be given and the program will stop. Assuming all is well, a display will also be initialised giving the current track and sector being searched.

Whenever the specified string is found, the 'track/sector' location (in decimal) is displayed along with the 'relative sector' version (in hexadecimal). All results are displayed in the lower part of the screen in a scrolling page. In this way the top half of the screen showing the initial inputs, the track and sector count, and the headings for the results, will remain throughout the search. If a printer is not being used it will be necessary to jot down the results before they get scrolled off the screen.

When the program has finished, it reports the fact, and the track and sector count will be found to be the last ones available for the specified disc, (on a 40 track disc this will be 'track 39','sector 9'). Now the Disc Sector Editor can be loaded and run. Entering either form of the result(s) obtained from the search program will enable the appropriate sectors to be displayed by the editor. You can use these displayed sectors to help in locating the start and end of the deleted program, and therefore obtain the required parameters to enable the program to be restored on the disc catalogue.

```
  100 REM Program DSERCH  Version 1A
  110 REM DISC SECTOR STRING SEARCH
  120 REM C.N.OPIE  BEEBUG  October 83
  130 REM
  140 ON ERROR GOTO 420
  150 listasm=FALSE:listasm=3+NOT(lista
sm)
  160 MODE7:@%=2:tmp1=&70:tmp2=&71:tmpy
=&72
  170 DIM blc% 256,blk% 12,str% 80,code
% 100
  180 DIM flag% 1,blklen% 1,tlen% 1,bch
klen% 1
  190 blk%!1=blc%:blk%?5=3:blk%?9=&21
  200 vert%=0:printer=FALSE
  210
  220 PROCassemble:PROCtitle:PROCcollec
t
  230 rcd%=0:blk%?6=&53:blk%?7=0:blk%?8
=1
  240 X%=blk%MOD256:Y%=blk%DIV256:A%=&7
F:CALL &FFF1:rcd%=blk%?10
  250 IF rcd%<>0 THEN SOUND1,-15,10,20:
PRINT"Disc error ":GOTO400
  260 IF blc%?7<>&90 AND blc%?7<>&20 TH
EN SOUND1,-15,10,20:PRINT "Unknown Disc
 Format":GOTO400
  270 IF blc%?7=&90 THEN tr%=40 ELSE tr
%=80
  280 PRINTTAB(17,2);CHR$129;tr%;" trac
k disc"
  290 PRINTTAB(17,3);CHR$134;"Track:";S
PC(5);"Sector: "
  300 FOR TRK%=0 TO tr%-1:FOR SCT%=0 TO
 9
  310 VDU28,0,24,39,0:PRINTTAB(25,3);TR
K%;TAB(37,3);SCT%:VDU28,0,24,39,7
  320 rcd%=0:blk%?7=TRK%:blk%?8=SCT%
  330 X%=blk%MOD256:Y%=blk%DIV256:A%=&7
F:CALL &FFF1
  340 IF rcd%<>0 THEN SOUND1,-15,10,20:
PRINTTAB(0,vert%);CHR$129;"Disc error "
:GOTO400
  350 ?blklen%=255
  360 CALL search:IF ?flag%>0 THEN PROC
report:GOTO 360
  370 NEXT SCT%,TRK%
  380 PRINTTAB(0,vert%);'CHR$129;"Finis
hed"'
  390 VDU28,0,24,39,0
  400 END
  410
  420 REM End of Program
  430 ON ERROR OFF
  440 MODE7:VDU3
  450 REPORT:PRINT" at ";ERL
  460 END
  470
  480 DEFPROCassemble
  490 FOR pass%=0 TO listasm STEP lista
sm:P%=code%
  500 [OPT pass%
  510 .search
  520 LDY blklen%     \Set Buffer Index
  530 INY
  540 .s2
  550 LDX #0          \Set String Index
  560 LDA str%,X      \Found Start?
  570 CMP blc%,Y
  580 BEQ check
  590 INY
  600 CPY bchklen%    \End of buffer?
  610 BEQ s3
  620 BCS s3
  630 JMP s2          \No-keep looking
  640 .s3
  650 LDA #0          \Yes-so reset flag
  660 STA flag%
  670 RTS             \and finish.
  680 \---
  690 .check
  700 LDA #1
  710 STA tmp1        \No.of chars searc
hed
  720 STA tmp2        \No.of chars match
ed
  730 STY tmpy        \Save buffer index
```

```
  740 .c1
  750 CMP tlen%       \Match?
  760 BEQ found       \Got it!!
  770 INC tmp1        \Keep looking...
  780 INY
  790 INX
  800 LDA str%,X
  810 CMP blc%,Y
  820 BNE c2          \No match!
  830 LDA tmp1
  840 JMP c1
  850 .c2
  860 LDY tmpy        \Restore buffer in
dex
  870 INY
  880 JMP s2
  890 .found
  900 LDA #1          \Set found flag
  910 STA flag%
  920 STY blklen%     \update buffer index
  930 RTS             \and return.
  940 ]
  950 NEXT pass%:CLS:ENDPROC
  960
  970 DEFPROCreport
  980 IF printer THEN VDU2
  990 @%=2:PRINTTAB(5,vert%);TRK%;TAB(3
);", ";SCT%;TAB(24);~(10*TRK%+SCT%)
 1000 VDU3
 1010 SOUND1,-15,150,1:vert%=VPOS
 1020 ENDPROC
 1030
 1040 DEFPROCcollect
 1050 PRINTTAB(1,2);"Drive  :";:INPUT""
D%:D%=D%MOD4:?blk%=D%
 1060 PRINTTAB(1,3);"String :";:INPUT""
$str%:K%=LEN($str%):?tlen%=K%:?bchklen%
=256-K%
 1070 PRINTTAB(1,4);"Printer(Y/N):";:IN
PUTP$
 1080 P%=LEFT$(P$,1):IF P$="Y" OR P$="y
" THEN printer=TRUE
 1090 PRINTTAB(8,4);SPC(19)
 1100 PRINTTAB(8,4);":ON";:IF NOT print
er THEN VDU8:PRINT "FF"
 1110 ENDPROC
 1120
 1130 DEFPROCtitle
 1140 VDU23;11,0;0;0;0
 1150 PRINT TAB(4,0);CHR$141;CHR$132;CH
R$157;CHR$131;"Sector String Search  ";
CHR$156
 1160 PRINT TAB(4,1);CHR$141;CHR$132;CH
R$157;CHR$131;"Sector String Search  ";
CHR$156
 1170 PRINTTAB(0,5);CHR$146;STRING$(38,
CHR$172)
 1180 PRINTTAB(2,6);CHR$134;"Track/Sect
or(Dec)  Rel.Sector(Hex)"
 1190 ENDPROC
```

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

LOCKING FILES WITHIN A PROGRAM - Trevor White

The following procedure allows a file to be locked from within a program by using a call to OSCLI. Users with Basic II can simplify the routine considerably.

For Basic I use:

```
 5DIM buf% 20
10DEF PROClock(name$)
20$buf%="ACCESS "+name$+" L"
30X%=buf%:Y%=buf% DIV 256
40CALL &FFF7
50ENDPROC
```

For Basic II use:

```
10DEF PROClock(name$)
20OSCLI"ACCESS "+name$+" L"
30ENDPROC
```

VDU 1 - B.Tennent

B.Tennent reminds us that control codes sent to a printer must always be preceded by VDU 1 (and also by VDU 2 on the new O.S.). Thus to generate the codes 27,70, use the following:

```
VDU 2,1,27,1,70
```

VDU 1 sends the next character to the printer only and so does not intercept the codes as it does when control codes are used normally.

ASCII CODES IN FUNCTION-KEYS - Tim Renouf

```
|?  gives 127 - delete
|!|? gives 255
```

POSTBAG POSTBAG POSTBAG POSTBAG POSTBAG POSTBAG

DISC STORES TOO LARGE!

ar Sir,
We note references that the BBC mputer will not support the full torage capacity of some disc drives. or example the Mitsubishi 1Mbyte (400k n BBC).

This presumably has some reference o the point made in the Disc Review in BEEBUG Vol.1 No.8 page 7 where it says hese large drives are capable of olding more than the BBC can handle.

Can the reason for this be elaborated please? What is the limiting factor? On the face of it the 100k of the BBC drive is more than the computer can handle but usually broken up into a number of short programs.

Tom Ward

Reply: Disc storage is always divided up into a large number of relatively small (typically 256 bytes) units of storage which we will call blocks. Each block is referenced by the micro by means of an address which identifies the exact physical location of a block on the disc. It is thus the hardware in the micro, in the form of the disc interface, together with the disc operating system, in this case DFS, which actually determines how much disc storage can be used, rather than the physical storage capacity of the disc itself. The BBC micro uses a disc controller chip (8271) which is only able to handle single density (10 sectors per track) discs. A double density disc controller would immediately double the potential storage capacity, but this would also need a modified version of DFS.

Mr Ward's letter also raises a second point. Of course nearly any disc can store, in total, far more information than can be held in a computer's memory. However, no program stored on disc can sensibly be larger than the micro's memory (32k for the Beeb less graphics and work space).

CHOICE OF 40 OR 80 TRACK DISCS

Dear Sir,

I wonder if you could explain why discs designed to run on 40 track drives cannot be read by 80 track drives. It seems to me that one should buy one of each to gain the best of all worlds.

Malcolm Waters

Reply: 40 track discs CAN be read on 80 track drives by writing a routine which doubles the step rate of the disc heads. Watford Electronics have built this into their own DFS and can be called up by a simple *FX command. See 'Watford DFS review' in BEEBUG Vol.2 No.4. A similar effect can also be achieved in the disc drive hardware, and some 80 track drives are supplied with a switch allowing them to read and write 40 track discs. Both Cumana and Microware supply switched drives. You should note however that when an 80 track drive is used to read a 40 track disc it is only using half of the track width. This will not generally be as reliable as reading a 40 track disc with a 40 track head and drive mechanism.

Dear Sir,

I have beaten the high score in hedgehog. My score was 3080. I like your magazine because it has lots of good games.

C.Eberhardt (aged 8)

Reply: Nice try, but J. Davis has beaten you to it. See high scores table in this issue.

DISC SECTOR EDITOR

Dear Sir,
re. your Disc Sector Editor (BEEBUG Vol.2 No.3).

Thanks for a useful program. The magazine arrived the same day I corrupted the catalogue of one of my discs. Using the program, I was able to recover all the programs on the disc.

I did find a small problem, however. If a filename is used to indicate the sector to be edited, the file will not be found if it is locked. The reason for this is that the top bit of the directory letter is set for a locked file. The remedy is to ignore the top bit when testing the filename, viz

```
1170 IF   F$=fnt$   AND   DIR$=CHR$(svd
AND127) THEN off=I%-dir%:I%=99999
```

I hope this is of use to someone.

Michael Smith

WOULD YOUR BEEB PASS O LEVEL ARITHMETIC?

Try running the following program:

```
10 FOR N=0 TO 1 STEP 0.001
20 PRINT N*1000
30 NEXT N
```

This should print integers 0 to 999 (i.e. 0.001 x 1000 = 1, 0.002 x 1000 = 2, etc.). Try running the program and see what you get!

My O.S. 1.2, BASIC II machine only scored 10.5% in this maths test, so I shan't enter it for the O level exam.

A.J.S. McMillan

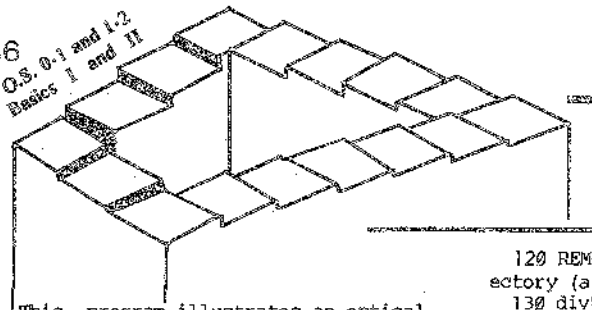
Reply: Eric Bramley has also written to us making the same point. Both correspondents are of course quite correct in pointing out inaccuracies in arithmetic on the BBC micro. This is true of all digital computers, and results from the fact that the computer stores all numbers in binary and not in decimal. Now as long as you deal only in whole (or integer) numbers, then no problems arise, as these can always be exactly represented in binary. Where numbers involving a fractional part are involved the situation is different. Many decimal fractions when converted to binary become recurring fractions and have to be rounded to the number of bits (digits) allowed by the micro for such numbers. For example the decimal fraction 0.2 when converted to binary becomes 0.00110011001100..... Thus inaccuracy is built in. Furthermore, arithmetic operations can often magnify the inaccuracy sufficiently to affect the running of the program. The problem cannot be avoided. Either confine such critical sections to using integer numbers only or adjust the values obtained.

A COMPUTER WIDOW'S LAMENT!

Our Computer
The pest in our house is not a mouse,
But a machine with which you play
Each and every single day.
You switch it on and press a button
Onto the screen they come astrutting
People and dragons, spiders and things
In and out of dungeons in Winter and Spring
Bleeping and buzzing, whirring around
Making such an awful sound.
One of these days I suppose I shall see
The point in all this machinery.

W.C. Beckley
(wife)

Tested on O.S. 0-1 and 1.2 and on Basics I and II



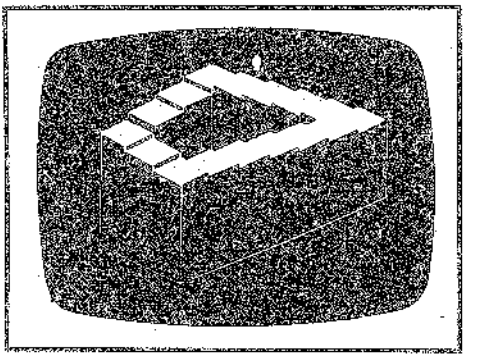
ILLUSIONS (32k)

by M Inglis

This program illustrates an optical and an auditory illusion. It is based on the well known Escher drawing of a never-ending staircase. This version is animated with a figure continually spiralling up the staircase to the accompaniment of musical chords which appear to go higher and higher - indefinitely.

The program requires no user participation, and can be terminated by pressing any key. The user can change some of the variables declared at the start of the program, to give a different number of steps, and a different speed for the figure, but only make slight adjustments to these and only change them one at a time otherwise invalid combinations might be obtained.

Acknowledgements : L.S & R. Penrose; Brit.J.Psych. Vol.49,31.



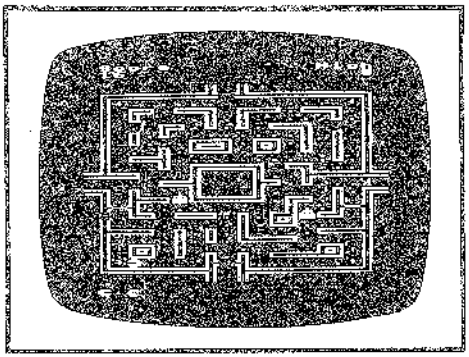
```
  10 REM Program name : Illusions
  20 REM Author : M. Inglis
  30 REM Version I A
  40
  50 ON ERROR GOTO 1670
  60 REM-Number of steps on each side
  70 i%=3:j%=4:k%=6:l%=7
  80 REM-Height of each step
  90 h%=20
 100 REM-Height of jump
 110 kht%=80
 120 REM-Number of points in jump traj
ectory (affects speed of mannikin)
 130 div%=10
 140 REM-Perspective angle
 150 theta=1.15
 160 REM-Colours of mannikin and stair
case
 170 CM%=3:CS%=2
 180 REM-Median sound pitch
 190  P0%=62
 200 REM-End of preset variables
 210
 220 MODE1
 230 P2%=P0%:P3%=P0%+16:P1%=P0%-20
 240 GCOL 0,2:VDU19,2,CS%;0;19,3,CM%;0
;19,1,CM%;0;
 250 d%=(l%-1)*h%+300
 260 VDU23,224,60,60,60,24,255,255,255
,255
 270 VDU23,225,255,189,189,60,60,60,60
,126
 280 steps%=i%+j%+k%+l%-4
 290 DIM centres%(1,steps%-1)
 300 centindex%=0
 310 sinth=SIN(theta):costh=COS(theta)
 320 sinfi=(j%-1%)*sinth/(i%-k%)
 330 cosfi=COS(ASN(sinfi))
 340 a%=h%*steps%/((l%-j%)*costh+(k%-i
%)*cosfi)
 350 asinfi=a%*sinfi:acosfi=a%*cosfi
 360 asinth=a%*sinth:acosth=a%*costh
 370 halfx%=(asinth+asinfi)/2
 380 halfy%=(acosth-acosfi)/2
 390 by%=d%+(i%-1)*h%+i%*acosfi
 400 MOVE 0,by%
 410 DRAW 0,i%*acosfi
 420 fx%=i%*asinfi
 430 DRAW fx%,0
 440 DRAW fx%,d%
 450 MOVE fx%,0
 460 DRAW fx%+l%*asinth,l%*acosth
 470 PLOT 1,0,d%-(l%-1)*h%
 480 x%=0:y%=by%
 490 MOVE 0,by%
 500
 510 FOR m%=1 TO j%-1
 520 x%=x%+asinth:y%=y%+h%+acosth
 530 MOVE x%,y%
 540 PROCrhomb(1,4)
 550 NEXT
 560 FOR m%=1 TO k%-1
 570 x%=x%+asinfi:y%=y%+h%-acosfi
 580 IF m%=1 THEN vx%=x%:vy%=y%
 590 MOVE x%,y%
```

```
  600 PROCrhomb(0,0)
  610 NEXT
  620
  630 FOR m%=1 TO 1%-1
  640 x%=x%-asinth:y%=y%+h%-acosth
  650 MOVE x%,y%
  660 PROCrhomb(-1,0)
  670 NEXT
  680
  690 FOR m%=1 TO i%-1
  700 x%=x%-asinfi:y%=y%+h%+acosfi
  710 MOVE x%,y%
  720 PROCrhomb(3,4)
  730 NEXT
  740 MOVE vx%,vy%:vy%=vy%-3*h%
  750 PLOT 21,vx%,vy%
  760 REPEAT
  770 vy%=vy%-h%:PLOT 21,vx%,vy%
  780 UNTIL POINT(vx%,vy%-5)<>0
  790
  800 centindex%=0
  810 notfirst%=FALSE
  820 T=TIME:VDU5
  830 REPEAT
  840 oldx=centres%(0,centindex%)
  850 oldy=centres%(1,centindex%)
  860 centindex%=centindex%+1
  870 IF centindex%=steps% THEN centind
ex%=0
  880 newx%=centres%(0,centindex%)
  890 newy%=centres%(1,centindex%)
  900  difx%=oldx-newx%
  910 a=-4*kht%/(difx%*difx%)
  920 b=(oldy-newy%)/difx% - a*(oldx+newx%)
  930 c=oldy-oldx*(a*oldx+b)
  940 incrx=-difx%/div%
  950 D=(TIME-T)/5
  960 PROCplonk
  970 T=TIME
  980 FORi%=1TOdiv%
  990 oldx=oldx+incrx
 1000 oldy=a*oldx*oldx + b*oldx +c
 1010 PROCblot(oldx,oldy)
 1020 NEXT
 1030
 1040 UNTILINKEY(1)>0
 1050 VDU4:MODE7
 1060 END
 1070
 1080 DEF PROCblot(x,y)
 1090 LOCAL dx%,dy%
 1100 dx%=x:dy%=y+64
 1110 IF notfirst% THEN MOVE vx%,vy%:GC
OL2,2:VDU224,8,10,225
 1120 MOVE dx%,dy%
 1130 GCOL 1,1:VDU224,8,10,225
 1140 vx%=dx%:vy%=dy%
 1150 notfirst%=TRUE
 1160 ENDPROC
 1170
 1180 DEF PROCrhomb(a%,b%)
 1190 LOCAL corner%,p1%,p2%
 1200 corner%=0
 1210 p1%=1:p2%=81
 1220 centres%(0,centindex%)=x%+halfx%
 1230 centres%(1,centindex%)=y%+halfy%
 1240 REPEAT
 1250 PROCdrop
 1260 PLOT p1%,asinth,acosth
 1270 PROCdrop
 1280 PLOT p2%,asinfi,-acosfi
 1290 PROCdrop
 1300 PLOT p1%,-asinth,-acosth
 1310 PROCdrop:IF a%=-1 THEN PROCblack(
asinth,0)
 1320 PLOT p2%,-asinfi,acosfi
 1330 IF a%=0 THEN PROCblack(0,-acosfi)
 1340 p1%=p1%+2:p2%=3
 1350 UNTIL p1%>3
 1360 centindex%=centindex%+1
 1370 ENDPROC
 1380
 1390 DEF PROCdrop
 1400 IF p1%<>3 THEN 1430
 1410 corner%=corner%+1
 1420 IF (a%=corner%)OR(b%=corner%) THE
N PLOT 1,0,-h%:PLOT 0,0,h%
 1430 ENDPROC
 1440
 1450 DEF PROCblack(xd,yd)
 1460 IF p1%<>3 THEN 1480
 1470 PLOT 83,xd,yd:PLOT 0,-xd,-yd
 1480 ENDPROC
 1490
 1500 DEF PROCplonk
 1510 IF ABS(P0%-P3%)>19 THEN TMP%=P3%:
P3%=P2%:P2%=P1%:P1%=TMP%-48
 1520 P1%=P1%+4:P2%=P2%+4:P3%=P3%+4
 1530  A1%=FNAMP(P1%):A2%=FNAMP(P2%):A3
%=FNAMP(P3%)
 1540 ENVELOPE1,1,0,0,0,0,0,0,0,127,-10,-
1,-127,A1%,A1%
 1550 ENVELOPE2,1,0,0,0,0,0,0,0,127,-10,-
1,-127,A2%,A2%
 1560 ENVELOPE3,1,0,0,0,0,0,0,0,127,-10,-
1,-127,A3%,A3%
 1570 SOUND &201,1,P1%,D
 1580 SOUND &202,2,P2%,D
 1590 SOUND &203,3,P3%,D
 1600 ENDPROC
 1610
 1620 DEF FNAMP(PITCH%)
 1630 LOCAL exp
 1640 exp=(ABS(P0%-PITCH%-2)/25)
 1650 =126*(1-exp*exp)
 1660 END
 1670 ON ERROR OFF
 1680 MODE 7
 1690 IF ERR<>17 REPORT:PRINT" at line
";ERL
 1700 END
```

Tested on O.S. 0.1 and 1.2 and on Basics I and II

MUNCH-MAN (32k)

by Andrew Hynes



Munch-Man is a very well written version of the popular arcade game PAC-MAN. The game, although written in Basic, is quite fast and enjoyable. You take the part of a little yellow 'mouth' scurrying around a maze eating dots to score points. There are three ghosts that chase you and try to kill you. These ghosts move through walls, thus making it harder for you. To combat these ghosts you can eat a power pill, situated in the 4 corners of the maze, and these cause the ghosts to turn blue and run away. While the ghosts are blue you can eat them to gain massive bonus points.

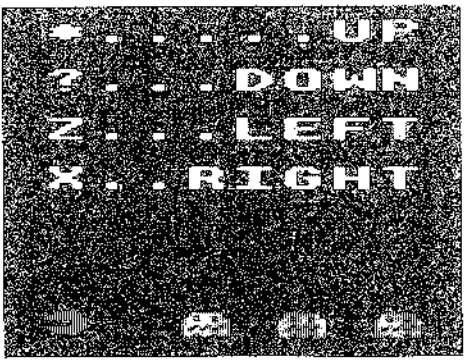
Once you have cleared a sheet, the dots are replaced and you carry on, until you lose three lives. Happy eating!

The best strategy for this game is to make use of the power pills, and the tunnels - because in this implementation the ghosts do not chase you through the tunnels.

The program is in two parts. The first is called "Munch" and sets up the user-defined graphics for use in the main program. It then CHAINs the main program which is called "Munch2". So type in the first program and save it as "Munch", then type in the main program and save it as "Munch2". To load Munch-Man, CHAIN in "Munch", and this should run and auto-load "Munch2"

If you are using a disc system, you will need to use the move-down routine given below to relocate the main program at &E00. Simply add these following lines to the start of "Munch2":

```
0 REM///Move down for Munch-Man
1 IF PAGE=&E00 GOTO 10
2 *KEY0 *T.|MFORI%=0 TO TOP-PAGE STEP4:
I%!&E00=I%!&1900:
  N.|MPAGE=&E00|MOLD|MG.10|M
3 *FX138,0,128
4 END
```




```
  10 REM Munch (Munch-Man Header)
  20 REM Version 1A
  30 REM by Andrew Hynes
  40
  50 VDU23,224,102,102,102,102,102,102
,102,102
  60 VDU23,225,0,255,255,0,0,255,255,0
  70 VDU23,226,0,127,127,96,96,103,103
,102
  80 VDU23,227,0,254,254,6,6,230,230,1
02
  90 VDU23,245,0,0,0,255,255,0,0,0
 100 VDU23,246,60,126,255,255,255,255,
126,60
 110 VDU23,228,102,103,103,96,96,127,1
27,0
 120 VDU23,229,102,230,230,6,6,254,254
,0
 130 VDU23,230,102,102,102,102,102,126
,126,0
 140 VDU23,231,0,126,126,102,102,102,1
02,102
```

```
  150 VDU23,232,0,254,254,6,6,254,254,0
  160 VDU23,233,0,127,127,96,96,127,127
,0
  170 VDU23,234,102,230,230,6,6,230,230
,102
  180 VDU23,235,102,231,231,0,0,255,255
,0
  190 VDU23,236,102,103,103,96,96,103,1
03,102
  200 VDU23,237,60,126,255,224,224,255,
126,60
  210 VDU23,238,60,126,255,7,7,255,126,
60
  220 VDU23,239,36,102,231,231,231,255,
126,60
  230 VDU23,240,60,126,255,231,231,231,
102,36
  240 VDU23,241,0,0,42,0,34,0,42,0
  250 VDU23,242,8,73,42,0,99,0,42,73
  260 VDU23,243,126,90,219,255,213,171,
255,219
  270 VDU23,244,198,56,108,222,190,222,
108,56
  280
  290 CLEAR:CHAIN "Munch2"
```

```
   10 REM Munch2
   20 REM Version 1A
   30 REM By Andrew Hynes
   40 *TV255
   50 *FX 11,1
   60 *FX 12,1
   70 ON ERROR GOTO 2120
   80 MODE2:VDU23;8202;0;0;0;
   90 ENVELOPE 1,1,4,-4,4,10,20,10,127,
0,0,-5,126,126
  100 H%=0:DIM maze$(19,24),ghost(2,2)
  110 REPEAT
  120 PRINTTAB(5,2)"MUNCHMAN"TAB(6,9)"*
.....UP"TAB(6,11)"?...DOWN"TAB(6,13)"Z.
..LEFT"TAB(6,15)"X..RIGHT"
  130 VDU17,3,31,6,21,238,17,6,32,32,24
3,32,243,32,243:PROCspace
  140 lives%=3:score%=0:sheet%=0:ghost=
0:dir$=" "
  150 PROCsetup_maze:TIME=0:fruit=0:EAT
=0
  160 COLOUR7:PRINTTAB(1,0)"Score"TAB(1
4)"High"
  170 PROCprint_score(0):PRINTTAB(19-LE
N(STR$(H%)),1);H%:COLOUR3:PRINTTAB(1,29
)STRING$(lives%-1,CHR$237+" ")" "
  180 REPEAT:*FX 15,0
  190 ghost=ghost+1:IF ghost=3:ghost=0
  200 PROCpacman:PROCcheck
  210 IF EAT=1:SOUND1,-14,150,1
  220 IFTIME>2000ANDfruit=0ANDtot%<150:
PROCfruit
  230 IFRND(1)>.9:PROCdoor
  240 IFfruit=1ANDTIME>3000/sheet%:PROC
no_fruit
  250 IFTIME>1500/sheet% ANDEAT=1:EAT=0
  260 UNTIL tot%=187 OR KILL=1
  270 IF tot%=187:GOTO 150
  280 PROCkilled:IF lives%>0:GOTO 170
  290 PROCend:UNTIL FALSE
  300
  310 DEFFNcalcX(X)
  320 IF X>19:X=0
  330 IF X<0:X=19
  340 =X
  350
  360 DEFFNcalcY(Y)
  370 IF Y>24:Y=0
  380 IF Y<0:Y=24
  390 =Y
  400
  410 DEFPROCtime(T)
  420 TM=TIME:REPEAT UNTIL TIME>TM+T
  430 ENDPROC
  440
  450 DEFPROCprint_score(S%)
  460 score%=score%+S%
  470 COLOUR7:PRINTTAB(1,1);score%
  480 ENDPROC
  490
  500 DEFPROCpacman
  510 OX%=pacX%:OY%=pacY%:SC=score%:IF
pac$<>CHR$246:OP$=pac$
  520 dir$=INKEY$(0)
  530 IF dir$=":" pac$=CHR$239:pacY%=pa
cY%-1
  540 IF dir$="/" pac$=CHR$240:pacY%=pa
cY%+1
  550 IF dir$="Z" pac$=CHR$238:pacX%=pa
cX%-1
  560 IF dir$="X" pac$=CHR$237:pacX%=pa
cX%+1
  570 pacY%=FNcalcY(pacY%):pacX%=FNcalc
X(pacX%)
  580 IF dir$="" AND pac$=CHR$246 THEN
 pac$=OP$ ELSE IF dir$="" pac$=CHR$246
  590 A$=maze$(pacX%,pacY%)
  600 IF A$=".":PROCprint_score(2):SOUN
D1,1,60,1:tot%=tot%+1
  610 IF A$=CHR$244:PROCprint_score(100
):PROCno_fruit:SOUND1,1,150,2
  620 IF A$="+":PROCprint_score(50):EAT
=1:TIME=0:SOUND1,1,100,2
  630 IF SC=score% AND A$<>" ":pacX%=OX
%:pacY%=OY%
  640 VDU17,3,31,OX%,OY%+3,32
  650 PRINTTAB(pacX%,pacY%+3);pac$
  660 maze$(pacX%,pacY%)=" "
  670 ENDPROC
  680
  690 DEFPROCrub_out_ghost(G%):COLOUR4
  700 M$=maze$(ghost(G%,1),ghost(G%,2))
  710 IF M$="." OR M$="+":COLOUR2
  720 IF M$=CHR$244:COLOUR1
  730 IF M$=CHR$245:COLOUR5
```

```
  740 PRINTTAB(ghost(G%,1),ghost(G%,2)+
3)maze$(ghost(G%,1),ghost(G%,2))
  750 ENDPROC
  760
  770 DEFPROCghost(G%):C=0
  780 PROCrub_out_ghost(G%)
  790 IF EAT=1:COLOUR4 ELSE COLOURghost
(G%,0)
  800 IFEAT=0:DX%=1:DY%=1 ELSE DX%=-1:D
Y%=-1
  810 IFG%=0:PROCvert:IFC=0:PROChoriz
  820 IFG%=1:PROChoriz:IFC=0:PROCvert
  830 IFG%=2:PROChoriz:PROCvert
  840 PRINTTAB(ghost(G%,1),ghost(G%,2)+
3)CHR$243
  850 ENDPROC
  860
  870 DEFPROCvert
  880 A=FNcalcY(ghost(G%,2)+DY%)
  890 B=FNcalcY(ghost(G%,2)-DY%)
  900 IFpacY%>ghost(G%,2):ghost(G%,2)=A
:C=1
  910 IFpacY%<ghost(G%,2):ghost(G%,2)=B
:C=1
  920 ENDPROC
  930
  940 DEFPROChoriz
  950 A=FNcalcX(ghost(G%,1)+DX%)
  960 B=FNcalcX(ghost(G%,1)-DX%)
  970 IFpacX%>ghost(G%,1):ghost(G%,1)=A
:C=1
  980 IFpacX%<ghost(G%,1):ghost(G%,1)=B
:C=1
  990 ENDPROC
 1000
 1010 DEFPROCdoor
 1020 D%=RND(4):RESTORE1850
 1030 FORF%=1 TO D%:READX,Y:NEXT
 1040 IF X=pacX% AND Y=pacY%:GOTO 1020
 1050 IF maze$(X,Y)=" ":maze$(X,Y)=CHR$
245 ELSE maze$(X,Y)=" "
 1060 VDU17,5,31,X,Y+3,ASC(maze$(X,Y))
 1070 ENDPROC
 1080
 1090 DEFPROCfruit
 1100 VDU17,1,31,9,18,244
 1110 fruit=1:TIME=0
 1120 maze$(9,15)=CHR$244
 1130 ENDPROC
 1140
 1150 DEFPROCno_fruit
 1160 VDU31,9,18,32:fruit=0
 1170 maze$(9,15)=" "
 1180 ENDPROC
 1190
 1200 DEFPROCcheck:KILL=0
 1210 FOR G%=0 TO 2
 1220 IF G%=ghost:PROCghost(G%)
 1230 IF ghost(G%,1)<>pacX% OR ghost(G%
,2)<>pacY%:GOTO 1250
 1240 IF EAT=1:PROCprint_score(250):SOU
ND1,1,250,1:PROCsetup_ghosts(G%) ELSE K
ILL=1
 1250 NEXT
 1260 ENDPROC
 1270
 1280 DEFPROCkilled:COLOUR3
 1290 FOR C=237 TO 242
 1300 VDU31,pacX%,pacY%+3,C
 1310 SOUND1,1,C/2,1:PROCtime(20)
 1320 NEXT
 1330 VDU31,pacX%,pacY%+3,32
 1340 FOR G%=0 TO 2
 1350 PROCrub_out_ghost(G%)
 1360 PROCsetup_ghosts(G%)
 1370 NEXT
 1380 PROCpac:lives%=lives%-1
 1390 ENDPROC
 1400
 1410 DEFPROCend
 1420 IF score%>H%:H%=score%
 1430 COLOUR7:PRINTTAB(19-LEN(STR$(H%))
,1);H%
 1440 PROCspace
 1450 ENDPROC
 1460
 1470 DEFPROCspace:*FX 15,0
 1480 VDU17,15,31,1,30:PRINT"SPACE BAR
TO START"
 1490 REPEAT UNTIL GET=32:VDU12,17,7
 1500 ENDPROC
 1510
 1520 DEFPROCsetup_ghosts(G%)
 1530 RESTORE 1840
 1540 FOR D%=0 TO G%:READ COL,X:NEXT
 1550 ghost(G%,1)=X
 1560 ghost(G%,2)=12
 1570 ghost(G%,0)=COL
 1580 ENDPROC
 1590
 1600 DEFPROCpac
 1610 pacX%=9:pacY%=20:pac$=CHR$237
 1620 ENDPROC
 1630
 1640 DEFPROCsetup_maze
 1650 FORG%=0TO2:PROCsetup_ghosts(G%):N
EXT
 1660 PROCpac:tot%=0:sheet%=sheet%+1
 1670 COLOUR128:RESTORE 1860
 1680 FOR D%=3 TO 27
 1690 READ maze$
 1700 FOR E%=0 TO 19
 1710 A$=MID$(maze$,E%+1,1)
 1720 IF A$>="A" AND A$<="M":M$=CHR$(AS
C(A$)+159):COLOUR4 ELSE M$=A$:COLOUR2
 1730 PRINTTAB(E%,D%)M$
 1740 maze$(E%,D%-3)=M$
 1750 NEXT
 1760 NEXT
 1770 VDU19,2,9,0,0,0,19,4,14,0,0,0
```

```
1780 FOR D%=1 TO 21
1790 SOUND 2,-15,D%*2,1
1800 SOUND 1,-15,D%*3,1
1810 NEXT:VDU20
1820 ENDPROC
1830
1840 DATA 1,1,5,9,6,18
1850 DATA 9,1,9,22,2,15,17,15
1860 DATA"          A A          "
1870 DATA" CBBBBBBL LBBBBBBBD "
1880 DATA" A+.............+A "
1890 DATA" A.CI.JBD.CBI.JBD.A "
1900 DATA" A.G....A.A.....G.A "
1910 DATA" A...CI.G.G.JBD...A "
1920 DATA" A.H.G........A.H.A "
1930 DATA" A.G...CBD.CD.A.A.A "
1940 DATA" A...H.EBF.EF.G.A.A "
1950 DATA" A.JBK..........G.A "
1960 DATA" A...G.CBBBD.JD...A "
1970 DATA"BLBI...A    A..MBBBLB"
1980 DATA"    ...JBK    MI.G...  "
1990 DATA"BD.H...A    A....H.CB"
2000 DATA" A.MBI.EBBBF.CI.A.A "
2010 DATA" M K..... ...A..M K "
2020 DATA" A.EI.JBI.H.CK.JF.A "
2030 DATA" A........A.EF...A "
2040 DATA" MBBI.H.H.A...JBBK "
2050 DATA" A....A.G.EBBI....A "
2060 DATA" A.CD.A.. .....CD.A "
2070 DATA" A.EF.G.H.H.JBBLF.A "
2080 DATA" A+.....M K.....+A "
2090 DATA" EBBBBBBK MBBBBBBBF "
2100 DATA"          A A          "
2110
2120 ON ERROR OFF
2130 MODE 7: *FX12
2140 REPORT:PRINT" at line ";ERL
2150 END
```

BRAIN TEASER

by Gareth Sugger

REVERSI COUNTING

Many readers will be familiar with the game of Reversi (or Othello). It is played on an 8 x 8 board, of which initially the central four squares are occupied by two counters of each colour (black and white). Each move consists of a player placing a counter on the board adjacent to a piece of the opposite colour, trapping at least one line of opposing counters between the new just placed and another of the same colour. All such lines are then 'reversed' in colour. If no such move is available, play passes to the opponent. If neither player can move the game is over.

In programming this game I wanted to set up an array which would, at each stage of the game, store a list of the moves available to the player. To save memory I wanted to keep the dimension of this array as small as possible. How large does it need to be ? In other words what is the largest number of moves that can be available to a player ? Another question connected with this game is "What is the shortest possible game ?"...

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

SIMPLIFYING CHARACTER DEFINITIONS - Alan Baker

When using VDU 23 it is much easier to use hex numbers as this splits the problem up into adding up 4 bit nibbles rather than 8 bit decimal numbers. eg.

```
VDU23, 224,60,126,219,255,126,60,36,66
```

can be replaced by:

```
VDU23,224,&3C,&7E,&DB,&FF,&7E,&3C,&24,&42
```

This can then be shortened to:

```
VDU23,224,&7E3C;&FFDB;&3C7E;&4224;
```

With a saving of three bytes.

ABBREVIATION FOR COLOUR - N.Sharma

N.Sharma reminds us that the abbreviation for colour is C. For a full list of abbreviations see the user guide, page 484 or the BEEBUG reference card page 4.

IF YOU WRITE TO US

BACK ISSUES (Members only)

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

NOTE OUR NEW SUBSCRIPTIONS ADDRESS

Subscriptions & Software Address
BEEBUG
PO BOX 109
Baker Street
High Wycombe
Bucks
HP11 2TD

SUBSCRIPTIONS

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

MEMBERSHIP COSTS:

£5.40 for 6 months (5 issues)
£9.90 for 1 year (10 issues)
European Membership £16 for 1 year.
Elsewhere (Postal zones)
Zone A £19, Zone B £21, Zone C £23

SOFTWARE AND ROM OFFER (Members only)

These are available from the subscription address, which is our NEW software address also. (Note that this does not apply to Wordwise - in this instance please see magazine for details).

IDEAS, HINTS & TIPS, PROGRAMS, AND LONGER ARTICLES

Substantial articles are particularly welcome and we will pay around £25 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 or disc in machine readable form using "Wordwise", "Minitext Editor" or other means. If you use cassette, please include a backup copy at 300 baud.

Editorial Address
BEEBUG
PO Box 50
St Albans
Herts
AL1 2AR

We will also pay £10 for the best Hint or Tip that we publish, and £5 to the next best. 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.

BEEBUG MAGAZINE is produced by Sheridan Williams and Dr David Graham.
This issue was edited by David Graham and Mike Williams.
Production Editor: Phyllida Vanstone. Technical Assistant: Alan Webster.
Thanks are due to John Yale, Adrian Calcraft, Tim Powys-Lybbe, Graham Greatrix, Matthew Rapier, Colin Opie and David Earlam for assistance with this issue.

All reasonable precautions are taken by BEEBUG to ensure that the advice and data given to readers are reliable. We cannot, however, guarantee it, and we cannot accept legal responsibility for it, neither can we guarantee the products reviewed or advertised. BEEBUG (c) October 1983.

BEEBUG NEW ROM OFFER

A special arrangement has been agreed between Acorn and BEEBUG whereby BEEBUG members may obtain the Series One Machine Operating System in ROM at the price of £5.85 including VAT and post and packing.

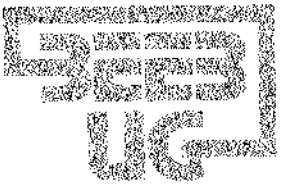
The ROM will be supplied with fitting instructions to enable members to install it in their machine.

If the computer does not subsequently operate correctly, members may take their machines to an Acorn dealer for the upgrade to be tested, which will be done at a charge of £6.00 plus VAT. This charge will be waived if the ROM is found to have been defective. If the computer has been damaged during the installation process, the dealer will make a repair charge.

Please note that we cannot accept EPROM-based operating systems (0.1 or 1.0) in lieu of payment. This can only be performed by Acorn dealers or by Acorn's service centre at Feltham, and applies only to users of the 0.1 O.S.

ADDRESS FOR 1.2 O.S. IF ORDERED ON ITS OWN:- ROM Offer, BEEBUG, PO Box 109, High Wycombe, Bucks, HP11 2TD. PLEASE ALLOW 28 DAYS FOR DELIVERY.

SPECIAL OFFERS FROM



THIS MONTH

WORDWISE Wordprocessor £38

EXMON Machine Code Monitor Cassette £6.90 EPROM £14.50

MASTERFILE File Management Program on Cassette £6.75
on Disc £11.00 and £12.00

OMEGA New Machine code game

SWARMER New Machine code game

NEW 1.2 OPERATING SYSTEM ROM £5.85

MAGAZINE CASSETTE SUBSCRIPTION £33 P.A.

BEEBUG MAGAZINE BINDERS £3.90

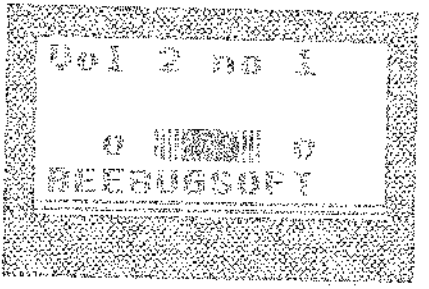
MEMOREX BLANK 5.25 INCH DISCS
40 TRACK £15.50 80 TRACK £25.50

NEW SOFTWARE CLUB FOR BEEBUG MEMBERS
Games from Program Power, Computer Concepts, and Virgin Games at a discount to members.

FOR FULL DETAILS ON THESE OFFERS AND THE COMPLETE BEEBUGSOFT SOFTWARE LIBRARY, SEE THE ADVERTISING SUPPLEMENT.

MAGAZINE CASSETTE OFFER

To save wear and tear on fingers and brain, we will be offering each month a cassette of the programs featured in the latest edition of BEEBUG. The first program on each tape is a menu program, detailing the tape's contents, and allowing the selection of individual programs. The tapes are produced to a high technical standard by the process used for the BEEBUGSOFT range of titles. Ordering information, and details of currently available cassettes are given below.



Previous cassettes: Vol.1 No.10, Vol.2 No.1, Vol.2 No.2, Vol.2 No.3, Vol.2 No.4.

This month's cassette (Vol.2 No.5) includes: Disc string search; Selective renumber; 3D surface; Speech count; Munchyman; Stairs; Fabrics; Invisible alarm; Teletext programs; Joystick initialisation; Prelude XII by Bach and an NEC printer dump.

For ordering information see BEEBUGSOFT advertisement at the back of this month's magazine supplement.

MAGAZINE CASSETTE SUBSCRIPTION

We are able to offer members subscription to our magazine cassettes. Subscriptions will be for a period of one year and are for ten consecutive issues of the cassette. If required, subscriptions may be backdated as far as Vol.1 No.10, which was the first issue available on cassette. This offer is available to members only, so when applying for subscription please write to the address below, quoting your membership number and the issue from which you would like your subscription to start.

CASSETTE SUBSCRIPTION ADDRESS:

Please send a sterling cheque with order, together with your membership number and the date from which the subscription is to run, to:
PO Box 109, Baker Street, High Wycombe, Bucks, HP11 2TD.

CASSETTE SUBSCRIPTION PRICE:
UK £33 inc VAT and p&p
OVERSEAS (inc Eire) £39 inc p&p (no VAT payable).

BEEBUG BINDER OFFER

BEEBUG MAGAZINE BINDER OFFER

A hard-backed binder for BEEBUG magazine is now available. These binders are dark blue in colour with 'BEEBUG' in gold lettering on the spine. They allow you to use the whole of the first volume of the magazine as a single reference book. Individual issues may be easily added or removed. The binders will also conveniently hold less than 10 issues, so that you can use it while you build up Volume 2 also.

BINDER PRICE
U.K. £3.90 inc p&p, and VAT.
Europe £4.90 inc p&p (VAT not charged)
Elsewhere £5.90 inc p&p (VAT not charged)

Make cheques payable to BEEBUG. Send to Binder Offer, BEEBUG, PO Box 109, Baker Street, High Wycombe, Bucks, HP11 2TD. Please allow 28 days for delivery on U.K. orders.

Please note that if you are ordering a binder now, you will not receive your binder until mid-August, we apologise for this, but have had to re-order.

Printed in England by Staples Printers St Albans Limited at The Priory Press.
ISSN 0263-7561