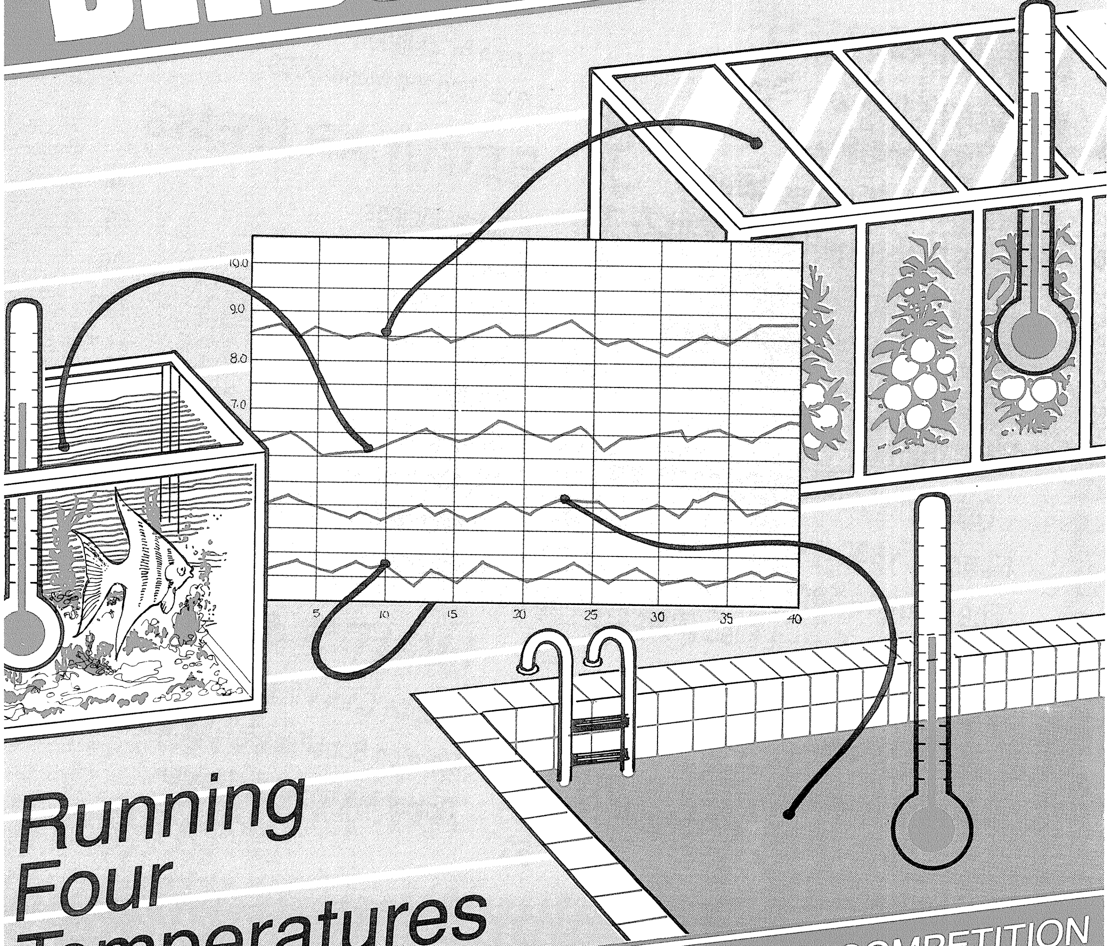
Vol.8 No.3 July 1989

BEEBUG

FOR THE BBC MICRO & MASTER SERIES



Running Four Temperatures

● MATHEMATICAL TRANSFORMATIONS ● A3000 COMPETITION

● EXPANDING MUSIC 5000 ● ADFS DISC SECTOR EDITOR

BEEBUG

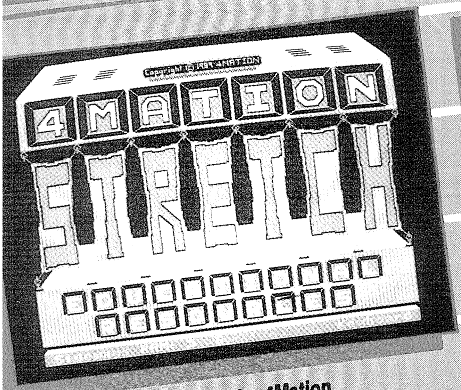
Vol.8 No.3 July 1989

FEATURES

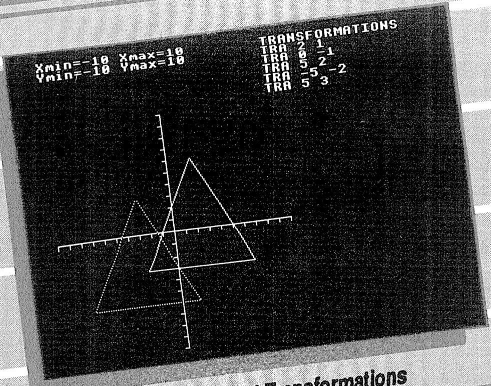
PROGRAM INFORMATION

All listings published in BEEBUG magazine are produced directly from working programs. They are formatted using LISTO 1 and WIDTH 40. The space following the line number is to aid readability only, and may be omitted when the program is typed in. However, the rest of each line should be entered exactly as printed, and checked carefully. When entering a listing, pay special attention to the difference between the digit one and a lower case l (L). Also note that the vertical bar character (Shift \) is reproduced in listings as ¦.

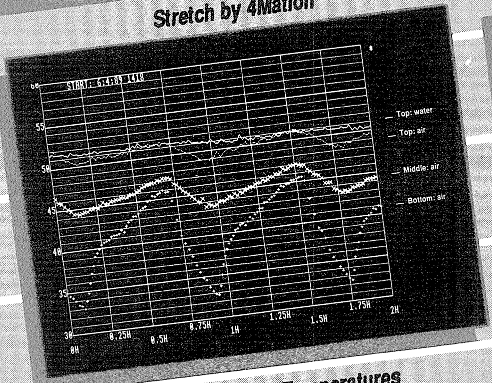
All programs in BEEBUG magazine will run on any BBC micro with Basic II or later, unless otherwise indicated. Members with Basic I are referred to the article on page 44 of BEEBUG Vol.7 No.2 (reprints



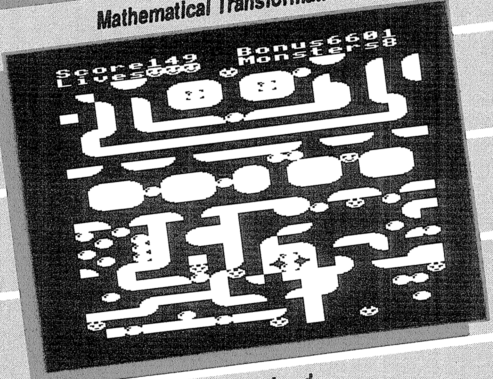
Stretch by 4Mation



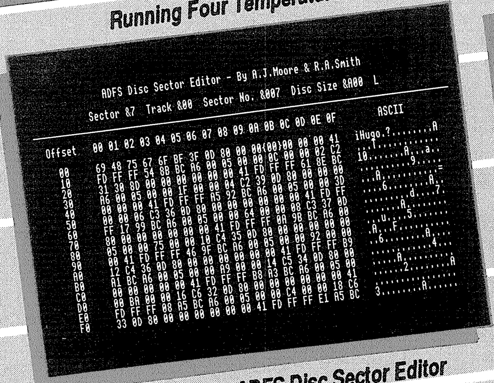
Mathematical Transformations



Running Four Temperatures



Egghead

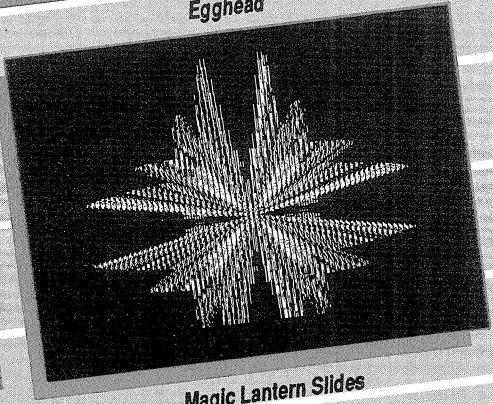


```
ADFS Disc Sector Editor - By A.J.Moore & R.A.Smith
Sector &7  Track &00  Sector No. &007  Disc Size &A00  L

Offset  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F     ASCII

  00    69 48 75 67 6F BF 3F 0D 80 00 00{00}00 00 00 41  iHugo.?........A
  10    FD FF FF 54 8B BC A6 00 05 00 00 0C 00 00 02 C2  ...T............
  20    31 30 8D 00 00 00 00 00 00 41 FD FF FF 61 8E BC  10.......A...a..
  30    A6 00 05 00 00 1F 00 00 04 C2 39 0D 80 00 00 00  ..........9.....
  40    00 00 00 41 FD FF FF A5 92 BC A6 00 05 00 00 3D  ...A...........=
  50    00 00 06 C3 36 0D 80 00 00 00 00 00 00 41 FD FF  ....6........A..
  60    FF 17 99 BC A6 00 05 00 00 64 00 00 08 C3 37 0D  .........d....7.
  70    80 00 00 00 00 00 00 41 FD FF FF 0A 9B BC A6 00  .......A........
  80    05 00 00 75 00 00 10 C4 35 0D 80 00 00 00 00 00  ...u....5.......
  90    00 41 FD FF FF 46 9F BC A6 00 05 00 00 92 00 00  .A...F..........
  A0    12 C4 36 0D 80 00 00 00 00 00 00 41 FD FF FF B9  ..6........A....
  B0    A1 BC A6 00 05 00 00 A9 00 00 14 C5 34 0D 80 00  ............4...
  C0    00 00 00 00 00 41 FD FF FF B8 A3 BC A6 00 05 00  .....A..........
  D0    00 BA 00 00 16 C6 32 0D 80 00 00 00 00 00 00 41  ......2........A
  E0    FD FF FF 08 A5 BC A6 00 05 00 00 C4 00 00 18 C6  ................
  F0    33 0D 80 00 00 00 00 00 00 41 FD FF FF E1 A5 BC  3........A......
```

An ADFS Disc Sector Editor



Magic Lantern Slides

available on receipt of an A5 SAE), and are strongly advised to upgrade to Basic II. Any second processor fitted to the computer should be turned off before the programs are run.

Where a program requires a certain configuration, this is indicated by symbols at the beginning of the article (as shown opposite). Any other requirements are referred to explicitly in the text of the article.

Program will not function on a cassette-based system.

Program needs at least one bank of sideways RAM.

Program is for Master 128 and Compact only.

Editor's Jottings

SUMMER COMPETITION

Elsewhere in this issue of BEEBUG you will find full details of our great Summer Competition. This is open equally to both RISC User and BEEBUG members, and the first prize is a brand new A3000 computer system kindly donated by Acorn and worth nearly £1000.

The competition involves writing a program in Basic to solve the popular mathematical problem often referred to as *Four Fours*. All entrants will be restricted to the subset of BBC Basic that works on a model B, and all programs will eventually be timed and compared on the same machine in fairness to all. The closing date for entries is the 31st August.

READER SURVEY

Despite the appearance of the Archimedes range during the last two years, there are still many tens of thousands of model Bs in regular use, not to mention the Master and Compact, and this is likely to be so for many years to come. However, times are changing and we feel that it is an opportune time for BEEBUG members to tell us the kind of magazine which they would like to see in the future, and a reader survey is included in the centrefold of this issue.

Please spare the time to fill in your answers if you can, and return to us by 4th August. Your efforts in doing this will be very much appreciated.

BEEBUG AT THE BBC ACORN USER SHOW

BEEBUG and RISC User will be on stand 55 at the BBC Acorn User Show from 21st to 23rd July. We will be very pleased to see members at our stand. More details on our latest products are featured elsewhere in this issue. Note the new times and venue for this show.

BEEBUG MOVES HOUSE

BEEBUG will be moving to new and larger premises in St Albans as from August 14th, which should enable us to significantly improve and expand our services to members. The new address is:

117 Hatfield Road, St Albans AL1 4JS.

See the leaflet with this issue for more details.

The next issue of BEEBUG is a two-month issue covering both August and September, and is scheduled to be mailed out at the end of August.

This month's telesoftware password is *cranberry*.

LOOKING AHEAD WITH BEEBUG

Both BEEBUG and RISC User are produced in two-month editions during the summer, the former for August/September, and the latter for July/August. As a result detailed plans for future issues of both magazines have yet to be finalised, but we have listed below some of the items which we expect to appear during the late summer and autumn.

RISC User is the largest circulation magazine devoted entirely to the Acorn Archimedes range. It is available on subscription to all BEEBUG members at a substantially reduced rate (see page 62).

BEEBUG magazine

Applications & Utilities:

- Curve Fitting
- Help ROM
- Using the Watford Digitiser
- Font Designer
- Comprehensive plotting package
- Printer Control Commands
- ROM Filing System
- Using Dichotomous Keys

Reviews:

- Watford Digitiser
- Opus music program
- Art Studio package

RISC User magazine

Features will include:

- The WIMP Manager
- Archimedes Hardware
- RISC OS
- Controlling Sound
- Using Applications Software

and reviews of selected products:

- Atelier from Minerva
- Acorn Desktop Publishing
- Archway from Simtron
- Hard discs
- Pipedream 3
- RISC OS Companion

*The latest details of the contents and distribution of both magazines are contained in the BEEBUG area of Micronet. Just type *BEEBUG# when on-line.*

News News News News News News

NEW A3000 AT SHOW

Acorn is working together with its dealers, and a major finance company, to ensure that people wishing to purchase the new A3000 computer at the BBC Acorn User Show can do so as easily as possible, despite the expectedly high demand. Prospective purchasers can view the machine on the BEEBUG and RISC User stand (stand 55), and then either pay for the computer there and then, or pay a deposit and organise 0% finance by visiting the Mercantile credit stand. Either way, the buyer will receive a voucher which they should then take to an Acorn lorry parked outside the main hall, where it will be exchanged for the actual computer. Acorn are confident that this method will streamline the purchases for both customers and dealers.

The BBC Acorn User Show takes place at Alexandra Palace from Friday 21st July to Sunday 23rd July. The opening times are 3pm - 9pm on the Friday, and 10am - 6pm on the other two days.

ARTISTIC FLAIR

Art Studio from Impact Software is a new art package for the model B, Master and Electron. *Art Studio* can be used in modes four or five, offering two or four colours respectively, and offers a large number of features including zoom, copy, move, square, circle, rubber and painting with a choice of 100 brush shapes and sizes. *Art Studio* costs £9.95 (inc. VAT) on cassette, or £12.95 (inc. VAT) on 5.25" disc. For more details, contact Impact Software, Neepsend House, 1 Percy Street, Sheffield S3 8AU, tel. (0742) 769950.

MORE MAC CONNECTIVITY

Human Computer Interface, the company which specialises in software to integrate the Beeb and the Apple Macintosh, has added two new products to its range. The first of these is a full implementation of BBC Basic for the Mac. This allows a wide range of programs to be transferred directly from the Beeb via a serial link, and run with little of no modifications. The implementation of BBC Basic includes many of the extensions found in Basic V on the Archimedes, and also features a 6502 emulator for machine code, and a BBC operating system emulator which provides many of the most used star commands and OS calls. BBC Basic for the Mac costs £171.35 (inc. VAT).

The other new product, BBC>>Mac, provides a bridge between the Mac and a Beeb. Using the system, the Beeb's filing system appears as a disc icon on the Mac's Desktop, and files can be transferred between the two machines simply by dragging them around. If the Beeb is connected to an Econet system, then the entire network can be accessed via the Mac's Desktop. An additional feature of BBC>>Mac is a print spooler which will print FX80 compatible output from the Beeb on a LaserWriter connected to the Mac. BBC>>Mac costs £109.25 (inc. VAT), with a suitable lead to connect the two computers available for £28.75 (inc. VAT). For more details contact Human Computer Interface Ltd., 25 City Road, Cambridge CB1 1DP, tel. (0223) 314934.

THE JEDI ARE BACK

Return of the Jedi is a new game released by Domark as a follow up to the *Star Wars* and *Empire Strikes Back* titles. As might be expected, the games are modelled on the Star Wars trilogy, with the player taking on the roles of various characters from the films. In *Return off the Jedi* you start by controlling Princess Leia on her speedbike, and try to kill the Imperial Stormtroopers. Having done this, you must manoeuvre Chewbacca through a storm of debris to meet Han Solo. Your final task is to fly the Millenium Falcon and destroy the central reactor. *Return of the Jedi* is available for the model B and Master, and costs £9.99 (inc. VAT) on cassette, or £12.99 (inc. VAT) on disc. For more details, contact Domark on 01-780 2222.

MAKING MUSIC

OPUS is a music composition program produced by the Hertfordshire Advisory Unit for Microtechnology in Education. A graphical display shows the notes as they are entered, and tunes can be played back to hear the effects of any changes. Completed pieces can be saved to disc and subsequently re-loaded for editing or playing. *OPUS* is suitable for the model B, B+ or Master, and costs £17.25 (inc. VAT). For more details, contact The Advisory Unit, Microtechnology in Education, Endymion Road, Hatfield, Herts AL10 8AU, tel. (07072) 65443.

News News News News News News

BEEBUG NEWS

On this page we take a look at some of the exciting new products and developments at BEEBUG which will shortly be available to BEEBUG and RISC User members.

NEW ARCHIMEDES PRODUCTS

BEEBUG has been busy developing a number of new products for the Archimedes. First and foremost is a full-feature **desktop publishing** and **word processing** package. It is fully WIMP based and makes extensive use of 'fancy fonts'. As well as entering and laying out text in many styles and sizes, sprites and draw files can be included in the document. There is also a fast built-in spelling checker.

A perfect companion to this is our new 200 dot-per-inch handheld **scanner** and **interface podule.** This allows photographs and pages of text to be read in as sprites, and then incorporated into documents or used in other packages. The scanner will be available in A6 and A4 widths, with an optional sheet-feeder for the A4 model.

The BEEBUG **Hard Disc Backup** will be of particular interest to all hard disc users. This WIMP driven package allows either full or incremental backup of files, and uses a compression technique to reduce the number of floppies needed. Also included is a multi-file archiver and de-archiver, and a utility to locate files on your hard disc.

All of these products can be seen at the BBC Acorn User Show (see below), and more details are given in the current retail catalogue.

BEEBUG EXPANDS PREMISES AND PRODUCT RANGE

BEEBUG and RISC User are moving to new and larger premises in St. Albans in early August. This long awaited move will enable us to offer a number of new and improved services.

The new showroom is about double the size of the existing one, enabling us to have more equipment on display. It will also allow us to offer Aries and Amstrad PC's for customers who have requirements in this area. We have now employed a full-time engineer and can offer a faster turnaround on repairs. Additionally, in response to requests from members, we now have a training facility, and will shortly be offering competitively priced courses on a variety of subjects.

Please see the enclosed leaflet for more details of our new premises, and an invitation to our open day to see our new facilities and meet the staff.

ORDER AND SUBSCRIPTION PROCESSING

We have recently installed a major networked computer system to handle order processing and subscriptions. This means that whether you phone in with an order or subscription, we will have all your details to hand. Our telesales staff, Julie and Debbie, will key your order straight into the computer, which now enables us to despatch over 80% of orders on the day they are received. On subscription processing, Mandy and Sarah can similarly renew your subscription or change your address etc., instantly over the phone.

The new system also keeps control of our large stock of hardware and software, thereby ensuring that we can re-order items before running out. As a consequence of this new system, your personal membership number is now more important than ever, and should always be quoted over the phone, or in correspondence.

BBC ACORN USER SHOW

The BBC Acorn User show at Alexandra Palace from 21st - 23rd July is the only Acorn-specific show this year. BEEBUG and RISC User will be there on **stand 55**. We will be demonstrating all the new products mentioned above, along with many other items including the new **Acorn A3000** computer which will be on sale at the show (including 0% finance facilities). The magazine and technical staff will be on hand to chat with members and try and help with any problems. Alternatively, why not take the opportunity to meet the faces behind the names seen on the pages of BEEBUG and RISC User.

NEW CATALOGUE

With this issue of BEEBUG you will have received a much larger retail catalogue than normal. This new format catalogue allows us to give more details, and pictures of products. Included in this new catalogue is the full range of Amstrad and Aries PCs which we stock.

WIN AN A3000 COLOUR SYSTEM

IN THE BEEBUG/RISC User SUMMER COMPETITION

Acorn has generously agreed to provide an A3000 colour system (worth over £1000) as the main prize in our Summer Competition.

In 1987 Acorn donated an Archimedes A305 colour system as a prize for our *"Alphametics" programming competition.* The interest and response to this was so impressive that we propose another competition along similar lines, involving the submission of a program which will be tested for speed and versatility. The competition will be open to all members of BEEBUG and RISC User. All programs will be timed on the same machine, thus making it fair for all. All programs must be written in a subset of BBC Basic which will run on any Acorn machine (except the Atom) without the use of any machine call, indirection operators, or operating system calls. This means that Master owners cannot use ON-PROC, while Archimedes owners must also avoid WHILE, CASE, IF-ENDIF, matrix operations, local error handling, RETURN parameters, local arrays, and the operators +=, -=, *=, /=. Programs may, however, use the standard Basic disc filing commands (DFS or ADFS) providing the file size does not exceed 100k.

WHAT YOU HAVE TO DO

Write a program to generate and print an expression using exactly four fours, which will produce the numbers 0 to 50 together with the expressions that generate them. These four fours may be combined in any way, provided that fours are the only digits that appear in the expression. Mathematical symbols are + - * / ^ ( ) . ! R G (R and G represent "Sqr Root" and "Gamma" respectively).

Here are some examples:

```
 0=4+4-4-4         1=44/44
 12=4/.4+4/R4     17=4*(4+R(R4^(-4)))
 20=G4-G4+!4-4     48 =!4+!4+4-4
 49=(R4+R4/.4)^R4
```

Where R means "Square Root", ! means "Factorial, and G means "Gamma".

```
  !n=n*(n-1)*(n-2)*...*1 e.g.   !4=4*3*2*1=24.
Gn=!(n-1) e.g. G4=!(4-1)=!3=6.
```

The examples above do not necessarily give the simplest solution. E.g. 20 could be represented more simply as 4*(4+4/4). Your program must be able to be easily changed to work with numbers other than "fours", for instance it should allow four "threes", or four "fives".

Send a postcard to be received by 31st August 1989 containing your ***Name, Membership Number, Day-time phone number, Time taken for 0-50, and the Machine used.***

Send to **BEEBUG A3000 Competition,**
117 Hatfield Road,
St Albans,
Herts, AL1 4JS

(please note that this is our new address as from 7th August).

If you are shortlisted, we will then ask you to submit your program, when it will be tested back to back against other programs. Do NOT be put off if you have a slow machine or if your program takes all day to provide expressions, or fails to find all solutions (they may not exist!) this will be taken into account in selecting the shortlist.

CRITERIA FOR JUDGING

Entries will be judged on a combination of the criteria listed below, and the winner must be able to supply a version suitably annotated for publication. BEEBUG will be the sole judge as to whether an entry conforms to the spirit of the competition, and sole judge when determining the winner. BEEBUG's decision will be final. No BEEBUG staff or their relatives are eligible to enter.

1. Speed - the program that finds expressions for the numbers 0 to 50 in the shortest time when run on the same machine as all the other entries.

2. Choosing simplest solutions - The program that gives the simplest expression for each number. Priorities for defining simplicity will be the following order + - * / ^ ( ) . R G "tricks", "approximations". For example the following solutions for the number 1 are graded in order of simplicity:

```
1 = 44/44
1 = (4*4)/(4*4)
1 = (4/.4)/(4/.4)
1 = G4/G(4-.4+.4)
1 = TAN(44+4/4) classed as a "trick"
1 = 4^(4/44!)  classed as an "approximation"
```

3. Digit Flexibility - The program should be simple to change to make it work for other digits such as four "threes".

4. Range Flexibility - The program should allow the range to be selected so as to find solutions from 50 - 100 say.

We are unable to undertake any correspondence on the competition, however we can supply a sheet expanding on various aspects upon receipt of an SAE.

Magic Lantern Slides

by Derek Greenacre

Most computer programmers think that moving pictures began with the advent of the cinema, and as if to somehow reinforce this idea their aspirations are often to emulate the the cinema's realism in their own programs.

It is difficult to conceive that the computer can compete on equal terms with the medium of film, for photography can capture every graduation of light and shade, every detail of tone and texture, and this can be faithfully recorded for posterity. This then is the virtue of film, yet ironically this is also its limitation, for its permanence is highly resistant to any variation. The computer however achieves animation in a much more laborious fashion but is able to compete in a different field by being able to vary its output by varying the colours or making use of random functions to generate interest through endless variations on a theme.

Strange as it may seem this situation has happened before in the history of moving pictures for animation did not begin with the invention of photography but had existed for at least two hundred years prior to this. The device which was to pioneer movement in two dimensions was the Magic Lantern which first seems to have been referred to by a Jesuit priest in 1646. Very quickly the Magic Lantern was to explore all aspects of projected images and in particular realise man's age old dream of being able to create moving pictures. All aspects of inferred movement, whether created by sophisticated computer graphics or more crudely with a smoky Magic Lantern, all rely on the phenomena which exists in the human eye known as the *persistence of vision*. The eye for some reason retains images for a split second after the object itself has vanished. Thus any similar object brought into the field of view in an obscured fashion (usually by means of a shutter) is mistaken by the brain as the first object but in a new position.

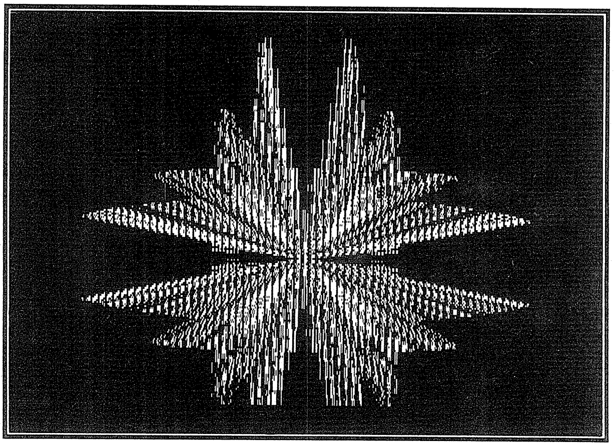


Image produced by program Slide1

It is somewhat of an interesting exercise to look at the comparisons between the Magic Lantern and the computer, for surprisingly there is much that they have in common. Firstly and most obviously they were both in their day at the forefront of technology, but more important was the way each created animation, for here indeed is a very fruitful area of inspiration.

The Magic Lantern, like the computer, was rather laborious in creating movement; as each piece of animation had to be hand drawn onto movable pieces of glass it is little wonder therefore that these early animators thought long and hard about their subject matter before spending hours painting the slide. These kinetic slides when viewed today show the Victorian's mastery of sympathetic subjects for animation. Indeed, the slides themselves seem today to be miniature works of art with their brass bindings and mahogany frames. As the Magic Lantern became more popular in the late Victorian period the moving slides too began to become more sophisticated. One of the most popular I have chosen to emulate here as a computer program. The Victorians were particularly keen to vary images as they were being projected through the use of coloured filters or (similar to the shadow memory on the computer) using a double lantern (called a Biunial) to superimpose one image onto another.

THE SLIDE PROGRAM

In this program the slide being emulated is called a *Chromatrope* in which two patterned pieces of coloured glass rotate by means of rack and pinion to produce a dazzling display on the screen, I have used the sine and cosine functions to achieve the rotation but employed a trial and error approach with the colour variables to achieve washes of colour to simulate the use of the colour filter.

Lines 210 and 220 are optional and are only included to terminate the program and prevent the output becoming muddied. I have tried to keep the program short yet retaining the spirit of the original slide. Changing the following lines to:

```
110 MODE 2
125 D%=RND(10):VDU 29,640;512;
200 IF H%<300 GCOL0,0 ELSE GCOL 0,X%*X%-4*Y%
210 NEXT
220 GOTO 115
230 :
```

produces an interesting variation although it takes a few seconds to start.

I would strongly recommend a closer look at some of the Victorian slides as source material for animation. In fact, I have already seen one commercially produced computer program which duplicates a Victorian slide of the dancing skeleton. A slide which seems to offer great possibilities for computer animation is the *Cycloidoscope* which etched cycloidal patterns into smoked glass and projected them on the screen.

Let's put our animation therefore in its historical context. There is much to re-learn, but somehow I think the past has got a great future ahead of it.

Magic Lantern Slides may be seen in the following museums:

Museum of the Moving Image,
South Bank, London, tel: 01-928 3535.

Science Museum,
South Kensington, London, tel: 01-589 3456.

National Museum of Photography Film and Television,
Princes View, Bradford, tel: (0274) 727488

Further information about Magic Lantern Slides may be obtained from *The Magic Lantern Society of Great Britain, "Prospect", Nutley High Street, East Sussex.*

```
  10 REM Program Slide1
  20 REM Version B1.0
  30 REM Author  Derek Greenacre
  40 REM BEEBUG  July 1989
  50 REM Program subject to copyright
  60 :
 100 ON ERROR:MODE7:PROCerror:END
 110 MODE2:D%=RND(10):VDU 29,640;512;
 120 VDU23,1,0;0;0;0;
 130 FOR H%=40 TO 500
 140 T=H%/40
 150 X%=H%*COS(T)*SIN(T/D%)
 160 Y%=H%*COS(T/D%)
 170 MOVE 0,0
 180 DRAWX%,Y%:MOVE 0,0:DRAW-X%,Y%:MOVE
0,0:DRAWX%,-Y%:MOVE 0,0:DRAW-X%,-Y%
 190 MOVE 0,0:DRAWY%,X%:MOVE 0,0:DRAW-Y
%,X%:MOVE 0,0:DRAWY%,-X%:MOVE 0,0:DRAW-Y
%,-X%
 200 GCOL 0,X%*X%-4*Y%:NEXT
 210 U%=RND(5):IF U%=5 THEN GOTO 240
 220 IF U%<3 CLS
 230 GOTO 110
 240 VDU7:END
 250 :
1000 DEF PROCerror
1010 VDU7:REPORT:PRINT" at line ";ERL
1020 ENDPROC
```

Running Four Temperatures

Ralph Maltby describes how to use your Beeb for monitoring up to four temperature sensors simultaneously.

It is always a pleasure to find a program which has some real value in everyday life. Bernard Hill's ingenious use of low cost components to make a recording thermometer is a case in point (BEEBUG Vol.7 No.4). I found it worthwhile to extend the system to utilise all the four channels available on the analogue port to provide a more flexible means of making a temperature survey. I have certainly found it useful in sorting out a heating problem in my sitting room, monitoring the temperature of seed trays, and finding the right setting for the fridge (see Fig.1).

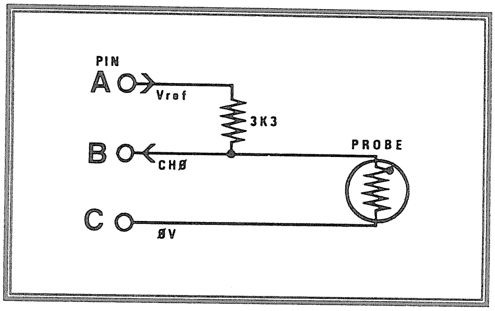


Figure 2. Circuit for a single probe

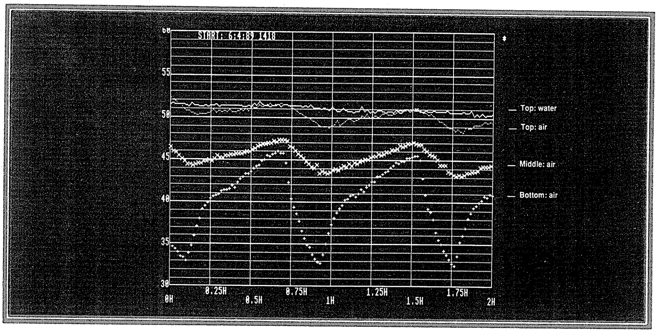


Figure 1. Refrigerator temperatures on different shelves

The components required are exactly as described in the previous article except that four probes and four resistors will be needed. All the resistors can be made to fit within the cover of the case but, if you are as clumsy with a soldering iron as I am, you might prefer to put them in a separate external box. On the D-plug, pins 2, 3, 5, 6, 8 are common to 0v, pins 11, 14 are common to Vref so there is some flexibility in the choice of which pins to use.

With four probes, the wiring diagram can look confusing, so I have tabulated the connections. One of the wires from each probe is connected to one end of its resistor. I have called the other free end of the resistor 'A', the joined end of the resistor 'B' and the free end of the probe 'C' (see Fig.2). A connects to Vref, C goes to 0v and each B goes to its appropriate channel (see Table 1).

Table 2 shows the pins that I used when I put the resistors inside the cover of the plug.

You will first need to calibrate the system, and to do this you should tie the four probes together and then follow the procedure as described in the earlier article (Vol.7 No.4). The new Probe Calibration program is given in Listing 1; it contains a number of modifications from the original.

It starts with a few brief instructions as an aide-memoire and asks how many probes are to be calibrated. It then displays the approximate ADVAL readings so that you will see when they have settled down well enough to take a calibration measurement (the exact values are too "busy"). When they have settled, hold the space bar down for a few seconds while the program averages each of the ADVAL readings. You should then enter the actual temperature. A table is built up showing the calibration temperatures and the recorded ADVAL readings.

When there are enough readings (preferably at least three), pressing Escape will provide the correlation coefficients, and the calibration functions (DEFFNtemp1 etc.) which are appended automatically at the end of the program in lines 32761 to 32764. These lines are also stored in the function key f0 to ease transfer to the Probe Plot program to which they MUST be added before use.

Probe	1	2	3	4
Pin	15	7	12	4

Table 1. Input connections for four probes

The Plot program also starts with a few brief instructions. The number of probes in use is then requested, followed by the screen mode to be used (mode 0 or 1). The temperature ranges, plot period and sample frequency are then entered, just as in the previous program except that the records for each probe are averaged over five readings rather than over the whole sample time (see lines 1960 to 2010).

Probe	1	2	3	4
A	11	11	14	11
B	15	7	12	4
C	6	3	2	2

Table 2. Complete set of pin connections

The next screen displays the temperature at each probe; this gives an opportunity to check that the correct temperature range has been chosen, and to indicate that the readings have settled enough for the plotting to start.

The plot is started at the end of the first cycle after the space bar has been pressed. The plots for each probe need to be identifiable, and this is looked after in lines 2240 and 2280: the Probe1 plot is a continuous line, Probe2 is a dotted line and the Probe3 and Probe4 points are squares and pluses (or small circles on the Master variant). These last two can be joined with lines by inserting PLOT5 commands in lines 2260 and 2270, but I have not found this to be necessary if the sample frequency is chosen carefully. Note that each set of records for the four probes will take about 0.5 second so intervals of less than 1 second should be avoided if all four probes are in use.

Once a plot has been completed a '*' marker appears at the top right corner of the Plot screen to indicate this clearly. At this point, pressing 'S' saves the plot as "SCREEN", and it can be retrieved by *SCREEN. If more than one plot is to be stored in this way, you should rename "SCREEN" before the next one is saved. Pressing 'P' calls a printer dump (see line 300), and this can be used as many times as required. Lastly, pressing the space bar will end the program.

Both programs will operate on the BBC-B and the Master though Basic II is required on the B for the calibration program. However, you can save a few lines in the plotting program on the Master by using some of its extra facilities as follows:

1. Omit lines 1240 to 1280.
2. Insert a line to register date and time:
```
    1465 tim$=TIME$
```
3. Insert a line to strengthen dotted line from PLOT69:
```
    1850 VDU23,6,54;0;0;0;
```
4. Replace lines 2260 to 2270 to give small circles round points for probes 2 & 3:
```
    2260 IF probe%=3 THEN PLOT145,4,0
    2270 IF probe%=4 THEN PLOT145,6,0
```
5. Omit lines 2300 to 2390

If you want to print a dump from a plot retrieved by *SCREEN, the following short program (using BEEBUG's Dumpmaster for example) will do the job:
```
   10 MODE0:REM or MODE1
   20 *SCREEN
   30 VDU2:*DUMP
   40 VDU3:REM
```
Insert the appropriate call in line 30 for your dump program.

NOTE: All the components required can be ordered from Maplin at P.O.Box 3, Rayleigh, Essex or phone (0702) 554161. The parts needed are shown below. You will need four probes and four resistors.

Temperature probe		
-40 to 50 C	£1.95	FP65V
20 to 110 C	£1.95	FP66W
D-plug	£0.45	BK58N
D-plug cover	£0.96	BK60Q
3300 ohm 1/8 watt resistor	£0.03	U3K3

Post and packing extra.

Listing 1

```
   10 REM Program 4 Probe Calibration
   20 REM Version B1.1
   30 REM Author  Ralph Maltby
   40 REM BEEBUG  July 1989
   50 REM Program subject to copyright
   60 REM Developed from a program
   70 REM by Bernard Hill
   80 REM In BEEBUG August 1988
   90 :
  100 MODE7:VDU23,1,0;0;0;0;
  110 Lim=10 :TIME=0 :s$="":flag=0
  120 ON ERROR PROCerror:IF flag=1 THEN
END ELSE IF flag=2 GOTO 170
  130 DIM adval(4,Lim),temp(Lim),g$(4),t
0$(4),line$(4),grecip$(4),S%(4)
  140 PROCtitle("Temperature Calibration
",132,135)
  150 PROCprobes
  160 PROCcalibrate
  170 PROCcalculate:IF flag=1 THEN END
  180 OSCLI("KEY0 "+s$+"|M"):*FX138,0,12
8
  190 END
  200 :
 1000 DEFPROCerror
 1010 IF ERR=17 flag=2:ENDPROC
 1020 REPORT
 1030 PRINT" at line ";ERL:flag=1
 1040 ENDPROC
 1050 :
 1060 DEF PROCtitle (t$,c1,c2)
 1070 LOCAL i,a$
 1080 a$=CHR$c1+CHR$157+CHR$c2
 1090 PRINT a$: FOR i=1 TO 2: PRINT a$;C
HR$141;TAB(18-LENt$/2);t$: NEXT: PRINTa$
 1100 ENDPROC
 1110 :
 1120 DEFPROCprobes
 1130 PRINTTAB(0,5)CHR$134"Instructions:
"
 1140 PRINTCHR$131"The next page display
s current ADVAL"'CHR$131"values. Wait fo
r these to settle.":PRINT
 1150 PRINTCHR$131"Values are averaged w
hile the Space"'CHR$131"Bar is held down
.":PRINT
 1160 PRINTCHR$131"Temperature drift err
ors if held too"'CHR$131"long; scatter e
rrors if too short."'CHR$131"Try about 1
0 secs.":PRINT
 1170 PRINTCHR$131"Press Escape when the
re are enough"'CHR$131"readings on the t
able to complete"'CHR$131"calibration."
 1180 PRINTTAB(0,23)CHR$134"How many pro
bes to calibrate? (1-4)";
 1190 num$=GET$: IF num$<"1" OR num$>"4"
 THEN 1180 ELSE CLS
 1200 num%=VAL(num$)
 1210 ENDPROC
 1220 :
 1230 DEFPROCcalibrate
 1240 PROCtitle("CALIBRATION",132,135)
 1250 M%=1
 1260 REPEAT
 1270 PRINTTAB(0,6)"Hold Space Bar to re
cord Probe Readings"'TAB(9)"or Escape to
 calculate:"
 1280 REPEAT Z=INKEY(30):PRINTTAB(11,3)"
Approximate ADVAL Readings: "
 1290 FOR p%=1 TO num%
 1300 PRINTTAB((3+(7*p%)),4)CHR$134;(ADV
AL(p%)DIV100)*100
 1310 NEXTp% :UNTIL INKEY-99
 1320 PROCmean_adval
 1330 REM Flush Keyboard Buffer
 1340 *FX15 1
 1350 PRINTTAB(0,8)"Enter temperature du
ring this time:";
 1360 INPUT ""temp(M%)
 1370 PROCtable
 1380 PRINTTAB(0,8)CHR$131CHR$157CHR$132
"READY FOR NEXT READING  "CHR$156"
    "
 1390 M%=M%+1
 1400 UNTIL M%=Lim
 1410 ENDPROC
 1420 :
 1430 DEFPROCmean_adval
 1440 LOCAL S%,N%
 1450 N%=0:FOR probe%=1 TO num%: S%(prob
e%)=0:NEXT
 1460 REPEAT
 1470 PRINTTAB(0,8)"Calibrating"+STRING$
(22,".")
 1480 FOR probe%=1 TO num%
 1490 S%(probe%)=S%(probe%)+ADVAL(probe%
)
 1500 NEXT probe%
 1510 N%=N%+1
 1520 UNTIL NOT INKEY-99:REM Until space
 bar off
 1530 PRINTTAB(0,8)"
              "
 1540 FOR probe%=1 TO num%
 1550 adval(probe%,M%)=S%(probe%)/N%
 1560 NEXT probe%
 1570 ENDPROC
 1580 :
 1590 DEFPROCtable
 1600 PRINTTAB(13,10)CHR$131STRING$(7,"-
")CHR$135"ADVAL"CHR$131STRING$(8,"-")
 1610 PRINTTAB(0,11)"Rdng.";TAB(5)"Temp"
;TAB(12)"No.1 ";TAB(19)"No.2";TAB(26)"No
```

```
.3 ";TAB(33)"No.4 "
 1620 PRINTTAB(1,11+M%);M%;TAB(5);temp(M
%)
 1630 FOR probe%=1 TO num%
 1640 PRINTTAB(5+7*probe%,11+M%);INT(adv
al(probe%,M%))
 1650 NEXT probe%
 1660 ENDPROC
 1670  :
 1680 DEFPROCcalculate
 1690 LOCAL i,a,a2,p,t,t2,t0,g,vt,va,vp,
avt,ava
 1700 M%=M%-1
 1710 IF M%<2 PRINTTAB(9,22)"Not enough
points":VDU26,31,0,24:flag=1:ENDPROC
 1720 CLS:PRINT"Calculation based on ";M
%;" points:"''
 1730 FOR probe%= 1 TO num%
 1740 a=0:a2=0:t=0:p=0:t2=0
 1750 FOR i=1 TO M%
 1760 a=a+adval(probe%,i):a2=a2+adval(pr
obe%,i)^2
 1770 t=t+temp(i):p=p+adval(probe%,i)*te
mp(i)
 1780 t2=t2+temp(i)^2
 1790 NEXT
 1800 avt=t/M%:ava=a/M%:vp=p/M%-a*t/M%^2
 1810 va=a2/M%-(a/M%)^2:vt=t2/M%-(t/M%)^
2
 1820 g=vp/va
 1830 t0=avt-g*ava:t0$(probe%)=STR$(t0)
 1840 grecip$(probe%)=STR$(-1/g)
 1850 line$(probe%)=STR$(32760+probe%)
 1860 s$=s$+"|M"+line$(probe%)+" DEFFNte
mp"+STR$(probe%)+" = "+t0$(probe%)+"-ADV
AL("+STR$(probe%)+")/"+grecip$(probe%)
 1870 IF M%>2 THEN PRINT"Correlation("+S
TR$(probe%)") = ";-vp/SQRva/SQRvt''
 1880 NEXT
 1890 ENDPROC
```

Listing 2

```
   10 REM Program 4 Probe Plot
   20 REM Version B1.3(B)
   30 REM Author  Ralph Maltby
   40 REM BEEBUG  July 1989
   50 REM Program subject to copyright
   60 REM From a program by Bernard Hill
   70 REM In BEEBUG August 1988
   80 :
  100 MODE7
  110 top=1199:H=940
  120 ON ERROR MODE7:REPORT:PRINT" at li
ne ";ERL:END
  130 DIM mul%(5),T$(4),t0(4),temp(5)
  140 mul%(1)=1:mul%(2)=60:mul%(3)=3600
  150 mul%(4)=86400:mul%(5)=604800
  160 PROCtitle("Temperature Graphs",132
,131)
  170 PRINTTAB(3,7)"*"CHR$131SPC(3)"appe
ars when graph is complete.":PRINTTAB(7,
9)CHR$131"Then you can press:"
  180 PRINTTAB(2,11)"<P>"CHR$131SPC(2)"t
o dump to printer.":PRINTTAB(7)CHR$131"R
epeat for extra prints."
  190 PRINTTAB(2,14)"<S>"CHR$131SPC(2)"t
o SAVE screen as 'SCREEN'."
  200 PRINTTAB(0,16)"<SPACE>"CHR$131"to
clear screen and exit."
  210 PRINTTAB(2,20)CHR$134"Now enter nu
mber of probes in use";:num$=GET$:IFnum$
<"1" OR num$>"4" THEN 210 ELSE num%=VAL(
num$):CLS
  220 PROCtitle("Set Display Mode",132,1
31)
  230 PRINTTAB(0,10)CHR$131"Mode 0 needs
 a high resolution monitor":PRINTTAB(3)C
HR$131"but it gives a better print."
  240 PRINTTAB(0,18)CHR$134"Enter displa
y mode wanted (0 or 1): ";:mode$=GET$:IF
 mode$<"0" OR mode$>"1" THEN 240 ELSE mo
de%=VAL(mode$):CLS
  250 PROCoptions
  260 MODE(mode%)
  270 PROCaxes
  280 PROCdraw
  290 REPEAT
  300 Z=GET:IF Z=80 OR Z=112 THEN VDU29,
0;0;:REM Call DUMP here, e.g. *DUMP to u
se BEEBUG's Dumpmaster
  310 IF Z=83 OR Z=115 THEN *SAVE SCREEN
 3000 8000
  320 UNTIL Z=32
  330 *FX15,1
  340 MODE7:VDU4
  350 END
  360 :
 1000 DEF PROCtitle(t$,c1,c2)
 1010 LOCAL i,a$:a$=CHR$c1+CHR$157+CHR$c
2
 1020 PRINTa$:FOR i=1 TO 2
 1030 PRINTa$;CHR$141;TAB(19-LENt$/2);t$
 1040 NEXT:PRINTa$
 1050 ENDPROC
 1060 :
 1070 DEF PROCoptions
 1080 LOCAL i,@%:@%=&1020104
 1090 REPEAT
 1105 FOR i=0 TO 1
 1110 PRINTTAB(0,i)CHR$134CHR$141"All pr
obes are set to the same":PRINTTAB(7,i+2
)CHR$134CHR$141"ranges and times."
 1120 NEXT
 1130 PRINTTAB(0,8)"Minimum :";SPC11;"Ma
```

```
ximum :";SPC11;TAB(9,8);
 1140 INPUT""m$:minT=VALm$
 1150 PRINTTAB(29,8);
 1160 INPUT""m$:maxT=VALm$
 1170 UNTIL maxT>minT
 1180 REPEAT
 1190 tspan=FNtime("Time Span:",9)
 1200 U1=U2:U1$=U2$: REM see 1350
 1210 tincr=FNtime("Sample Interval",15)
 1220 UNTIL tspan>=2*tincr AND tincr>=1
 1230 CLS
 1235 REM Omit lines 1240-1280 for Maste
r
 1240 PRINTTAB(1,5)CHR$134"Enter Date an
d/or Time in one line"
 1250 PRINTTAB(7,6)CHR$134"to label top
of graph"
 1260 PRINTTAB(1,8)CHR$134"(Just press R
eturn if not required)"
 1270 INPUTLINE TAB(2,10)tim$
 1280 CLS
 1290 PRINTTAB(0,0):PROCtitle("Current t
emperatures:",132,135)
 1300 PRINTTAB(3,24)CHR$134"Press SPACE
to start recording"
 1310 REPEAT
 1320 FOR probe%=1 TO num%:PRINTTAB(10,5
)"Reading probe ";probe%
 1330 time%=TIME
 1340 tempsum=0:N%=0:T=0:M%=0
 1350 REPEAT
 1360 IF probe%=1 THEN tempsum=tempsum+F
Ntemp1
 1370 IF probe%=2 THEN tempsum=tempsum+F
Ntemp2
 1380 IF probe%=3 THEN tempsum=tempsum+F
Ntemp3
 1390 IF probe%=4 THEN tempsum=tempsum+F
Ntemp4
 1400 N%=N%+1
 1410 UNTIL TIME-time%>100:tempsum=temps
um/N%
 1420 T=T+tempsum:M%=M%+1:T$(probe%)=STR
$tempsum
 1430 PRINTTAB(8*probe%-2,10)T$(probe%)
 1440 t0(probe%)=T/M%
 1450 NEXT probe%
 1460 UNTIL INKEY1<>-1
 1465 REM Insert line for Master
      ie: 1465 tim$=TIME$
 1470 ENDPROC
 1480 :
 1490 DEF FNtime(t$,L)
 1500 LOCAL m$,t
 1510 REPEAT:PRINTTAB(0,L)t$'
 1520 PRINTTAB(0,L+2)"Size";SPC(13)":";S
PC10
 1530 PRINTTAB(0,L+3)"Units (S/M/H/D/W):
";SPC10
 1540 PRINTTAB(20,L+2);:INPUT ""m$
 1550 t=VALm$:PRINTTAB(20,L+3);
 1560 INPUT ""U2$:IF U2$="" THEN U2$="s"
 1570 U2=(INSTR("SsMmHhDdWw",U2$)+1) DIV
2
 1580 t=t*mul%(U2):UNTIL t>=1:=t
 1590 :
 1600 DEF FNy(temp(probe%))=(temp(probe%
)-minT)/(maxT-minT)*H
 1610 DEF FNx(t)=t/tspan*(top+1)
 1620 :
 1630 DEF PROCaxes
 1640 LOCAL @%,t:@%=&10000401
 1650 VDU29,32;72;
 1660 VDU5
 1670 FOR t=minT TO maxT
 1680 FOR t=minT TO maxT:y=FNy(t)
 1690 IF y<0 OR y>H THEN 1730
 1700 IF t MOD 5=0 THEN MOVE -32,y+19:PR
INTt;
 1710 MOVE 0,y
 1720 IF t MOD 5=0 THEN PLOT 5,top,y ELS
E PLOT21,top,y
 1730 NEXT
 1740 line=FNline(tspan):e=FALSE
 1750 FOR t=0 TO tspan STEP line
 1760 x=FNx(t):MOVEx,0:DRAWx,H
 1770 S2=t/mul%(U1)
 1780 IF e THEN MOVE x-16,-8 ELSE MOVE x
-16,-40
 1790 PRINTS2;U1$;:e=NOT e:NEXT
 1800 MOVE 100,938:PRINT"START: ";tim$
 1810 ENDPROC
 1820 :
 1830 DEF PROCdraw
 1840 LOCAL J%,calctime%,t%,tt%
 1850 REM Insert line 1850 for Master
      ie 1850VDU23,6,54;0;0;0;
 1860 TIME=0:J%=0
 1870 FOR probe% =1 TO num%:temp(probe%)
=t0(probe%):NEXT probe%
 1880 REPEAT :time%=TIME:tt%=0
 1890 J%=J%+1
 1900 FOR probe% =1 TO num%
 1910 tempsum=0:N%=0:IF J%=1 THEN calcti
me%=50
 1920 REPEAT UNTIL TIME >= time%+tincr*1
00-calctime%
 1930 t%=TIME
 1940 MOVE FNx(time%/100),FNy(temp(probe
%))
 1950 REPEAT
 1960 IF probe%=1 THEN tempsum=tempsum+F
Ntemp1:N%=N%+1
```

Continued on page 38

Foreign Language Tester (2)

Eddy Hunt shows how to monitor your progress at learning foreign languages, and provides a handy foreign language key display.

In last month's article I described a program for testing foreign vocabulary. An integral part of the system is providing of feedback on progress, and keeping files up-to-date by removing from them words which have been learned.

We will continue with the previous example, working through French vocabulary in a file E.FRVOCA. A record of the progress made during each session is maintained in the file E.FRVOCZ. To check on progress you need to type in and save the program VOCPROG, listed this month. When you run the program, enter the same file name, E.FRVOCA, in answer to the prompt for a file name. A scale will appear at the left of the screen, and the number of words at each point of the scale appears to the right of this.

The meaning of the scale is as follows. As each word is entered into the file it is given a value of zero. Each time it is correctly identified this value is increased by one, and each time it is presented and not correctly recognised, its score is decreased by one. The number of words at each level is displayed to the right. Words with negative scores are 'problem' words, while those with scores of +5 or more are considered 'learnt', and the corresponding numbers appear in green on the screen. You will have had to run VOCTEST at least five times to reach this point.

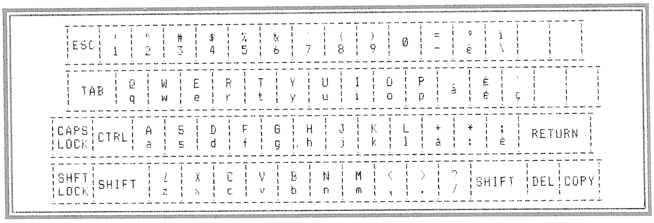
If any such words are present in the file, then you will be offered the chance of removing them from the file. Assuming you press "Y" to update, then the words will be divided between two files:

E.FRVOCA Revision file containing words which have been learnt.
E.FRVOCB Current file containing words which you are learning.

You can still run VOCTEST with the revision file, but no record of progress will be made. For normal testing now use E.FRVOCB. When this file is eventually updated, the current version will become E.FRVOCC, etc (but remember that a final letter of 'Z' is reserved for the progress file).

Clearly, to maintain a stock of words in the current version of the vocabulary file, new words will need to be added at the end. In this way progress is maintained: new words are added and old words are left behind in the revision files. In practice you may wish to keep a record of progress with the revision files as well. This can be achieved by renaming such a file to E.FRREVA (for example) or, more likely, join a number of revision files together using View (or whatever word processor you use).

The method of working is very much down to the individual. Ordinary vocabulary files should be tested at intervals of at least a day but no more than a week. Revision files should be tested at much longer intervals, probably at least a week. Anyone studying a language intensively will probably have several files operating in parallel, collecting all the 'A' revision files into a single renamed file, with similar treatment given to the 'B', 'C' and 'D' revision files, when they are produced.


ESC ! 1 " 2 # 3 $ 4 % 5 & 6 ' 7 (8) 9 0 = - ° è ì \
TAB Q q W w E e R r T t Y y U u I i O o P p à É é ` ç
CAPS LOCK CTRL A a S s D d F f G g H h J j K k L l + à * : ; è RETURN
SHFT LOCK SHIFT Z z X x C c V v B b N n M m < , > . ? / SHIFT DEL COPY

Keyboard layout with French characters

Finally I have included a program for producing a layout diagram of the keyboard using and Epson FX80 printer. All that is required is to run the program to produce the foreign character set, as described in BEEBUG Vol.7 No.4, and then run the program KEYBD.

NOTE: In addition to the two programs listed with this article, the magazine disc also contains character re-definitions for Italian, Portuguese, and the Scandinavian languages.

Listing 1

```
   10 REM Program VocProg
   20 REM Version B1.5
   30 REM Author  Eddy Hunt
   40 REM BEEBUG  July 1989
   50 REM Program subject to copyright
   60 :
  100 MODE7:ON ERROR GOTO 600
  110 DIM ct%(25),st$(250)
  120 ML%=5:ml$=CHR$(77+ML%)
  130 VDU23,1,0;0;0;0;
  140 PRINTTAB(10,2)CHR$131;"Vocabulary
Progress"
  150 PRINT:INPUT"FILE: "fn$
  160 vr$=RIGHT$(fn$,1):fr$=LEFT$(fn$,LE
N(fn$)-1)
  170 C%=OPENUP(fr$+"Z")
  180 IF C%=0 THEN PRINT''"Index file do
es not exist.":END
  190 INPUT#C%,ix$
  200 CLOSE#C%
  210 IW%=ASC(LEFT$(ix$,1))+256*ASC(MID$
(ix$,2,1)):v$=MID$(ix$,3,1)
  220 IF vr$<>v$ PRINT"Current version (
";fr$+v$;") required.":END
  230 ix$=MID$(ix$,4):nw%=LEN(ix$)
  240 PROCtots
  250 h$=CHR$131+"PROGRESS"+STRING$(15,C
HR$32)+"FILE: "+fn$
  260 CLS:PRINT'h$':WL%=0
  270 FOR L%=max% TO min% STEP -1
  280 PRINT FNval(L%),ct%(L%)
  290 IF L%>=12+ML% THEN WL%=WL%+ct%(L%)
  300 NEXT
  310 PRINT'"Words previously learned:";
IW%
  320 PRINT'"Words learned this file:";W
L%''
  330 IF WL%=0 END
  340 IF FNno("Do you wish to update fil
es") THEN END
  350 IF v$="Y" PRINT'"Further updates n
ot possible.":END
  360 PRINT'"UPDATING FILES"
  370 v%=ASC(v$):nv$=CHR$(v%+1)
  380 nf$=fr$+nv$:PROCread
  390 nx$="":ct%(0)=0:ct%(1)=0
  400 FOR RV%=-1 TO 0
  410 IF RV% C%=OPENOUT(fn$) ELSE C%=OPE
NOUT(nf$)
  420 FOR w%=1 TO nw%
  430 x$=MID$(ix$,w%,1)
  440 IF NOT RV% AND x$<ml$ THEN nx$=nx$
+x$:PROCstout
  450 IF RV% AND x$>=ml$ THEN PROCstout
  460 NEXT
  470 BPUT#C%,13
  480 CLOSE#C%
  490 NEXT
  500 IW%=IW%+WL%
  510 PRINT'" Revision file:";fn$;" ";ct
%(1);" words"
  520 PRINT'" Current  file:";nf$;" ";ct
%(0);" words"
  530 nx$=CHR$(IW% MOD 256)+CHR$(IW% DIV
 256)+nv$+nx$
  540 C%=OPENOUT(fr$+"Z")
  550 PRINT#C%,nx$
  560 PRINT'"Files updated."
  570 CLOSE#C%
  580 END
  590 :
  600 MODE7:CLOSE#0
  610 REPORT:PRINT" AT LINE ";ERL
  620 END
  630 :
 1000 DEF PROCtots
 1010 FOR p%=0 TO 25:ct%(p%)=0:NEXT
 1020 max%=0:min%=25
 1030 FOR p%=1 TO LEN(ix$)
 1040 v%=ASC(MID$(ix$,p%,1))-65
 1050 IF v%>25 THEN v%=25
 1060 IF v%<0 THEN v%=0
 1070 ct%(v%)=ct%(v%)+1
 1080 IF v%<min% THEN min%=v%
 1090 IF v%>max% THEN max%=v%
 1100 NEXT
 1110 ENDPROC
 1120 :
 1130 DEF PROCstout
 1140 sv$=st$(w%)
 1150 FOR p%=1 TO LEN(sv$)
 1160 BPUT#C%,ASC(MID$(sv$,p%,1))
 1170 NEXT
 1180 ct%(-RV%)=ct%(-RV%)+1
 1190 ENDPROC
 1200 :
 1210 DEF FNno(m$)
 1220 PRINTm$;"?"
 1230 REPEAT
 1240 G%=GET AND NOT 32
 1250 UNTIL G%=78 OR G%=89
 1260 =(G%=78)
 1270 :
 1280 DEF FNval(n%)
 1290 IF n%<12 =CHR$129+"-"+STR$(12-n%)
 1300 IF n%=12 =CHR$131+" 0"
 1310 IF n%<12+ML% =" +"+STR$(n%-12)
 1320 =CHR$130+"+"+STR$(n%-12)
 1330 :
 1340 DEF PROCread
 1350 C%=OPENUP(fn$)
 1360 IF C%=0 PRINT"File does not exist"
:END
 1370 FOR w%=1 TO nw%
 1380 x$="":oldc%=0
 1390 REPEAT
 1400 x%=BGET#C%
 1410 IF x%<>&1A x$=x$+CHR$(x%)
 1420 flag%=(oldc%=13) AND (x%=13)
 1430 oldc%=x%
 1440 UNTIL flag%
 1450 st$(w%)=x$
 1460 NEXT
 1470 CLOSE#C%
 1480 ENDPROC
```

Listing 2

```
   10 REM Program Keyboard
   20 REM Version B1.1
   30 REM Author  Eddy Hunt
   40 REM BEEBUG  July 1989
   50 REM Program subject to copyright
   60 :
  100 ON ERROR GOTO 170
  110 DIM L$(15,2),L%(15)
  120 VDU2,1,27,1,65,1,5,1,27,1,85,1,49
  130 FOR I%=1 TO 4:PROCline(I%):NEXT I%
  140 VDU3,1,27,1,64
  150 END
  160 :
  170 MODE7:REPORT:PRINT" at line ";ERL
  180 END
  190 :
 1000 DEF PROCline(I%)
 1010 L%=0:READ K$
 1020 REPEAT
 1030 IF LEN(K$)>2 THEN PROCword ELSE PR
OCchar
 1040 L%=L%+1:READ K$
 1050 UNTIL K$="#"
 1060 PROCprint(I%)
 1070 ENDPROC
 1080 :
 1090 DEF PROCchar
 1100 L%(L%)=3
 1110 IF LEN(K$)=2 THEN PROCpair(K$):END
PROC
 1120 IF K$>="A" AND K$<="Z" THEN PROCpa
ir(CHR$(ASC(K$)+32)+K$):ENDPROC
 1130 L$(L%,0)="":L$(L%,2)=""
 1140 IF K$="" THEN K$=" "
 1150 L$(L%,1)=" "+K$+" "
 1160 ENDPROC
 1170 :
 1180 DEF PROCpair(P$)
 1190 L$(L%,0)=" "+LEFT$(P$,1)+" "
 1200 L$(L%,2)=" "+RIGHT$(P$,1)+" "
 1210 L$(L%,1)="":ENDPROC
 1220 :
 1230 DEF PROCword
 1240 IF INSTR(K$,"@") THEN PROCsplit:EN
DPROC
 1250 L$(L%,1)=K$:L$(L%,0)=""
 1260 L$(L%,2)=""
 1270 L%(L%)=LEN(K$)
 1280 ENDPROC
 1290 :
 1300 DEF PROCsplit
 1310 L%(L%)=INSTR(K$,"@")-1
 1320 L$(L%,2)=LEFT$(K$,L%(L%))
 1330 L$(L%,1)=""
 1340 L$(L%,0)=MID$(K$,L%(L%)+2,LEN(K$))
 1350 ENDPROC
 1360 :
 1370 DEF PROCprint(N%)
 1380 LOCAL I%,J%
 1390 FOR I%=1 TO 7
 1400 IF N%<3 THEN PRINT "  ";
 1410 PROCbar(I%)
 1420 FOR J%=0 TO L%-1
 1430 PROCkey(I%,J%):PROCbar(I%)
 1440 NEXT J%:VDU 13:NEXT I%
 1450 ENDPROC
 1460 :
 1470 DEF PROCbar(I%)
 1480 IF I%=1 OR I%=7 THEN PRINT "-"; EL
SE PRINT "!";
 1490 ENDPROC
 1500 :
 1510 DEF PROCkey(I%,J%)
 1520 IF I%=1 OR I%=7 THEN PRINT STRING$
(L%(J%),"-");:ENDPROC
 1530 IF I%=2 OR I%=6 THEN PRINT SPC(L%(
J%));:ENDPROC
 1540 IF L$(J%,5-I%)="" THEN PRINT SPC(L
%(J%)); ELSE PRINT L$(J%,5-I%);
 1550 ENDPROC
 1560 :
 1570 DATA ESC,1!,2",3#,4$,5%,6&,7',8(,9
),0,-=,^~,\|,,,#
 1580 DATA " TAB ",Q,W,E,R,T,Y,U,I,O,P,@
,[{,_£,,,#
 1590 DATA CAPS@LOCK,CTRL,A,S,D,F,G,H,J,
K,L,;+,:*,]},"  RETURN ",#
 1600 DATA SHFT@LOCK,SHIFT ,Z,X,C,V,B,N,
M,",<",.>,/?,SHIFT ,DEL,COPY,#
```

An ADFS Disc Sector Editor

A.J Moore and R.A. Smith present a useful utility for Master owners using ADFS.

There are many disc sector editors available for DFS format discs, and these can be incredibly useful. By the same token, however, when using ADFS, the lack of a disc sector editor can be extremely infuriating. The program here fills this gap.

To enter the program, type in and save the listing given here. Running the program will assemble a sideways ROM image, and save this to disc with the filename 'ADFSED'. To load the image use a command such as:

```
    *SRLOAD ADFSED 8000 ZQ
```

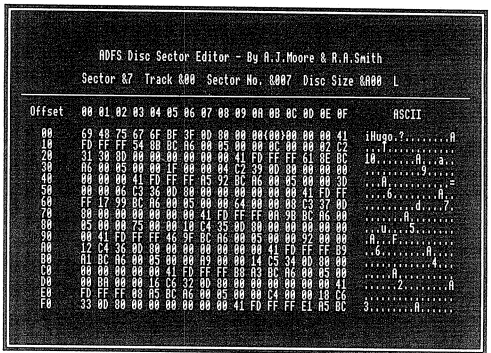
After loading, press Ctrl-Break in order to initialise the ROM image.

The sector editor will only work with a floppy disc in drive 0. It is not suitable for use with hard discs. To invoke the editor, use the command:

```
    *ADFSED (<sector number>)
```

The *sector number* is the number of the absolute sector at which to start editing (in hex). If this is omitted then sector zero is assumed. Absolute sector numbers are given as the last value on the line returned by *INFO or *EX. Thus, this method makes it very easy to start editing at the start of a particular file. If you wish to specify the start as a track and sector, then you need to translate this into an absolute sector number by multiplying the track number by sixteen and adding the sector number.

The editor shows a status bar at the top of the screen, giving the current sector, the current track and the current absolute sector number. Also, the disc size in sectors is shown, and a single letter indicating the type of ADFS disc format - 'S' for a small disc (single sided, 40 track), 'M' for a medium disc (single sided, 80 track) and 'L' for a large disc (double sided, 80 track). If the disc is a non-standard size, the letter 'X' is displayed.



```
          ADFS Disc Sector Editor - By A.J.Moore & R.A.Smith
       Sector &7  Track &00  Sector No. &007  Disc Size &A00  L
 ---------------------------------------------------------------------
 Offset   00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F       ASCII

   00     69 48 75 67 6F BF 3F 0D 80 00 00{00}00 00 00 41   iHugo.?........A
   10     FD FF FF 54 8B BC A6 00 05 00 00 0C 00 00 02 C2   ...T............
   20     31 30 8D 00 00 00 00 00 00 41 FD FF FF 61 8E BC   10.......A...a..
   30     A6 00 05 00 00 1F 00 00 04 C2 39 0D 80 00 00 00   ..........9.....
   40     00 00 00 41 FD FF FF A5 92 BC A6 00 05 00 00 3D   ...A...........=
   50     00 00 06 C3 36 0D 80 00 00 00 00 00 00 41 FD FF   ....6........A..
   60     FF 17 99 BC A6 00 05 00 00 64 00 00 08 C3 37 0D   .........d....7.
   70     80 00 00 00 00 00 00 41 FD FF FF 0A 9B BC A6 00   .......A........
   80     05 00 00 75 00 00 10 C4 35 0D 80 00 00 00 00 00   ...u....5.......
   90     00 41 FD FF FF 46 9F BC A6 00 05 00 00 92 00 00   .A...F..........
   A0     12 C4 36 0D 80 00 00 00 00 00 00 41 FD FF FF B9   ..6........A....
   B0     A1 BC A6 00 05 00 00 A9 00 00 14 C5 34 0D 80 00   ............4...
   C0     00 00 00 00 00 41 FD FF FF B8 A3 BC A6 00 05 00   .....A..........
   D0     00 BA 00 00 16 C6 32 0D 80 00 00 00 00 00 00 41   ......2........A
   E0     FD FF FF 08 A5 BC A6 00 05 00 00 C4 00 00 18 C6   ................
   F0     33 0D 80 00 00 00 00 00 00 41 FD FF FF E1 A5 BC   3........A......
```

The sector editor in action

The sector is displayed as 16 lines, each with 16 bytes, with the display being in mode 3. Each byte of the sector is shown as a two digit hexadecimal number, and the corresponding ASCII equivalent of this byte is displayed to the right of the data. On entering the editor, the cursor is placed on byte zero of the sector. The current cursor position is indicated by a bracket either side of the two digit value, and a flashing cursor placed under the corresponding ASCII value. In this particular mode, data is entered in ASCII i.e. any characters entered on the keyboard will be entered directly as ASCII characters, and the cursor will move along one place. In order to switch to hexadecimal editing, press the Copy key which toggles between modes. The cursor will change to two hyphens. Data is now entered as a hexadecimal number, and the cursor position remains static until you move it yourself.

To move around the display use the cursor keys. Used by themselves, the cursor keys will move the cursor one column or one row, up or down. When the cursor keys are pressed in combination with the Shift key, the cursor will be moved to the extreme left, right, top or bottom on the same row or column. To move to a different sector, the cursor keys are used in combination with the Ctrl key. Ctrl with cursor left or right moves backwards or forwards by one sector, whereas Ctrl cursor down or up moves backwards or forwards by a whole track.

If the data in a sector has been modified, you will be asked whether you wish to save that particular sector before moving on to the next one. When editing a sector, alterations are never made to the disc unless you confirm this when moving to a different sector or when you exit from the editor. To exit, simply press the Escape key.

The first two sectors (0 and 1) of an ADFS disc are special because they contain the free space map for the disc. These two sectors hold a check sum in the last byte of each sector, and it is absolutely vital that this is correct before quitting the editor, otherwise the disc contents will be lost. To help with this, if you are editing sectors 0 or 1, pressing Shift and Copy together will automatically re-calculate the checksum, and enter it into the correct place.

Finally, be careful not to accidentally destroy any data while editing a disc. If you are using the editor to recover lost files, then next months Workshop will contain some useful hints.

```
 10 REM Program ADFS Sector Editor
 20 REM Version B 1.00
 30 REM Authors A.J.Moore & R.A.Smith
 40 REM BEEBUG  July 1989
 50 REM Program Subject to Copyright
 60 :
100 pointer=&F2:buffer=&DD00
110 osargs=&FFDA:blk=&900
120 osasci=&FFE3:osbyte=&FFF4
130 osnewl=&FFE7:osrdch=&FFE0
140 osword=&FFF1:oswrch=&FFEE
150 cx=blk+16:cy=blk+17:res=blk+18
160 x=blk+19:y=blk+20
170 ox=blk+21:oy=blk+22
180 saveflag=blk+23:tstor=blk+24
190 tempstor=blk+25:mulstor=blk+26
200 DIM mem &1000
210 FOR pass%=4 TO 7 STEP 3
220 P%=&8000:O%=mem
230 [OPT pass%
240 EQUB 0:EQUB 0:EQUB 0
250 JMP service:EQUB &82
260 EQUB copy:EQUB 100
270 EQUS "ADFS Sector Editor"
280 .copy EQUB 0
290 EQUS "(C) 1989 BEEBUG":EQUB 0
300 .oc EQUS "ADFSED":EQUB &FF
310 .service CMP #4:BEQ comm:RTS
320 .comm PHY:LDX #0
330 .comm2 LDA (&F2),Y:AND #&DF
340 CMP oc,X:BNE comm3:INY
350 INX:BRA comm2
360 .comm3 LDA (&F2),Y:CMP #&2E
370 BEQ comm4:LDA oc,X:BMI comm5
380 .comm35 LDA #4:LDX &F4:PLY:RTS
390 .comm4 INY:LDA oc,Y:BMI comm35
400 .comm5 LDA (&F2),Y:INY:CMP #32
410 BEQ comm5:DEY:PLA:CLC
420 TYA:ADC &F2:STA &F2:LDA &F3
430 ADC #0:STA &F3:LDX #31
440 .setblk CLR blk,X:DEX:BPL setblk
450 LDA #1:STA blk+9
460 LDA #buffer MOD &100:STA blk+1
470 LDA #buffer DIV &100:STA blk+2
480 JSR setup:LDA #22:JSR oswrch
490 LDA #131:JSR oswrch:LDA #4
500 LDX #1:JSR osbyte:JSR printbar
510 JSR readsect:JSR getsize
520 JSR printcont
530 JSR print:LDA #0:STA saveflag
540 LDA #229:LDX #1:LDY #0:JSR osbyte
550 .mainloop LDA chksmflag
560 BEQ do_not_move:JSR colcur
570 LDA #15:LDX #0:JSR osbyte
580 JSR osrdch:PHA:LDA #202
590 LDX #0:LDY #255:JSR osbyte
600 TXA:AND #72:CMP #8
610 BEQ shift:CMP #64:BEQ ctrl
620 JMP keytable
630 .do_not_move:JMP dnm
640 .ctrl:JMP jmpctrl
650 .shift:JMP jmpshift
660 .keytable:PLA:CMP #136:BEQ left
670 CMP #137:BEQ right:CMP #138
680 BEQ down:CMP #139:BEQ up
690 CMP #135:BEQ change
700 .notcurkey:CMP #27:BNE ne
710 JMP escape:.ne PHA
720 LDA #255:STA saveflag:PLA
730 STA tempstor:LDA brac:CMP #ASC"-"
740 BEQ hexin:LDA tempstor
750 JMP printchar
760 .change JSR changebrac:JSR print
770 JMP mainloop
```

```
780 .left DEC x:BMI bac:JSR print
790 JMP mainloop:JMP do_move
800 .right INC x:LDA x:CMP #16:BEQ for
810 .do_move JSR print:JMP mainloop
820 .down INC y:LDA y:CMP #16
830 BEQ top:JSR print
840 .loopy JMP mainloop
850 .up DEC y:BMI bot:JSR print
860 JMP mainloop
870 .bac LDA #15:STA x:JMP up
880 .for LDA #0:STA x:JMP down
890 .top LDA #0:STA y:JSR print
900 JMP mainloop
910 .hexin LDY #255:.hexloop INY
920 LDA hexdat,Y:CMP #ASC"@"
930 BEQ loopy:CMP tempstor:BNE hexloop
940 TYA:PHA:LDA y:JSR asl:CLC:ADC x
950 TAY:LDA buffer,Y:JSR asl
960 STA &A8:PLA:ORA &A8
970 STA tempstor:JMP printchar
980 .hexdat:EQUS "0123456789ABCDEF@"
990 .escape:JSR saveit:LDA #11
1000 JSR oswrch:LDA #4:LDX #0
1010 JSR osbyte:LDA #229:LDX #0:LDY #0
1020 JSR osbyte:JSR osnewl:LDA #0:RTS
1030 .bot LDA #15:STA y:JSR print
1040 JMP mainloop
1050 .print LDA #32:STA vdu+3:LDA ox
1060 JSR mul3:STA vdu+1:LDA oy
1070 CLC:ADC #7:STA vdu+2
1080 JSR printcur:LDA ox:JSR mul3
1090 CLC:ADC #3:STA vdu+1
1100 JSR printcur:LDA y:STA oy
1110 LDA x:STA ox:JSR mul3
1120 STA vdu+1:LDA y:ADC #7
1130 STA vdu+2:LDA brac:STA vdu+3
1140 JSR printcur:LDA x:JSR mul3
1150 CLC:ADC #3:STA vdu+1
1160 LDA brac2:STA vdu+3:JSR printcur
1170 .colcur LDA x:CLC:ADC #61
1180 STA vdu+1:LDA y:CLC
1190 ADC #7:STA vdu+2:LDA #0
1200 STA vdu+3:JSR printcur:RTS
1210 .printcur LDX #0
1220 .pcloop:LDA vdu,X:JSR oswrch:INX
1230 CPX #4:BNE pcloop:RTS
1240 .mul3 STA mulstor:CLC:ADC mulstor
1250 CLC:ADC mulstor:LDY #10
1260 .mul3loop CLC:ADC #1:DEY
1270 BNE mul3loop:RTS
1280 .printcont JSR updatebar
1290 LDA #buffer MOD 256:STA &AA
1300 LDA #buffer DIV 256:STA &AB:LDA #0
1310 STA &AC:LDA #7:STA &A8
```

```
1320 .botscr LDA #11:STA &A9:LDA #61
1330 STA &AD:.curloop LDA #31
1340 JSR oswrch:LDA &A9
1350 JSR oswrch:LDA &A8:JSR oswrch
1360 JSR pritoa:LDA #31:JSR oswrch
1370 LDA &AD:JSR oswrch:LDA &A8
1380 JSR oswrch:JSR prchar:INC &AD
1390 INC &A9:INC &A9:INC &A9
1400 LDA &A9:CMP #57:BCC curloop
1410 INC &A8:LDA &A8:CMP #23
1420 BCC botscr:RTS
1430 .pritoa LDY &AC:LDA (&AA),Y:TAX
1440 CPX #16:BCC zero:LDA #16
1450 LDY #0:JMP itoa
1460 .zero LDA #ASC"0":JSR oswrch
1470 LDA #16:LDY #0
1480 .itoa CPY #0:BEQ noY:TXA
1490 PHA:TYA:JSR printit
1500 PLA:TAX:CPX #16:BCC zro
1510 .orz TXA:JSR printit:RTS
1520 .zro LDA #48:JSR oswrch:JMP orz
1530 .noY TXA
1540 .printit STA ln:JSR lsr:TAX
1550 LDA hexdat,X:CMP #ASC"0"
1560 BEQ misszero:JSR oswrch
1570 .misszero LDA ln:AND #15:TAX
1580 LDA hexdat,X:JSR oswrch:RTS
1590 .ln EQUB 0
1600 .prchar LDY &AC:LDA (&AA),Y
1610 INC &AC
1620 .dorspc CMP #32:BCC dot:CMP #127
1630 BCS dot:JMP oswrch
1640 .dot LDA #ASC".":JMP oswrch
1650 .printchar LDA x:JSR mul3:CLC
1660 ADC #1:STA vdu+1:LDA y
1670 CLC:ADC #7:STA vdu+2
1680 LDA #0:STA vdu+3:JSR printcur
1690 LDA tempstor:CMP #16:BCC printzero
1700 .checkinput CMP #32:BCC checkbrac
1710 CMP #127:BCS checkbrac:TAX:LDY #0
1720 LDA #16:JSR itoa:JSR colcur
1730 LDA tempstor:JSR oswrch:JSR storit
1740 LDA brac:CMP #ASC"-":BEQ no
1750 JMP right
1760 .printzero PHA:LDA #48:JSR oswrch
1770 PLA:JMP checkinput
1780 .no JMP mainloop
1790 .checkbrac TAX:LDY #0:LDA #16
1800 JSR itoa:JSR colcur:LDA #ASC"."
1810 JSR oswrch:JSR storit:LDA brac
1820 CMP #ASC"-":BEQ no:JMP right
1830 .storit LDA y:JSR asl:CLC
1840 ADC x:TAY:LDA tempstor
1850 STA buffer,Y:RTS
```

```
 1860 .changebrac LDA brac:CMP #ASC"{"
 1870 BEQ tohex:LDA #ASC"{":STA brac
 1880 LDA #ASC"}":STA brac2:RTS
 1890 .tohex LDA #ASC"-":STA brac
 1900 LDA #ASC"-":STA brac2:RTS
 1910 .jmpctrl PLA:PHA:CMP #136
 1920 BCC brakeytable:CMP #140
 1930 BCS brakeytable
 1940 JSR saveit:PLA:CMP #136
 1950 BEQ decsec:CMP #137:BEQ incsec
 1960 CMP #138:BEQ dectrk:CMP #139
 1970 BEQ inctrk:JMP mainloop
 1980 .brakeytable JMP keytable
 1990 .decsec
 2000 DEC blk+8:LDA blk+8:CMP #255
 2010 BNE seced:DEC blk+7:LDA blk+7
 2020 CMP #255:BNE seced:LDA size+1
 2030 STA blk+7:LDA size:STA blk+8
 2040 DEC blk+8:LDA blk+8:CMP #255
 2050 BNE seced:DEC blk+7
 2060 .seced JSR readsect:JSR printcont
 2070 JMP mainloop
 2080 .incsec INC blk+8:LDA blk+8
 2090 CMP size:BEQ endofdisc
 2100 .falloff LDA blk+8:CMP #0
 2110 BNE seced:INC blk+7:LDA blk+7
 2120 CMP size+1:BCC seced:LDA #0
 2130 STA blk+7:JMP seced
 2140 .inctrk JMP inctrak
 2150 .endofdisc LDA blk+7:CMP size+1
 2160 BNE falloff:LDA #0:STA blk+7
 2170 STA blk+8:JMP seced
 2180 .dectrk LDA blk+8:SEC:SBC #16
 2190 STA blk+8:BCS seced:DEC blk+7
 2200 LDA blk+7:CMP #255:BNE seced
 2210 CLC:LDA size:ADC blk+8
 2220 STA blk+8:LDA size+1:ADC blk+7
 2230 STA blk+7:JMP seced
 2240 .inctrak LDA blk+8:CLC:ADC #16
 2250 STA blk+8:BCC falloff2:INC blk+7
 2260 .falloff2 LDA blk+7:CMP size+1
 2270 BCC notend:SEC:LDA blk+8:SBC size
 2280 STA blk+8:LDA blk+7:SBC size+1
 2290 STA blk+7
 2300 .notend JMP seced
 2310 .readsect LDA #8:STA blk+5:JMP osc
all:.writesect LDA #&A:STA blk+5
 2320 .oscall:LDA #&72:LDX #blk MOD256
 2330 LDY #blk DIV256:JSR osword:RTS
 2340 .jmpshift:PLA:CMP #136
 2350 BEQ shiftleft:CMP #137
 2360 BEQ shiftright:CMP #138
 2370 BEQ shiftdown:CMP #139:BEQ shiftup
 2380 CMP#135:BEQ checksum:JMP notcurkey
```

```
 2390 .checksum LDA blk+7:BNE notFS
 2400 LDA blk+8:CMP #2:BCS notFS:CLC
 2410 LDY #255:TYA
 2420 .chksumloop ADC buffer-1,Y:DEY
 2430 BNE chksumloop:STA tempstor:LDA x
 2440 STA cx:LDA y:STA cy:LDA #0
 2450 STA chksmflag:LDA #15:STA x
 2460 STA y:LDA #255:STA saveflag
 2470 JMP printchar
 2480 .notFS JMP mainloop
 2490 .shiftleft LDA #0:STA x:JSR print
 2500 JMP mainloop
 2510 .shiftright LDA #15:STA x
 2520 JSR print:JMP mainloop
 2530 .shiftup LDA #0:STA y:JSR print
 2540 JMP mainloop
 2550 .shiftdown LDA #15:STA y:JSR print
 2560 JMP mainloop
 2570 .printbar:JSR printtext:EQUB 31
 2580 EQUB 11:EQUB 2:EQUS "Sector &0  Tr
ack &00  Sector No. &000  Disc Size &000
  X":EQUB 31
 2590 EQUB 14:EQUB 0:EQUS "ADFS Disc Sec
tor Editor - By A.J.Moore & R.A.Smith"
 2600 EQUB 31:EQUB 0:EQUB 3
 2610 EQUS STRING$(80,"_"):EQUB 13:EQUS
"  Offset"+STRING$(3," ")+"00 01 02 03 0
4 05 06 07 08 09 0A 0B 0C 0D 0E 0F"+STRI
NG$(8," ")+"ASCII@"
 2620 LDX #0
 2630 .offset LDA #31:JSR oswrch:LDA #4
 2640 JSR oswrch:TXA:CLC
 2650 ADC #7:JSR oswrch:LDA hexdat,X
 2660 JSR oswrch:LDA #ASC"0":JSR oswrch
 2670 INX:CPX #16:BNE offset:RTS
 2680 .updatebar LDA #19:JSR vdu31
 2690 LDA blk+8:AND #15:TAX:LDY #0
 2700 LDA #16:JSR itoa:LDA #29
 2710 JSR vdu31:LDA blk+8:JSR lsr
 2720 STA tstor:LDA blk+7:JSR asl
 2730 ORA tstor:CMP #16:BCS printzero2
 2740 PHA:LDA #ASC"0":JSR oswrch:PLA
 2750 .printzero2 TAX:LDY #0:LDA #16
 2760 JSR itoa:LDA #45:JSR vdu31
 2770 LDA blk+7:BNE showsecno:LDA blk+8
 2780 CMP #16:BCS writezero:LDA #ASC"0"
 2790 JSR oswrch
 2800 .writezero LDA #ASC"0":JSR oswrch
 2810 .showsecno LDX blk+8:LDY blk+7
 2820 LDA #16:JMP itoa
 2830 .vdu31 PHA:LDA #31:JSR oswrch
 2840 PLA:JSR oswrch:LDA #2:JMP oswrch
 2850 .getsize
 2860 LDA #61:JSR vdu31:LDA size+1
 2870 BNE printdiscsize:LDA size:CMP #16
 2880 BCS writezero2:LDA #ASC"0":JSR osw
rch
 2890 .writezero2 LDA #ASC"0":JSR oswrch
 2900 .printdiscsize
 2910 LDX size:LDY size+1:LDA #16
 2920 JSR itoa:LDA #66:JSR vdu31
 2930 LDA size:CMP #&80:BEQ smalldisc
 2940 CMP #0:BNE rts:LDA size+1
 2950 CMP #5:BEQ mediumdisc:CMP #10
 2960 BEQ largedisc:.rts RTS
 2970 .largedisc LDA #ASC"L":JMP oswrch
 2980 .mediumdisc LDA #ASC"M":JMP oswrch
 2990 .smalldisc LDA size+1:CMP #2
 3000 BNE rts:LDA #ASC"S":JMP oswrch
 3010 .saveit
 3020 LDA saveflag:CMP #0:BEQ end
 3030 LDX #0:JSR printtext:EQUB 31:EQUB
34:EQUB 24:EQUS "Save (Y/N)?@"
 3040 JSR osrdch:JSR oswrch:CMP #ASC"Y"
 3050 BEQ savesec
 3060 .end
 3070 LDA #0:STA saveflag:JSR printtext
 3080 EQUB 31:EQUB 34:EQUB 24
 3090 EQUS STRING$(12," "):EQUS "@":RTS
 3100 .savesec JSR writesect:JMP end
 3110 .done3 RTS
 3120 .convert
 3130 LDY #&FF:CLR res:CLR res+1
 3140 .skipspaces
 3150 INY:LDA (pointer),Y:CMP #32
 3160 BEQ skipspaces:CMP #48:BEQ skipspa
ces
 3170 CMP #13:BEQ done3:JSR findnum
 3180 CPX #255:BEQ bdm:STX res
 3190 JSR loadreg:JSR findnum:CPX #255
 3200 BEQ bdm:LDA res:JSR asl
 3210 STA res:TXA:ORA res
 3220 STA res:JSR loadreg:JSR findnum
 3230 CPX #255
 3240 .bdm BEQ bdnm:LDA res:STA res+1
 3250 STX res:LDA res+1:JSR asl
 3260 ORA res:STA res:LDA res+1
 3270 JSR lsr:STA res+1:JSR loadreg
 3280 JSR findnum:CPX #255:BEQ bdnm
 3290 LDA res+1:JSR asl:STA res+1
 3300 LDA res:JSR lsr:ORA res+1
 3310 STA res+1:LDA res:JSR asl
 3320 STA res:TXA:ORA res
 3330 STA res:PHA:PHA
 3340 JMP done2
 3350 .done PLA:PLA:RTS
 3360 .lsr LSR A:LSR A:LSR A:LSR A:RTS
 3370 .asl ASL A:ASL A:ASL A:ASL A:RTS
```

```
 3380 .findnum LDX #0
 3390 .findit CMP hexdat,X:BEQ gotit:INX
 3400 CPX #16:BEQ bm:JMP findit
 3410 .gotit RTS
 3420 .bm LDX #255:RTS
 3430 .bdnm:.badnum LDX #errors2-errors:
BRA err
 3440 .done2 INY:LDA (pointer),Y:CMP #32
 3450 BEQ done2:CMP #13:BEQ done
 3460 JMP bdnm
 3470 .loadreg
 3480 INY:LDA (pointer),Y:CMP #13
 3490 BEQ done:CMP #32:BEQ done2:RTS
 3500 .setup LDA #0:LDY #0:JSR osargs
 3510 LDX #0:CMP #8:BNE err:JSR readsect
 3520 LDA buffer+252:STA size:LDA buffer
+253
 3530 STA size+1:LDA #16
 3540 JSR convert:LDA res+1:CMP size+1
 3550 BCS toobig:STA blk+7:LDA res
 3560 STA blk+8:RTS
 3570 .bd BRA badnum
 3580 .errors EQUB 254
 3590 EQUS "Not ADFS":EQUB 0
 3600 .errors2 EQUB 254
 3610 EQUS "Bad sector number"
 3620 EQUB 0
 3630 .err CLR &100:LDY #0
 3640 .err2 LDA errors,X:STA &101,Y
 3650 INX:INY:CMP #0:BNE err2:JMP &100
 3660 .toobig BEQ toobig2:BCS bd
 3670 .toobig2 STA blk+7:LDA res
 3680 CMP size:BCS bd:STA blk+8:RTS
 3690 .printtext PLA:STA &72:PLA
 3700 STA &73:INC &72:BNE printtextloop
 3710 INC &73
 3720 .printtextloop
 3730 LDY #0:LDA (&72),Y:CMP #64
 3740 BEQ endprintloop:JSR osasci:INC &7
2
 3750 BNE printtextloop:INC &73:JMP prin
ttextloop
 3760 .endprintloop
 3770 INC &72:BNE endprint:INC &73
 3780 .endprint JMP (&72)
 3790 .dnm LDA#255:STA chksmflag:LDA cx
 3800 STA x:LDA cy:STA y:JMP do_move
 3810 .vdu EQUB 31:EQUW 0:EQUB 0
 3820 .if EQUB 255:.size EQUW &0A00
 3830 .brac EQUB ASC("{")
 3840 .brac2 EQUB ASC("}")
 3850 .chksmflag EQUB 255:]:NEXT
 3860 OSCLI "SAVE ADFSED "+STR$~mem+" "+
STR$~O%
```

Mathematical Transformations (Part 1)

Keith Sumner describes his latest contribution to our educational programs, a highly professional approach to mathematical transformations.

The program TFORM listed at the end of this article implements a set of 2-D transformation matrices which enables the manipulation of geometrical shapes to be performed with ease. The program as it stands uses the implementation to provide a self teaching tutorial environment for school children who are being introduced to geometrical transforms in the maths class.

This month we shall cover the main features of the program, leaving a number of additional functions to be added in part two, where we shall also be providing more detailed information on the internal workings of the program itself. To start with we'll look at how to run the program, and then take a look at the theory behind it. You may prefer to read this first before trying out the program.

Because the program is being presented in two parts, please make sure that you keep to the line numbering in the listing to ensure that the additional code will correctly 'mesh' in next time. You are also recommended to save a copy of the program before you start using it.

USING THE PROGRAM FOR TEACHING

We will now deal with the use of the program. The first step is to define the object to be the subject of your transformations. When the program is run it prompts for a shape. There are three options available at this stage in response to this prompt:

1. RECTANGLE - gives a predefined 3 by 2 rectangle located at the origin.
2. TRIANGLE - gives a predefined isosceles triangle located at the origin.
3. END - Stops execution of the program.

Since each of the three options begins with a different letter (R, T or E), just typing this letter followed by Return is sufficient to select the required action.

TRANSFORMING THE DEFINED OBJECT

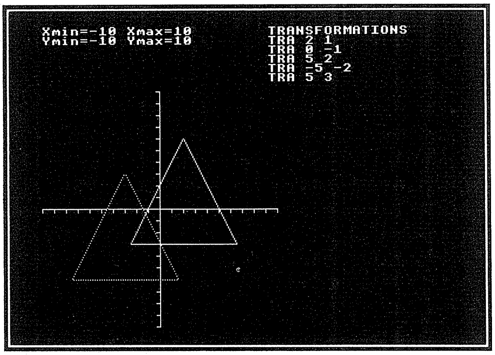
Three basic transformations are implemented this month. Others will be added in part 2. The three transformations are:

1. TRANSLATE: T(tx,ty) - This causes all co-ordinates defining the object to be translated by some user-specified amounts (tx,ty). The matrix for this operation is given by:

```
                                [1  0  0]
[x' y' 1] = [x y 1] . [0  1  0]
                                [tx ty 1]
```

The net effect is:

```
x' = x + tx
y' = y + ty
```



```
Xmin=-10 Xmax=10           TRANSFORMATIONS
Ymin=-10 Ymax=10           TRA 2 1
                           TRA 0 -1
                           TRA 5 2
                           TRA -5 -2
                           TRA 5 3
```

Example of translation

This command is executed by typing the word TRANSLATE, followed by two parameters giving the required displacement of the shape, firstly in the x direction and then in the y direction. The TRANSLATE command can be entered as the abbreviated 3 character word TRA. Thus the command format could be:

```
TRANSLATE 2 3 <Return>
```

or:

```
TRANS 2 3 <Return>
```

or at its shortest:

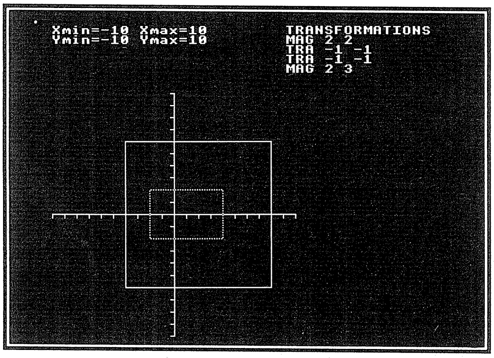
```
TRA 2 3 <Return>
```

2. ROTATE: R(theta) - This causes the object to be rotated anti-clockwise about the origin by some specified angle theta. The matrix for this operation is:

```
                          [ cos(theta) sin(theta) 0]
[x' y' 1] = [x y 1] . [-sin(theta) cos(theta) 0]
                          [     0          0      1]
```

The net effect is:

```
   x' = x.cos(theta) - y.sin(theta)
   y' = x.sin(theta) + y.cos(theta)
```



```
Xmin=-10 Xmax=10        TRANSFORMATIONS
Ymin=-10 Ymax=10        MAG 2 2
                        TRA -1 -1
                        TRA -1 -1
                        MAG 2 3
```

Example of rotation

This command is executed by typing the word ROTATE followed by a single parameter giving the required angle of rotation in degrees. A positive angle will cause rotation in an anti-clockwise direction, while a negative angle causes rotation in a clockwise direction. The ROTATE command can be entered abbreviated to the 3 characters ROT. Thus the command format could be:

```
   ROTATE -75 <Return>
```

or at its shortest:

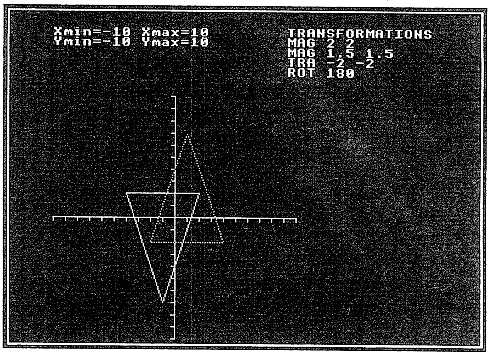
```
   ROT -75 <Return>
```

3. MAGNIFY: M(mx,my) - This causes the object to be magnified by some user specified factors in the x and y directions. The matrix for this operation is:

```
                            [mx 0  0]
[x' y' 1]  = [x y 1] . [0  my 0]
                            [0  0  1]
```

The net effect is:

```
   x' = mx.x
   y' = my.y
```



```
Xmin=-10 Xmax=10        TRANSFORMATIONS
Ymin=-10 Ymax=10        MAG 2 2
                        MAG 1.5 1.5
                        TRA -2 -2
                        ROT 180
```

Example of enlargement

This command is executed by typing the word MAGNIFY followed by two parameters giving the required expansion factor of the shape in first the x direction and then the y direction. The MAGNIFY command can be abbreviated to MAG. Thus the command format could be:

```
   MAGNIFY 0.5 2 <Return>
```

or at its shortest:

```
   MAG 0.5 2 <Return>
```

Two other useful commands are :

4. RESET (RES) - This initializes the 3x3 transformation matrix used to be the identity matrix, thus returning the shape to its original proportions, orientations and position.

5. END - Exits from the transformation mode, and returns the user to the specify/design shape mode.

THEORY OF MATRIX TRANSFORMATIONS

The transformations are implemented using a set of 3x3 matrices which act on a position vector representing the co-ordinates of a vertex of the object. The matrix can be represented as:

```
   [a d 0]
   [b e 0]
   [c f 1]
```

and the position vector of a vertex as:

```
   [x y 1]
```

so that a transformation of (x,y,1) is executed by the multiplication of the matrix and the vector so:

```
                           [a d 0]
[x' y' 1] = [x y 1]  .  [b e 0]
                           [c f 1]
```

which can be written in expanded form as

```
   x' = a.x + b.y + c
   y' = d.x + e.y + f
```

where the prime (') denotes the new transformed co-ordinates.

Multiple transformations on a position vector are achieved by consecutive application of transformation matrices. Thus let:

P represent the co-ordinates of the starting point,
T1 represent the matrix for some transformation, and
T2 represent the matrix for a second transformation

Hence:

```
   P1 = P.T1
```

and:

```
   P2 = P1.T2
```

so that:

```
   P2 = (P.T1).T2 = P.(T1.T2)
```

where the brackets indicate the priority of the operation, and hence the order in which the multiplications are carried out.

Part two of the Transformation program will be published next month. In the meantime you may find it helpful to familiarise yourself with the use of the program by experimenting with the transformations described above.

```
   10 REM Program TFORM Part 1
   20 REM Version B1.1
   30 REM Author  K.N.Sumner
   40 REM BEEBUG  July 1989
   50 REM Program subject to copyright
   60 :
  100 MODE4:VDU19,0,4,0,0,0
  110 ON ERROR GOTO180
  120 PROCsetup
  130 PROCsetup_matrix
  140 REPEAT
  150 PROCaxes(TRUE):PROCshape
  160 IF NOT exit PROCtforms
  170 UNTIL exit
  180 @%=10:MODE7:*FX4
  190 IF NOT exit REPORT:PRINT" at line
";ERL
  200 END
  210 :
 1000 DEF PROCsetup
 1010 DIM ta(2)
 1020 xmin=-10:ymin=-10
 1030 xmax=10:ymax=10
 1040 nmarkers=10:xcent=400:ycent=400
 1050 len=400:step=40:tick=12:@%=10
 1060 xfact=len/nmarkers:yfact=xfact
 1070 PRINTTAB(0,0)"Xmin=";xmin;" Xmax="
;xmax
 1080 PRINTTAB(0,1)"Ymin=";ymin;" Ymax="
;ymax
 1090 @%=&20204:exit=FALSE
 1100 ENDPROC
 1110 :
 1120 DEF PROCaxes(clr%)
 1130 IF clr% VDU24,0;0;800;800;:GCOL0,1
:CLG
 1140 MOVE0,ycent:DRAW xcent+len,ycent
 1150 MOVExcent,0:DRAW xcent,ycent+len
 1160 FOR X%=0 TO 800 STEP step
 1170 MOVE xcent,X%:DRAW xcent-tick,X%
 1180 MOVE X%,ycent:DRAW X%,ycent-tick
 1190 NEXT
 1200 ENDPROC
 1210 PRINT:pos=VPOS
 1220 DEF PROCshape
 1230 PROCscale(0,xfact,yfact,TRUE)
 1240 PROCtrans(0,xcent,ycent,FALSE)
 1250 REPEAT
 1260 VDU28,24,6,39,0,12:ok=FALSE
 1270 xpos=0:ypos=0:nc=0
 1280 PRINTTAB(0,0)"SHAPE"SPC12;:INPUTTA
B(5,0);" :"S$
 1290 IFFNstr("R",0) ok=TRUE:PROCrectang
 1300 IFFNstr("T",0) ok=TRUE:PROCtriang
 1320 exit=FNstr("E",0)
 1330 UNTIL (exit OR ok)
 1340 ENDPROC
 1350 :
 1360 DEF PROCrectang
 1370 xsize=3:ysize=2:nc=4
 1380 X(0)=xpos:Y(0)=ypos
 1390 X(1)=X(0)+xsize:Y(1)=Y(0)
 1400 X(2)=X(1):Y(2)=Y(1)+ysize
 1410 X(3)=X(0):Y(3)=Y(2)
 1420 PROCtransform(0,nc)
 1430 PROCdraw(4,nc)
 1440 ENDPROC
```

```
1450 :
1460 DEF PROCtriang
1470 xsize=2:ysize=3:nc=3
1480 X(0)=xpos:Y(0)=ypos
1490 X(1)=X(0)+xsize:Y(1)=Y(0)
1500 X(2)=X(1)-xsize/2:Y(2)=Y(1)+ysize
1510 PROCtransform(0,nc)
1520 PROCdraw(4,nc)
1530 ENDPROC
1540 :
1990 DEF PROCtforms
2000 CLS:PRINTTAB(0,0)"TRANSFORMATIONS"
:first=FALSE:pos=0
2020 REPEAT:VDU28,24,6,39,1
2030 PRINTTAB(0,pos)SPC(15);:INPUTTAB(0
,pos)S$;
2040 IF FNstr("TRA",2) PROCexec(0)
2050 IF FNstr("ROT",1) PROCexec(1)
2060 IF FNstr("MAG",2) PROCexec(2)
2130 IF FNstr("RES",0) PROCexec(9)
2180 UNTIL FNstr("END",0)
2190 ENDPROC
2200 :
2210 DEF FNstr(L$,V%)
2220 L%=LENL$:S%=LENS$
2230 IF LEFT$(S$,L%)=L$ THEN =FNgetval(
V%,INSTR(S$," ")) ELSE =FALSE
2240 :
2250 DEF FNgetval(V%,ptr%)
2260 IF V%=0 =TRUE
2270 IF ptr%=0 PROCargs:=FALSE
2280 v%=0
2290 REPEAT:v%=v%+1
2300 ptr%=FNstrip(v%,ptr%)
2310 UNTIL ptr%=-1 OR v%=V%
2320 IF ptr%=-1 THEN =FALSE
2330 =TRUE
2340 :
2350 DEF FNstrip(v%,p%)
2360 REPEAT:p%=p%+1
2370 UNTIL MID$(S$,p%,1)<>" " OR p%>S%
2380 IF p%>S% PROCargs:=-1
2390 pt%=p%
2400 REPEAT:pt%=pt%+1
2410 UNTIL MID$(S$,pt%,1)=" "OR pt%>S%
2420 ta(v%)=VAL(MID$(S$,p%,pt%-p%))
2430 =pt%
2440 :
2450 DEF PROCargs
2460 PRINTTAB(0,pos)SPC15;TAB(0,pos)"Ar
guments";:VDU7
2470 A=INKEY(200)
2480 ENDPROC
```

```
2490 :
2500 REM >>>>>>> GRAPHICAL TRANSFORMATI
ON KERNEL <<<<<<<
2510 :
2520 DEF PROCsetup_matrix
2530 nmat=3:npts=10
2540 DIM X(npts),Y(npts)
2550 DIM XT(npts),YT(npts)
2560 DIM XTMP(npts),YTMP(npts)
2570 DIM mat(3,3,nmat-1)
2580 FOR M=0 TO nmat-1:PROCinitialize(M
):NEXT
2590 ENDPROC
2600 :
2610 DEF PROCset_matrix(M,A,B,C,D,E,F)
2620 mat(1,1,M)=A:mat(2,1,M)=D
2630 mat(1,2,M)=B:mat(2,2,M)=E
2640 mat(1,3,M)=C:mat(2,3,M)=F
2650 ENDPROC
2660 :
2670 DEF PROCinitialize(M)
2680 PROCset_matrix(M,1,0,0,0,1,0)
2690 mat(3,1,M)=0
2700 mat(3,2,M)=0
2710 mat(3,3,M)=1
2720 ENDPROC
2730 :
2740 DEF PROCtrans(M,tx,ty,set)
2750 IF set PROCset_matrix(M,1,0,tx,0,1
,ty) ELSE PROCmmi(M,1,0,tx,0,1,ty)
2760 ENDPROC
2770 :
2780 DEF PROCscale(M,mx,my,set)
2790 IF set PROCset_matrix(M,mx,0,0,0,m
y,0) ELSE PROCmmi(M,mx,0,0,0,my,0)
2800 ENDPROC
2810 :
2820 DEF PROCrotate(M,angle,set)
2830 A=COS(RAD(angle)):B=-SIN(RAD(angle
)):D=-B:E=A
2840 IF set PROCset_matrix(M,A,B,0,D,E,
0) ELSE PROCmmi(M,A,B,0,D,E,0)
2850 ENDPROC
2860 :
2870 DEF PROCshear(M,shx,shy,set)
2880 IF set PROCset_matrix(M,1,shx,0,sh
y,1,0) ELSE PROCmmi(M,1,shx,0,shy,1,0)
2890 ENDPROC
2900 :
2910 DEF PROCmultmat(T1,T2,R)
2920 REM T1=transform matrix 1
2930 REM T2=transform matrix 2
2940 REM  R=resultant composite transfo
```

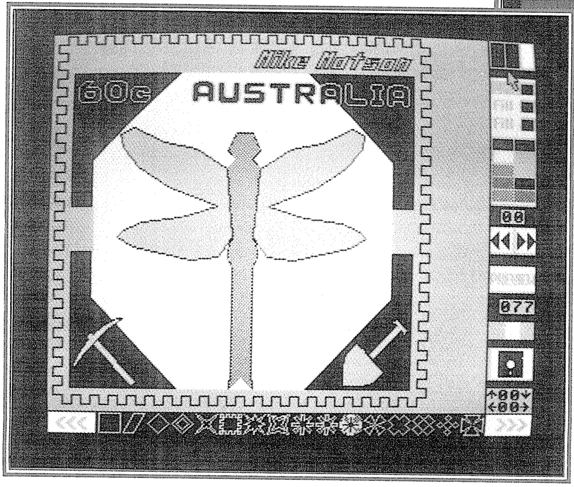
```
rmation matrix : R=T1.T2
 2950 IF R=T1 OR R=T2 PRINT"Error: Matri
x T1/T2 cannot hold result matrix":ENDPR
OC
 2960 FOR R%=1 TO 3:FOR C%=1 TO 3
 2970 mat(R%,C%,R)=mat(R%,1,T1)*mat(1,C%
,T2)+mat(R%,2,T1)*mat(2,C%,T2)+mat(R%,3,
T1)*mat(3,C%,T2)
 2980 NEXT:NEXT
 2990 ENDPROC
 3000 :
 3010 DEF PROCmmi(M,A,B,C,D,E,F)
 3020 REM multiply matrix M with matrix
 3030 REM specified in parameter list
 3040 tl=mat(1,1,M)*A+mat(2,1,M)*B
 3050 ml=mat(1,2,M)*A+mat(2,2,M)*B
 3060 bl=mat(1,3,M)*A+mat(2,3,M)*B+C
 3070 tr=mat(1,1,M)*D+mat(2,1,M)*E
 3080 mr=mat(1,2,M)*D+mat(2,2,M)*E
 3090 br=mat(1,3,M)*D+mat(2,3,M)*E+F
 3100 mat(1,1,M)=tl:mat(2,1,M)=tr
 3110 mat(1,2,M)=ml:mat(2,2,M)=mr
 3120 mat(1,3,M)=bl:mat(2,3,M)=br
 3130 ENDPROC
 3140 :
 3150 DEF PROCtransform(M,N)
 3160 REM transform N coordinates
 3170 REM according to matrix M
 3180 LOCAL N%
 3190 FOR N%=0 TO N-1
 3200 XT(N%)=FNtfmx(M,X(N%),Y(N%))
 3210 YT(N%)=FNtfmy(M,X(N%),Y(N%))
 3220 NEXT
 3230 ENDPROC
 3240 :
 3250 DEF FNtfmx(M,x%,y%)
 3260 =INT(.5+x%*mat(1,1,M)+y%*mat(1,2,M
)+mat(1,3,M))
 3270 :
 3280 DEF FNtfmy(M,x%,y%)
 3290 =INT(.5+x%*mat(2,1,M)+y%*mat(2,2,M
)+mat(2,3,M))
 3300 :
 3310 DEF PROCerase(P%,N)
 3320 REM delete contents of point
 3330 REM buffer from screen
 3340 IF N=0 ENDPROC
 3350 LOCAL X%:GCOL0,0
 3360 PLOT4,XTMP(0),YTMP(0)
 3370 FOR X%=1 TO N:PLOT P%+1,XTMP(X%MOD
N),YTMP(X%MODN):NEXT
 3380 ENDPROC
 3390 :
 3400 DEF PROCdraw(P%,N)
 3410 REM draw contents of point
 3420 REM buffer to screen
 3430 IF N<2 ENDPROC
 3440 LOCAL X%:GCOL0,1
 3450 PLOT4,XT(0),YT(0)
 3460 FOR X%=1 TO N:PLOT P%+1,XT(X%MODN)
,YT(X%MODN):NEXT
 3470 ENDPROC
 3480 :
 3490 DEF PROCstore_mat(M,N)
 3500 REM stores matrix M in matrix N
 3510 FOR A%=1 TO 3:FOR B%=1 TO 3
 3520 mat(A%,B%,N)=mat(A%,B%,M):NEXT:NEX
T
 3530 ENDPROC
 3540 :
 3550 DEF PROCload_mat(M,N)
 3560 REM loads matrix M with matrix N
 3570 FOR A%=1 TO 3:FOR B%=1 TO 3
 3580 mat(A%,B%,M)=mat(A%,B%,N):NEXT:NEX
T
 3590 ENDPROC
 3600 :
 3910 REM >>>>>>> GRAPHICAL TRANSFORMATI
ON KERNEL <<<<<<<
 3920 :
 4010 DEF PROCexec(func)
 4020 IF NOT first first=TRUE:PROCinitia
lize(1)
 4030 IF func=0 PROCtrans(1,ta(1),ta(2),
FALSE)
 4040 IF func=1 PROCrotate(1,ta(1),FALSE
)
 4050 IF func=2 PROCscale(1,ta(1),ta(2),
FALSE)
 4120 IF func=9 PROCinitialize(1)
 4130 PROCscale(0,xfact,yfact,TRUE):PROC
trans(0,xcent,ycent,FALSE)
 4140 PROCmultmat(0,1,2)
 4150 FOR X%=0 TO nc:XTMP(X%)=XT(X%)
 4160 YTMP(X%)=YT(X%):NEXT
 4170 PROCtransform(2,nc)
 4180 PROCdraw(20,nc):PROCrastinate(100)
 4190 PROCerase(20,nc):PROCrastinate(100
)
 4200 PROCdraw(4,nc):PROCrastinate(100)
 4210 PROCerase(4,nc):PROCdraw(4,nc)
 4220 PROCaxes(FALSE)
 4230 pos=VPOS
 4240 ENDPROC
 4250 :
 4370 DEF PROCrastinate(T%)
 4380 TIME=0:REPEAT UNTIL TIME>T%/3
 4390 ENDPROC
```

STRETCH by 4Mation

If you ever want to produce banners up to two metres long on your printer, consider using Stretch. David Somers examines this utility, aimed at the educational market, but with many other applications.

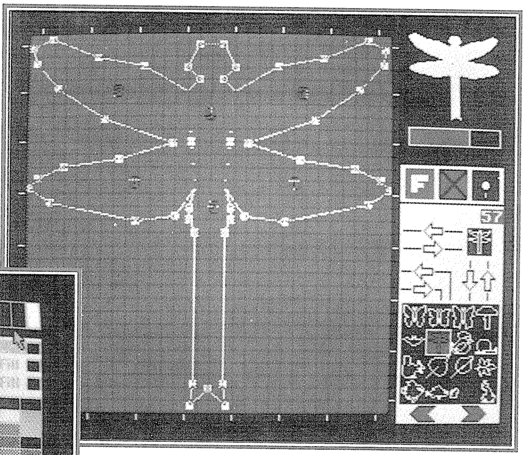
Product	Stretch
Supplier	4Mation Educational Resources, "Linden Lea", Rock Park, Barnstable, Devon EX32 9AQ. Tel. (0271) 45566.
Price	£26.45 inc. VAT and p&p

Stretch is a simple utility program enabling banners to be printed out on any Epson-compatible printer (including colour ribbon ones) and the Integrex and Canon colour ink-jet printers. The program is supplied either on one 80 track DFS or ADFS disc, or on two 40 track DFS discs. It works on the BBC B, B+, Master and Master Compact. It is also able to utilise sideways RAM for data storage, thus reducing the need for disc accesses.



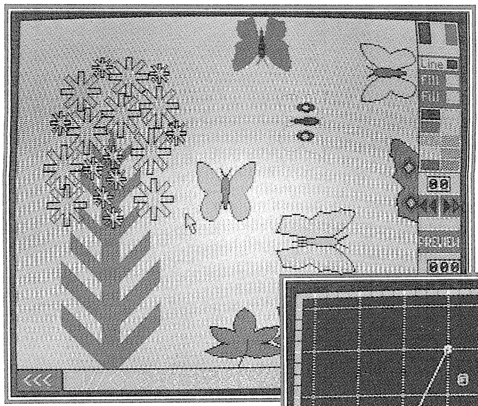
Unusually, the distribution discs are protected. They can be partially copied onto other discs to make working copies for everyday use, but these require the original disc to be inserted at the start of each session.

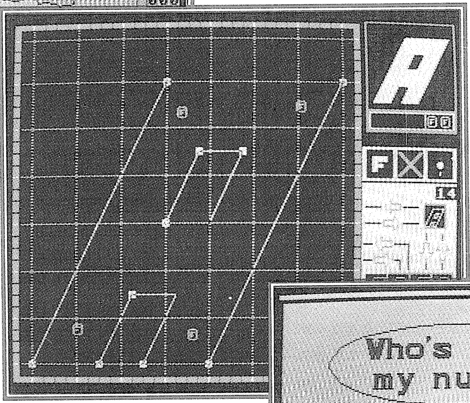
After booting up, a pretty title page is displayed before entering the WYSIWYG editing screen. The operations are controlled from the keyboard, with the additional facility to use either a mouse or trackerball. The editing screen shows only a portion of the banner, which can be scrolled horizontally as necessary.

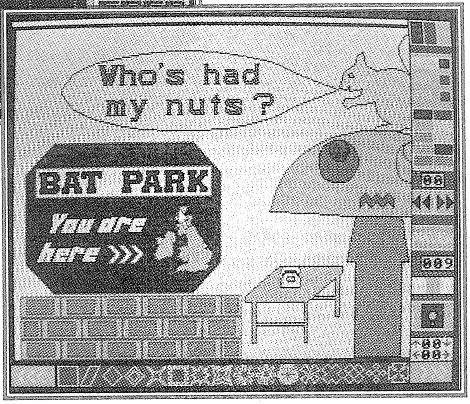


Characters are placed onto the banner by selecting the ones required from a display at the bottom of the screen, then dragging them to their desired position on the banner. The characters can be stretched both horizontally and vertically, rotated, and flipped before being placed.

To assist in locating characters, cross-lines can be displayed for accurate positioning. A preview facility can be called upon to show the banner in reduced form, so that you can see more easily how it is progressing.







positioned where required, with rubber-banding aiding the process. Once created, fonts can be saved to disc for later retrieval.

In use, Stretch is an extremely friendly package. The manual is printed in full colour, containing samples of all the fonts available and printouts to show what can be achieved.

Most operations of Stretch can be mastered in a matter of minutes, thanks to the clear presentation of on-screen information and icons. One slightly annoying feature is that the character scaling is shown on screen, but you can't simply move the pointer over it and click a button to change the values; the keyboard has to be used.

All in all, an impressive piece of software.

Additionally, block move and delete operations can be performed on the banner's contents as required.

Once a banner has been created, it can be saved to disc for later retrieval, which is achieved using a simple-to-use selection of those available.

When banners are printed, colours are represented as dither patterns by monochrome printers. On colour printers, they are reproduced in full colour. A number of print density options and scaling factors are available, depending upon the printer used.

A number of different fonts, 18 in all, are provided, with an editor to design new ones. The editor is extremely simple to use. Characters are composed of a number of vertices, defining the shape. These can be

Points Arising....Points Arising....Points Arising....Points Arising....

DECIMAL ARITHMETIC

(Volume 7 No.9 Page 28)

Last month's correction to the Decimal Arithmetic program failed to cope with all the possible problems. The author has informed us that a better solution is to change line 1170 to:

```
1170 STX offset:TXA:ASL A:STA shift_flag
```

leaving line 1160 as in the original listing.

Using the DOS+ PATH command

by Robin Burton

We've covered a lot of varied territory in the twelve months since the first Forum, so I thought it was about time that we got back to basics for a change.

REVISION

Those who have read the Forum since its inception in July last year will remember that our initial topics were batch files and the memory disc. Since then there hasn't been much about standard DOS+ facilities so we'll put that right now.

We'll start with discs. You're probably quite familiar with most of the facilities, but even so you may not use them all to their full advantage. The trouble with some of them is that they tend to be used when you set up a new application and are then forgotten.

I was reminded by an accident this morning, while I was preparing to write the Forum, so I'll tell you the story. My micro, disc drives etc. are in a PC style case with a separate keyboard and one of the extras is a cooling fan. Last week I had to partly dismantle the micro to change some ROMs, needed for a talk I was to give. As micro cooling fans are remarkably good at collecting and compacting airborne dust, at the same time I cleaned the fan. Obviously I missed a bit. Today that bit moved and landed on my DOS+ word processor disc, ruining it by scratching a visible groove round 60% of track zero.

I heard it immediately (an unpleasant repeated scratching) but it was too late for the disc. Fortunately the drive heads were OK but I was particularly sad because that disc was the first that I ever bought for my first drive in 1983. What do you mean? Of course I had a backup even if it wasn't quite the latest version!

A few 'fine tuning' tweaks were missing from some batch files, and three or four user-defined key functions from the word processor definitions file. It only took half an hour to bring it up to date, but along the way I was reminded of some of the things we tend to forget (e.g. "I'll back it up afterwards". I remembered today!).

'PATH'

Of course you know what a path is in disc filing systems, but do you exploit the DOS+ 'PATH' command to full advantage?

There's no excuse for neglecting 'PATH'; it's a permanent command, which means it's always immediately available. This is because it's built into the memory resident portion of DOS+, as opposed to transient commands like 'COLOUR' which must be loaded from disc each time. Initially 'PATH' isn't set. In effect it defaults to the root directory of the current drive, unless you change it.

It's usual to include the setting of 'PATH' in your 'AUTOEXEC.BAT' file, especially if you use a hard disc, but it can be altered at any time when you are in DOS command mode, either by another batch file or by a manually entered command.

The simplest form of the command is to enter 'PATH' with no parameters and it will then display the current setting. Hands up those who tried it and saw nothing but the normal prompt?

It's a usual and convenient practice to keep the most frequently used DOS utilities in their own directory on each application disc. Most people call these directories 'DOS' (which perhaps surprisingly is not a prohibited name) though 'DOSTOOLS', 'UTILS' or 'DOSUTILS' are also popular.

This technique is useful because it means your 'working' directory display won't be unnecessarily cluttered by the DOS commands. Of course it's also usual to have the disc containing this in drive A: (or C: for hard disc).

On the other hand if you're working in a directory called 'DOCFILES' in drive B: and wanted to change the attributes of a file called 'FORUM.DOC' to *read only*, if you issue it the hard way the command is:

```
    A:\DOSUTILS\FSET FORUM.DOC [RO]
```

which is not exactly short and convenient. This is especially important since the full pathname of the utility must be entered every time you use it if it's done this way.

It won't surprise you to hear that there's a short cut, and that it's 'PATH'. Using our example names if we had issued the command:

```
    PATH \DOSUTILS
```

while we were in the root directory, DOS would always search the 'DOSUTILS' directory for any command which it couldn't find in the current directory. It becomes irrelevant where our wanderings take us on the disc and how many directory levels down we may have gone. There's just one point to note here, and that is that in this example there's no drive identifier. This means that DOS will always expect to find 'DOSUTILS' on the current drive, whichever it happens to be at the time.

It can, in some circumstances, be useful to issue the 'PATH' command without a drive identifier, but more often than not you'll want to specify a particular drive too. This requires only the addition of the drive ID in the usual fashion, so for drive A: the above command becomes:

```
    PATH A:\DOSUTILS
```

When the command is issued like this, with a drive specified, it no longer matters what the current drive happens to be. The correct directory will always be searched for on the original drive.

Naturally, this means that you shouldn't change the disc referred to by 'PATH' unless 'PATH' is reset or the new disc also has a similar directory of the same name at the same level, in which case that will be used.

So far it's much like 'LIB' in ADFS, but actually it's more flexible and tolerant. For a start, as mentioned, even if you do change discs, so long as the new one contains the right directory at the right level there's no problem. Fortunately DOS+ doesn't have the annoying 'Disc changed' error. By sticking to a common naming convention on all your discs that's one problem you can completely eliminate.

Secondly, PATH will be used to search for any command that DOS doesn't recognise, so any 'CMD', 'COM', 'EXE' and 'BAT' files can all be accessed by this means from anywhere in the system.

MULTIPLE PATHS

Setting a path is obviously useful, but what happens if your 'UTILS' directory is on one drive and you also want a path on another, or you would like to keep your batch files in a directory of their own, separate from the DOS utilities? This is no problem. DOS permits multiple simultaneous paths to be declared, and they're just as easy to set up.

For illustration, suppose that we decided to keep each of the four executive file types in separate directories, all accessible from any other directory without explicit references. Using the directory names 'CMD', 'COM', 'EXE' and 'BAT' for each file type, the command would be:

```
    PATH \CMD;\COM;\EXE;\BAT
```

Notice that the only addition needed is that each of the paths we define is separated from the next by a semi-colon.

As it stands the above definition assumes that all these directories are on the current drive and that they are all just one level down from the root. In fact each path can be of a different length and/or can be on different drives - just supply the information as appropriate, remembering to separate all directory names in each path by a back-slash, like this:

```
PATH A:\DIR1\DIR2\DIR3;B:\DIR1A\DIR2A\DIR3A
```

POINTS TO WATCH

There is one restriction to be remembered, but in practice it's unlikely (or at least it ought to be) that you'll be troubled by it.

This is simply the total length of the command. In DOS, the maximum allowed for any single command line entry is 127 characters. This 'restriction' is because, on loading, 'COM' files have a header of &100 (256) bytes added at the front by DOS, and half of this (minus a byte) must be capable of holding the complete original command line as passed to the program from 'COMMAND.COM'.

This is done so that no matter what happens to the rest of the memory during execution, a program can always refer back to an unchanged (by anything else) copy of its original command parameters.

As a result of this restriction, if each of the pathnames you were to specify consisted of several full length directory names, you wouldn't be able to enter so many separate paths. Remember that the command itself takes five characters including the following space, and each full length directory name can take nine with its back-slash, plus another two for a drive ID for each path if this is needed too.

As I said, this shouldn't be a problem. If you commonly use more than two or three directory names in any one path I'd suggest the contents of your discs urgently need sorting out, re-structuring and simplifying.

There are three more points to remember when you use the PATH command. First, whenever you issue the command all previous paths are cancelled, so you can't just add an extra path. You must re-enter all previous paths, plus any new one(s), as a single command.

Second, no validation is carried out when you define your paths so care is needed when the command is used manually. A mis-typed directory name will NOT cause an error at the time the path is set, even if there's no such directory anywhere in the system. If you then issue a command which should be found in the (correct) path directory, but isn't, the error you'll get is the usual 'Bad command or filename'. There is no 'Can't find Path directory' or 'Bad Path' error. For example, try:

```
  PATH RUBBISH
```

and you can immediately confirm that it has been accepted. Follow this with an invalid command (e.g. HELLO). These errors can be very difficult to spot when a batch file fails some time later, or an unexpected 'Cannot load COMMAND.COM' pops up.

The last item concerns the way DOS works in looking for commands. The first directory examined is always the current one, the last is the root. In between, if multiple paths are set, each of them is searched in the order you specified them, so ensure that the most frequently used paths are defined first.

It's easy to practice with the path command if you're not completely sure yet. In this example we'll assume drive A: as the current drive, but the principle is the same no matter which drive you're using. You'll need a writable disc in drive B: because we'll be creating a new directory and putting a file in it. When you're ready enter the following, pressing Return after each line:

```
B:
MD BATCHDIR
CD BATCHDIR
COPY CON HELLO.BAT
ECHO OFF
ECHO Hello sailor!
<CTRL-Z>
A:
PATH B:\BATCHDIR
HELLO
```

The commands select drive B: and create a directory called 'BATCHDIR'. In this we create a file called 'HELLO.BAT' containing a message. Control-Z ends the file copy and drive A: is reselected. Finally the path is set and you'll see that the command 'HELLO', issued from drive A: causes our message to be displayed from the file in drive B:.

Repeat the exercise with another disc and then try swapping the two discs in drive B:. You'll see that DOS doesn't care which disc is used as long as the path names are the same. You can also then set the path with no drive ID. Put one disc in each drive and change drives to prove that PATH's drive can remain variable, even though the directory is fixed. Don't forget to tidy up afterwards. Delete the batch files first. You might also need to reset the path before you can remove the directories.

Next Forum we'll look at a few more under-used facilities, including floating drives.

BEEBUG READER SURVEY

We hope you will take this opportunity to express your views about BEEBUG. These will be vitally important to us in determining the future direction of the magazine and the services provided by BEEBUG. We appreciate that answering the questionnaire will take some thought and time, but we would ask you to answer as many questions as possible. Please return completed questionnaires by 4th August.

Where alternatives are listed, please select the most appropriate response(s). Opportunity is often provided for your own answers. Please use this whenever appropriate. Leave blank any questions you do not wish to answer.

I. PERSONAL DETAILS

1. Age: ☐ under 20 ☐ 20-40 ☐ 41-60 ☐ over 60

2. Occupation: ....

3. Which BBC micro do you use?

☐ Model B	☐ Model B+	☐ Master 128
☐ Compact	☐ 512 co-processor	☐ none

4. What are the main uses of your BBC micro:

☐ Hobby/Recreation	☐ Office work at home	☐ Business
☐ Education at home	☐ School/College/University	☐ Home/personal work
☐ Club/Society	☐ Other:	

....

5. What major items of software do you use (or intend to purchase in the next six months):

☐ Wordwise (Plus)	☐ ViewSheet	☐ SpellMaster	....
☐ View	☐ Alphabase	....	....
☐ InterWord	☐ Masterfile (II)	....	....
☐ InterSheet	☐ Interbase	....	....

6. What hardware items do you own (or intend to buy within the next six months) - give make and model (if known):

Modem	....	Shadow RAM	....
Floppy disc drive	....	Mono Printer	....
Hard disc	....	Colour Printer	....
Digitiser	....	Laser Printer	....
Sideways RAM	....	FAX system	....

7. Which computer languages do you use or are familiar with on your BBC micro?

☐ Basic ☐ 6502 Assembler ☐ C ☐ Pascal ☐ Forth

☐ Other..

8. Please assess your technical competence regarding the BBC micro:

☐ Beginner ☐ Moderately Competent ☐ Very Competent ☐ Expert

9. How long do you expect to go on using your current BBC micro?

☐ less than one year ☐ one to two years ☐ more than two years

10. If you were to replace your current machine, what would you most likely choose?

☐ Master/Compact ☐ Archimedes ☐ other manufacturer

11. What are likely to be the main uses of your BBC micro in the future (e.g. programming, word processing etc.)?..

....

II. BEEBUG MAGAZINE

12. What was your main reason for subscribing to BEEBUG?

....

13. Have your expectations been satisfied (Yes/No)?...................

If not, why not?...

14. Do you consider other magazines to offer better value for money (Yes/No)?......................

If so, which magazine(s)..
What is it in other magazines which makes you think this way?

....

15. From the last three issues of BEEBUG (May to July 1989), name up to six items which you consider to be the most interesting or useful.

....

....

....

16. From the same last three issues name up to six items which you consider to be the least interesting or useful.

....

....

....

17. Please score the following types of article for interest and usefulness on a scale of 1 (lowest) to 5 (highest)

❐ Beginners' Basic	❐ Comms	❐ Educational Computing
❐ Assembler Workshop	❐ Basic Workshop	❐ Other Languages
❐ Applications	❐ Product Reviews	❐ Utilities
❐ Hints & Tips	❐ Graphics	❐ Sound
❐ Games	❐ Postbag	❐ News
❐ Business Computing	❐ Competitions	❐ Hardware Projects

Other (please specify)..

18. Do you have any other comments on the content of BEEBUG?

....

....

19. If you find BEEBUG program listings difficult to read, do you have any suggestions to improve this? ....

....

20. How do you rate the style of presentation of the whole magazine?

❐ Very good ❐ Good ❐ Poor ❐ Very poor

21. What is it about the style of presentation of the magazine you particularly like/dislike?

....

....

22. In general do you find the content of the magazine

❐ Too technical ❐ Just right ❐ Not technical enough

III. BEEBUG IMAGE

23. Which three of the following words best describe BEEBUG to you:

❐ friendly	❐ informative	❐ authoritative	❐ condescending
❐ useful	❐ patronising	❐ interesting	❐ difficult
❐ serious	❐ exciting	❐ light	❐ trivial
❐ daunting	❐ boring	❐ instructive	

24. Assuming that the number of articles, reviews etc. remains the same, do you think a change from A5 to A4 format would improve the magazine.

❐ Yes ❐ No ❐ Wouldn't matter

Please elaborate further if you wish..

....

25. Would you prefer more tutorial and explanatory articles in the magazine if this meant reducing the number of program listings? ....

26. Do you believe that the magazine should provide more of a forum for a variety of short contributions from readers, even if this was partly at the expense of more professionally written articles and programs?

....

....

27. What three changes to BEEBUG would make you more likely to renew your subscription when it becomes due...

....

IV. MISCELLANEOUS

28. Do you feel that the level of service from our Technical Support Department is sufficient, bearing in mind that it is a free service?

Yes ☐ No ☐ No opinion ☐

29. Would you like us to offer an additional "priority hotline" service with *guaranteed turnaround time* on queries on the basis of a separate annual subscription?

Yes ☐ No ☐ No opinion ☐

30. Is the technical advice service a *major* reason for your subscribing to BEEBUG?

Yes ☐ No ☐ No opinion ☐

31. If you have purchased products from us within the last three months:
Were you satisfied with the service?

Yes ☐ No ☐ No opinion ☐

If you were not satisfied what was the.reason? ....

....

Do we stock all of the products that you require?

Yes ☐ No ☐ No opinion ☐

If not please specify...

....

32. When dealing with BEEBUG:
Do you find our staff generally polite and helpful?..................
Are responses to your enquiries dealt with efficiently? If not please specify...............

....

33. Do you have any comments on BEEBUG Magazine or BEEBUG as an organisation?

....

....

....

Thank you for taking the time to answer these questions.

Nimble Typer

Save your keyboard, fingers and sanity with this abbreviation substitution utility from William Tunstall-Pedoe.

I am sure that when most people type in a Basic program, they make good use of Basic's ability to expand abbreviated keywords, for example P. for PRINT. The situation is similar for operating system commands (when did you last type *CAT instead of *.?). However, when it comes to entering plain text, say in a word processor, there is not normally any alternative to typing everything in full. The program presented here corrects this by allowing user-defined abbreviations to be expanded as text is being typed, regardless of the particular application in use. For example, you could set up the abbreviation DS to present the phrase 'Dear Sir or Madam'. Then, whenever you were writing a letter, typing DS would enter the above phrase.

ENTERING THE PROGRAM

The program is supplied in two parts. The first of these, given in listing 1, assembles the machine code which performs the abbreviation substitution. This should be typed in and saved using a filename other than 'Nimble'. When run, the program will assemble the machine code and save it in a file called 'Nimble'.

The second part of the program is the editor which allows you to define abbreviations, and this is given in listing 2. Again, this should be typed in and saved, using a name such as 'NimEdit'.

USING THE PROGRAMS

The first stage in using the system is to define one or more sets of abbreviations. This is done by running the editor program. You are first asked whether you wish to modify an existing file, or create a new one. Initially, choose the latter. When modifying a file, you are asked to enter the filename of the file to be altered. You are then presented with a list in which each item consists of a word or phrase, and the abbreviation assigned to it. Each item is numbered, and at the end of the list is a blank item, which will be the only one present in a new file. You are then prompted to enter the number of an item. To edit an existing item, enter its number, while to create a new entry enter the number of the blank item. Pressing Return on its own at this point will quit the editor, prompting for a filename under which to save the abbreviation list. When entering or editing an entry, you are first prompted for the word or phrase, and then for the abbreviation to represent it, up to a maximum of five characters. To delete an existing entry, select it and then just press Return for the phrase and abbreviation.

You need to be fairly careful when choosing abbreviations. On one hand, you want to make them easily remembered, which is best done by making them similar to the word or phrase they are abbreviating. On the other hand, you don't want the program to mistake other words. For example, if you abbreviated 'Received' to 'RE', then if you tried to enter 'Recall', it would appear as 'Receivedcall'. One way around this is to prefix all abbreviations with a letter such as 'Q'. Incidentally, an abbreviation will only be expanded if it occurs at the start of a word.

To use the abbreviations, type *NIMBLE to load and install the machine code. This will in turn prompt for the name of a file created with the editor. After this, the machine will appear to behave normally, except that whenever one of the abbreviations is entered, it will be expanded to the corresponding word or phrase. This will occur whether you are in an application, or at the Basic prompt. If you wish to change the abbreviation file at any time, simply type *LOAD <file> where *file* is the new filename. The program will remain active until the machine is turned off. Alternatively, typing *FX247 followed by Break will disable it.

Listing 1

```
   10 REM Program Nimble Typer
   20 REM Version B 1.0
   30 REM Author  William Tunstall-Pedoe
   40 REM BEEBUG  July 1989
   50 REM Program Subject to Copyright
   60 :
  100 abb_buff_len=5
  110 max_filename=10
  120 osbyte=&FFF4 : osasci=&FFE3
  130 osword=&FFF1 : osfile=&FFDD
  140 osnewl=&FFE7 : nvoswrch=&FFC8
  150 RDCHV=&210
  160 :
  170 base=&900
  180 DIM G% 256*2
  190 FOR PASS=4 TO 7 STEP 3
  200 O%=G%
  210 P%=base
  220 [OPT PASS
  230 :
  240 \ Start of the code
  250 .intercept
  260 \ This intercept happens after
  270 \ the call.
  280 JSR nvoswrch
  290 PHP:STA tmp:PHA:TXA:PHA:TYA:PHA
  300 \ tmp contains the chr typed.
  310 LDA tmp
  320 LDX skipper:BNE s_failed
  330 \ Reset pointers if chr not
  340 \ alphanumeric.
  350 CMP#65:BCC failed
  360 CMP#128:BCS failed
  370 CMP#127:BNE i_over
  380 \ If delete pressed.
  390 LDX buff_ptr:BEQ exit
  400 DEC buff_ptr
  410 JMP exit
  420 .i_over
  430 \ Add character to buffer.
  440 AND#(255-32) : \ Upper-case it.
  450 LDY buff_ptr:STA abb_buff,Y
  460 INY:CPY#abb_buff_len:BCC over1
  470 LDY#0
  480 .over1
  490 STY buff_ptr
  500 BNE match
  510 .exit
  520 \ Return back to calling program.
  530 PLA:TAY:PLA:TAX:PLA:PLP
  540 RTS
  550 :
  560 \ Try to match abbreviation
  570 .match
  580 LDX buff_ptr:BEQ failed
  590 LDY#0:LDX#0
  600 \ Y points to list of abbrs
  610 \ X points to chr in abbr buffer.
  620 .loop1
  630 LDA store,Y:STA tmp
  640 LDA abb_buff,X
  650 CMP tmp:BNE skip
  660 INX
  670 INY:BEQ exit
  680 \ '#' terminates each abbr.
  690 \ If reached they match.
  700 LDA store,Y:CMP#ASC"#":BEQ success
  710 CPX buff_ptr:BCS exit
  720 BCC loop1
  730 .s_failed
  740 DEC skipper
  750 .failed
  760 LDA#0:STA buff_ptr
  770 JMP exit
  780 .skip
  790 \ Skip this abbreviation and try
  800 \ to match the next.
  810 LDX#2
  820 .s_loop
  830 LDA store,Y
  840 CMP#ASC"#":BEQ sl_over
  850 INY:BEQ exit:BNE s_loop
  860 .sl_over
  870 \ If end of list reached, exit.
  880 INY:BEQ exit
  890 DEX:BNE s_loop
  900 BEQ loop1
  910 :
  920 \ Successful match
  930 .success
  940 INY:STY tmp
  950 .s_loop
  960 \ Insert deletes to remove the
  970 \ abbreviation.
  980 LDA#&8A:LDX#0:LDY#127:JSR osbyte
  990 INC skipper
 1000 DEC buff_ptr:BNE s_loop
 1010 \Now insert the actual text.
 1020 .si_loop
 1030 LDY tmp
 1040 LDA store,Y:CMP#ASC"#":BEQ failed
 1050 TAY:LDA#&8A:JSR osbyte
 1060 INC skipper
 1070 INC tmp:BNE si_loop
 1080 JMP failed
 1090 \ Temporary byte store.
 1100 .tmp EQUB 0
 1110 :
 1120 \ Allows code to ignore its own
 1130 \ insertions. This is not 100%
 1140 \ necessary but allows for
 1150 \ example 'ESP' to abbreviate
 1160 \ 'especially' without infinite
 1170 \ loop."
 1180 \ This byte stores the number of
 1190 \ chrs to ignore.
 1200 .skipper EQUB 0
 1210 :
 1220 .buff_ptr EQUB 0
 1230 .abb_buff EQUS STRING$(abb_buff_le
n,CHR$0)
 1240 .of_blk
 1250 EQUW filename
 1260 EQUD store
 1270 EQUD 0
 1280 EQUD 0:EQUD 0
 1290 .return
 1300 JSR osfile
 1310 .set_up_vec
 1320 LDA#intercept MOD 256:STA RDCHV
 1330 LDA#intercept DIV 256:STA RDCHV+1
 1340 .quit
 1350 RTS
 1360 ]
 1370 IF (O%-G%)>256 THEN PRINT "First b
it of prog bigger than 256 bytes.":STOP
 1380 NP%=(P%+256) AND &FF00
 1390 REM All the above code must fit
 1400 REM in one page. This gives the
 1410 REM number of bytes remaining.
 1420 IF PASS=6 THEN PRINT "Code Space l
eft:";(NP%-P%);" bytes."
 1430 O%=O%+(NP%-P%):P%=NP%
 1440 [OPT PASS
 1450 .store
 1460 \ Where Execution Starts
 1470 \ All this code gets overwritten
 1480 \ when the abbr file is loaded.
 1490 \ All code from here on must not
 1500 \ exceed 256 bytes.
 1510 .start
 1520 LDX#0
 1530 .o_loop
 1540 LDA prompt,X:BEQ o_over
 1550 JSR osasci
 1560 INX : BNE o_loop
 1570 .o_over
 1580 LDA#0:LDX#rd_blk MOD 256:LDY#rd_bl
k DIV 256:JSR osword
 1590 BCC o_over2
 1600 BRK : EQUB &11 : EQUS "Escape"+CHR
$0
 1610 .o_over2
 1620 \Set up BREAK key intercept
 1630 LDA#&F8:LDX#set_up_vec MOD 256:LDY
#0
 1640 JSR osbyte
 1650 LDA#&F9:LDX#set_up_vec DIV 256:LDY
#0
 1660 JSR osbyte
 1670 LDA#&F7:LDX#&4C:LDY#0
 1680 JSR osbyte
 1690 LDA#&FF
 1700 LDX#of_blk MOD 256:LDY#of_blk DIV
256
 1710 JMP return
 1720 :
 1730 .prompt EQUS "file:"+CHR$0
 1740 .rd_blk
 1750 EQUW filename
 1760 EQUB max_filename
 1770 EQUB 33:EQUB ASC"z"
 1780 .filename
 1790 EQUS STRING$(max_filename,CHR$13)
 1800 ]NEXT
 1810 :
 1820 REM Save the code
 1830 S$="*SAVE Nimble "+STR$~G%+" "+STR
$~O%+" "+STR$~start+" "+STR$~base
 1840 OSCLI S$
```

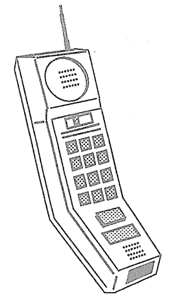
Listing 2

```
   10 REM Program Abreviation Editor
   20 REM Version B 1.0
   30 REM Author  William Tunstall-Pedoe
   40 REM BEEBUG  July 1989
   50 REM Program Subject to Copyright
   60 :
  100 MODE 7
  110 base=&A00:P%=base
  120 FOR A%=P% TO P%+255:?A%=0:NEXT
  130 num%=0
  140 DIM abbr$(100),text$(100)
  150 PROCheader
  160 PRINT TAB(0,10);
  170 PROCdouble(CHR$131+"1)"+CHR$134+"E
dit old file.",TRUE,FALSE)
  180 PROCdouble(CHR$131+"2)"+CHR$134+"C
reate new file.",TRUE,FALSE)
  190 PRINT 'CHR$134+"Please choose :"+C
HR$133;:INPUT C%
  200 IF C%=1 THEN PROCload
  210 IF C%>2 OR C%<1 THEN RUN
  220 A%=1
  230 REPEAT
  240 PROCdisplay
  250 PRINT TAB(0,21);
  260 N$="":PRINTCHR$134"Item:"CHR$133;
  270 INPUT N$
  280 N%=VAL(N$)
  290 IF N%>(num%+1) THEN N%=0
  300 IF N%<1 THEN N%=0
  310 IF N%>num% THEN num%=N%
  320 IF N%=0 THEN 440
  330 PRINTCHR$134"Text:"CHR$133;
  340 INPUT LINE text$(N%)
  350 PRINTCHR$134"Abbr:"CHR$133;
  360 INPUT LINE abbr$(N%)
  370 abbr$(N%)=FNupper(abbr$(N%))
  380 IF LEN(abbr$(N%))=0 THEN 430
  390 A%=1
  400 c%=ASC(MID$(abbr$(N%),A%,1))
  410 IF c%<65 OR c%>90 THEN VDU 7,11,13
:GOTO 350
  420 A%=A%+1:IF A%<=LEN(abbr$(N%)) THEN
 400
  430 PROCcompress
  440 UNTIL N%=0 AND N$="" AND sum%<256
  450 PRINTCHR$134"Filename:"CHR$133;
  460 INPUT f$
  470 FOR A%=1 TO num%
  480 PROCoutput(abbr$(A%))
  490 PROCoutput(text$(A%))
  500 NEXT
  510 OSCLI("SAVE "+f$+" "+STR$~base+" +
100")
  520 END
 1000 DEFPROCload
 1010 PRINTCHR$134"Filename:"CHR$133;
 1020 INPUT f$
 1030 H%=OPENIN(f$)
 1040 N%=0
 1050 REPEAT
 1060 N%=N%+1
 1070 abbr$(N%)=FNread(H%)
 1080 text$(N%)=FNread(H%)
 1090 UNTIL EOF#H%
 1100 CLOSE#H%
 1110 num%=N%-1
 1120 ENDPROC
 1130 DEFPROCdisplay
 1140 s%=0
 1150 PROCheader
 1160 PRINT
 1170 PRINTCHR$134;TAB(4);"Text";TAB(32)
;"Abbr"
 1180 FOR A%=1 TO (num%+1)
 1190 PRINT ;CHR$134A%;CHR$131;TAB(4);te
xt$(A%);TAB(32);abbr$(A%)
 1200 s%=s%+LEN(text$(A%))+1+LEN(abbr$(A
%))+1
 1210 NEXT
 1220 sum%=s%
 1230 PRINTCHR$134"(Space Used:"CHR$131;
(INT((sum%/256)*1000+0.5))/10;CHR$134"%)
"
 1240 ENDPROC
```

```
 1250 DEFPROCoutput(s$)
 1260 LOCAL A%
 1270 FOR A%=P% TO (P%+LEN(s$)-1)
 1280 ?A%=ASC(MID$(s$,(A%-P%+1),1))
 1290 NEXT
 1300 ?A%=ASC"#"
 1310 P%=P%+LEN(s$)+1
 1320 ENDPROC
 1330 DEFFNread(h%)
 1340 LOCAL s$
 1350 IF EOF#h% THEN =""
 1360 s$=""
 1370 REPEAT
 1380 c%=BGET#h%
 1390 IF c%<>ASC"#" AND c%<>0 THEN s$=s$
+CHR$(c%)
 1400 UNTIL c%=ASC"#" OR EOF#h%
 1410 =s$
 1420 DEFPROCcompress
 1430 LOCAL A%
 1440 FOR A%=1 TO num%
 1450 IF NOT(text$(A%)="" AND abbr$(A%)=
"") OR A%>num% THEN 1530
 1460 IF A%=num% THEN num%=num%-1:GOTO 1
530
 1470 FOR B%=A% TO num%-1
 1480 text$(B%)=text$(B%+1)
 1490 abbr$(B%)=abbr$(B%+1)
 1500 NEXT
 1510 text$(num%)="":abbr$(num%)=""
 1520 num%=num%-1
 1530 NEXT
 1540 ENDPROC
 1550 DEFPROCdouble(s$,double%,centre%)
 1560 IF double% THEN s$=CHR$141+s$
 1570 IF centre% THEN s$=STRING$(INT(17-
(LEN(s$)/2))," ")+s$
 1580 PRINT s$
 1590 IF double% THEN PRINT s$
 1600 ENDPROC
 1610 DEFPROCheader
 1620 CLS
 1630 PRINT CHR$141CHR$131"         Nimbl
e Editor"
 1640 PRINT CHR$141CHR$134"         Nimbl
e Editor"
 1650 ENDPROC
 1660 DEFFNupper(s$)
 1670 IF s$="" THEN =""
 1680 LOCAL f$,z%
 1690 f$=""
 1700 FOR z%=1 TO LEN(s$)
 1710 f$=f$+CHR$(ASC(MID$(s$,z%,1)) AND
(255-32))
 1720 NEXT
 1730 =f$
```

The Comms Spot

by Peter Rochford



We've talked briefly in the Comms Spot before about bulletin boards and taken an in-depth look at a few of the viewdata ones around. But what about the scrolling text types that abound, often referred to as a BBS (Bulletin Board System)? A look at the latest list that we have published will show that there are now a large number of these in the UK, and I have to admit that our list is not totally comprehensive. There are many more besides.

The whole bulletin board scene originated in the USA, where the first one sprung up on a university mainframe computer as a means for students to leave messages for each other. From that beginning they have proliferated, and there are literally thousands of BBS at the present time in the USA, and the idea has spread worldwide. Most of the pioneering work in terms of the software that drives them has been done in the States, and you will find the majority of UK BBS using software of American origin.

A BBS consists of a microcomputer with a large amount of mass storage usually in the form of a hard disc, an auto-answer modem capable of working at a number of baud rates and the controlling software. The system, once set up and running, can be left unattended for many days, until the system operator (SYSOP) does a periodic check that all is OK, performs a certain amount of general 'housekeeping' to it, and possibly provides some updated information to be displayed.

So, what is the attraction of these boards and what do they offer those who log-on? Well the main function of the bulletin board is a means of communication with other people of similar interests, exchange of information and ideas and exchange of computer software. In many cases the use of the BB is free to anyone who wishes to log-on, but quite a few are now asking for subscriptions, and those who don't normally require you to add something to the board in terms of either software or regular use of the general messaging areas. Sadly, too many people in the past have just used BBS for the downloading of the free software and nothing more. This makes for a boring bulletin board.

You can use a BB as a place to leave messages for others to read, either on a general or a personal basis. As a means of personal communication this is only practical if the person you want to send a message to is a user of the same board. The other big drawback is that most BBS are on a single telephone number in one part of the country and therefore telephone costs will be high if you have to dial long distance. This is unlike, say, Prestel or Telecom Gold where a local call is all that is needed wherever you live.

Every BB usually has a general messaging area where you can leave notes on a variety of subjects for others to read and comment on, either publicly or by sending you a personal reply. Along with this area there are always a number of special interest group (SIG) areas. Quite often these are devoted to computing, and a lot of boards you will find cater for the BBC amongst others. This is the place to exchange ideas and to ask for help from others if you have a particular computing problem.

There are number of boards now that are devoted to other subjects of interest, with topics as diverse as organic gardening or astronomy.

Apart from messaging and conferencing facilities most BBS provide files for downloading. This is usually public domain or shareware software that costs nothing to download. A variety of error-checked file transfer options are offered, including Xmodem, Ymodem, SEALINK and Kermit. For most users Xmodem will suffice, and all of the competent comms packages around are normally equipped with Xmodem.

I have to say at this point with reference to downloadable software, that sadly few of the boards that I have come across recently offer much in the way of BBC files to download. Most of what is on offer is for PCs and normally originates from the USA where the PC has become more common as a home machine. Users of the Master 512 will find plenty, of course, to please them, and can easily transfer the downloaded file to a DOS disc using the Getfile utility provided. Be advised though that the files available for downloading are normally in a compressed or archived form, and you will need to download a special program to expand them before use.

Many people I have spoken to are daunted with the prospect of logging on to a scrolling text BB. Don't be! You are missing out on a fascinating area of comms. Yes, it is a lot different to going on Prestel but it can be a lot more fun.

When you first log on to a BB you will usually be asked for your first and last names. Don't give false ones, it's pointless and means you will have to remember them next time you log on.

Next, you will be taken through a once-off registration routine and be asked some questions about the computer or terminal software you are using. This usually means what number of lines per screen your terminal supports, if you need line feeds and possibly a few other questions. This is all pretty straightforward and takes but a few minutes. Then you will be asked to select a password for future use on log-on. Once all this has been done the details are logged to disc and the settings you asked for will be used each time you log on.

Having got this far, you will probably see a general message from the SYSOP detailing any new changes to the board or announcing any special items of interest that have been added. After this, you should get the main menu page from which you can select any area of the board you want to look at by a single keypress. You should find that most boards have a help section for new users, and it is definitely worth looking at this and saving it to disc for printing out later when offline.

So that's really it. Do log on to a few BBS and explore them. They can be very interesting and rewarding to use. Do observe a few rules when on line though. Don't hog the board for too long. Most are on one line and you will prevent others from getting on. As I said before, don't use false names as SYSOPs hate this and it causes them a lot of grief. Remember most boards are free and provided by enthusiasts for general use by others. Make good use of the board and in particular the messaging areas. They can be a great source of help and information and a good way of striking up new friendships with like-minded people.

Running Four Temperatures (continued from page 14)

```
 1970 IF probe%=2 THEN tempsum=tempsum+F
Ntemp2:N%=N%+1
 1980 IF probe%=3 THEN tempsum=tempsum+F
Ntemp3:N%=N%+1
 1990 IF probe%=4 THEN tempsum=tempsum+F
Ntemp4:N%=N%+1
 2000 UNTIL N%=5
 2010 temp(probe%)=tempsum/N%
 2020 IF probe%=1 THEN PROCplot(5)
 2030 IF probe%=2 THEN PROCplot(21)
 2040 IF probe%=3 OR probe%=4 THEN PROCp
lot(69)
 2050 tt%=TIME-t%+tt%
 2060 NEXT probe%
 2070 N%=0:tempsum=0:time%=TIME:calctime
%=tt%
 2080 UNTIL TIME/100>tspan:MOVE 1230,938
:PRINT"*"
 2090 ENDPROC
 2100 :
 2110 DEF FNline(T)
 2120 IF T>=4320000 THEN =604800
 2130 IF T>=432000 THEN =86400
 2140 IF T>=86400 THEN =21600
 2150 IF T>=21600 THEN =3600
 2160 IF T>=3600 THEN =900
 2170 IF T>=900 THEN =300
 2180 IF T>=300 THEN =30
 2190 IF T>=120 THEN =10
 2200 IF T>=30 THEN =5
 2210 IF T>=10 THEN =1
 2220 =1
 2230 :
 2240 DEF PROCplot(line%)
 2250 PLOTline%, FNx(TIME/100),FNy(temp(
probe%))
 2255 REM See text for lines 2260-2270 f
or Master
 2260 IF probe%=3 THEN PROCx
 2270 IF probe%=4 THEN PROCplus
 2280 ENDPROC
 2290 :
 2295 REM Omit lines 2300-2380 for Maste
r
 2300 DEF PROCx
 2310 PLOT0,0,0:PLOT1,-12,-12
 2320 PLOT 0,0,12:PLOT1,12,-12
 2330 ENDPROC
 2340 :
 2350 DEFPROCplus
 2360 PLOT0,0,4:PLOT1,0,-8
 2370 PLOT0,4,4:PLOT1,-8,0
 2380 ENDPROC
 2390 :
```

Spin a Disc (4)

David Spencer presents a high speed ADFS disc formatter which utilises the disc ROM from last month.

This month I will use the disc access ROM which was given last month to implement a high-speed ADFS disc formatter. First off though, humble apologises for a number of bugs in the program given last month. The first of these meant that the format command sent to the 1770 was incorrect on all but the first two tracks!. Secondly, the OSWORD call chosen (115) is already used by ADFS for a different purpose. To get around this, we will change our call to have the number 116 (&74). The third and final problem is much more subtle and interesting. Occasionally (about one time in ten), the 1770 fails to issue an interrupt at the end of a formatting command. This means that the 'command done' flag never gets set, and the program effectively hangs up. To get round this it is necessary to make the format command a special case, and check the actual 1770 status register to determine when it is completed. Listing 1 gives all the necessary modifications to last month's program. After making these changes, the program should be run to save a new version of the ROM image.

THE FORMATTER PROGRAM

To use the formatter, type in listing 2 and save it. Before running the program you must install the Disc ROM into sideways RAM. When run, the program prompts for a drive number and a size (S, M or L). After confirmation, the disc is formatted and verified, with any error being reported.

HOW IT WORKS

Formatting a disc is in fact a two stage process. The first task is to lay down the necessary marks on the disc to split each track into a number of sectors. This can be thought of as the hard formatting, and is done using the format track option of the disc ROM. The second operation involves writing to the disc the basic information needed by the filing system. For an ADFS disc this consists of the free space map and the root directory. This can be viewed as the soft format. We will look at soft formatting next month when we examine disc structure and give some tips on recovering lost files.

DISC LAYOUT

You will recall from the first part of this series that each track is split into sectors with a gap at the start of the track, a gap at the end, and a gap between each sector. In turn, each sector consists of an ID field and a data field, with another gap between the two. When formatting a disc with the 1770, we have to specify the exact data which makes up the track. This means not only the data within the sectors, but also the ID fields, and the gaps. Figure 1 shows the data that makes up each sector. However, this is not exactly the same as the data that we have to set up in memory. This is because you will remember from the first part of this series that certain bytes on the disc contain missing clock pulses. For this reason, the 1770 supports a number of special values which write the necessary combination of clock and data pulses. In figure 1, the values given in brackets indicate the actual values that must be passed to the 1770.

With this in mind, the process of formatting is reduced to setting up the entire data for one track in memory, and then calling the disc ROM to write the track to the disc using the 1770's write sector command. The track data is made up of an initial gap consisting of 60 bytes of

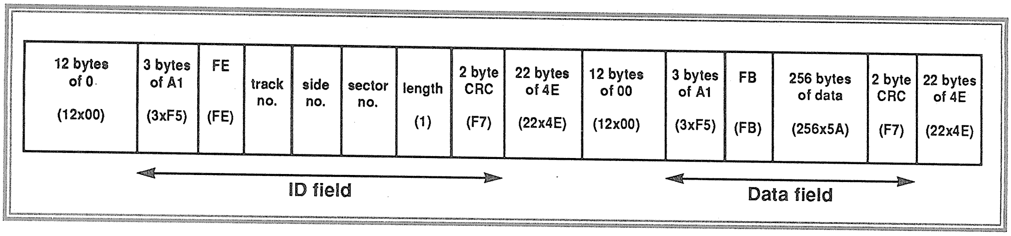


Figure 1. Format of a single ADFS sector

value &4E, followed by the data from figure 1 repeated sixteen times (once for each sector), and finally a trailing gap of around 700 bytes of &4E. The procedure PROCtdata in the formatter program sets up the basic data for a single track, and the routine PROCnxttrk inserts the correct track and sector numbers into this as formatting proceeds.

Finally, the best way to understand the working of the program is to follow it through procedure by procedure, with reference to the material from the first three articles in this series. Incidentally, an interesting (and not particularly hard) exercise would be to modify the program to format Archimedes D and E format discs. A slightly harder task is to write an IBM PC disc formatter.

Listing 1

```
 270 .osword LDA &EF:CMP #116
 590 .write LDY track:CPY #60:BCS read
 805 LDA #17:STA &A0
 825 CMP #&F0:BCS ffcomm
 851 .ffcomm LDA status:ROR A
 852 BCC ffcomm
 853 .ffcomm2 BIT &EF:BMI error3
 854 LDA status:ROR A:BCS ffcomm2
 855 RTS
```

Listing 2

```
   10 REM Program ADFS Formatter
   20 REM Version B1.0
   30 REM Author  David Spencer
   40 REM BEEBUG  July 1989
   50 REM Program Subject to Copyright
   60 :
  100 MODE 7:speed=0
  110 DIM block 100, ws &1C00
  120 REPEAT INPUT "Drive number "dr%
  130 UNTIL dr%>=0 AND dr%<=3
  140 REPEAT INPUT '"Size (S, M, L) "s$
  150 UNTIL INSTR("SML",s$)
  160 s%=&280*2^(INSTR("SML",s$)-1)
  170 PRINT'"Are you sure (Y or N)"
  180 A=GET AND &DF:IF A<>ASC"Y" END
  190 PROCformat(ws,block,s%,dr%)
  200 END
  210 :
 1000 DEF PROCformat(buffer,block,size,d
rive)
 1010 PROChardformat(buffer,block,size,d
rive)
 1020 PROCsoftformat(buffer,block,size,d
rive)
 1030 PROCverify(buffer,block,size,drive
)
 1040 ENDPROC
 1050 :
 1060 DEF PROChardformat(buffer,block,si
ze,drive)
 1070 LOCAL track,mtrack
 1080 PROCtdata(buffer)
 1090 IF size=&280 THEN mtrack=39 ELSE m
track=79
 1100 PRINT"Formatting ";
 1110 FOR track=0 TO mtrack
 1120 PRINT RIGHT$("0"+STR$track,2);CHR$
8;CHR$8;
 1130 PROCformtrk(buffer,0,track,block,d
rive)
 1140 IF size=&A00 PROCformtrk(buffer,1,
track,block,drive)
 1150 NEXT:PRINT"  "
 1160 ENDPROC
 1170 :
 1180 DEF PROCsoftformat(buffer,block,si
ze,drive)
 1190 PROCfsmap(buffer,size)
 1200 PROCroot(buffer+&200)
 1210 ?block=1
 1220 block!1=buffer
 1230 block?5=drive
 1240 block?6=0:block?7=0
 1250 block?8=7
 1260 PROCdcomm(block)
 1270 ENDPROC
 1280 :
 1290 DEF PROCverify(buffer,block,size,d
rive)
 1300 LOCAL track,mtrack
 1310 IF size=&280 THEN mtrack=39 ELSE m
track=79
 1320 PRINT"Verifying ";
 1330 FOR track=0 TO mtrack
 1340 PRINT RIGHT$("0"+STR$track,2);CHR$
8;CHR$8;
 1350 PROCvertrk(buffer,0,track,block,dr
ive)
 1360 IF size=&A00 PROCvertrk(buffer,1,t
rack,block,drive)
 1370 NEXT:PRINT"  "
 1380 ENDPROC
 1390 :
 1400 DEF PROCvertrk(buffer,side,track,b
lock,drive)
 1410 ?block=0:block!1=buffer
 1420 block?5=drive+side*4:block?6=track
 1430 block?7=0:block?8=16:block?9=speed
 1440 PROCdcomm(block)
 1450 ENDPROC
 1460 :
 1470 DEF PROCnxttrk(buffer,side,track)
 1480 buffer=buffer+76
 1490 FOR sector=0 TO 15
 1500 ?buffer=track:buffer?1=side
 1510 buffer?2=sector
 1520 buffer=buffer+340
 1530 NEXT
 1540 ENDPROC
 1550 :
 1560 DEF PROCformtrk(buffer,side,track,
block,drive)
 1570 PROCnxttrk(buffer,side,track)
 1580 ?block=2:block!1=buffer
 1590 block?5=4*side+drive
 1600 block?6=track:block?9=speed
 1610 IF track=0 THEN block?10=-1
 1620 PROCdcomm(block)
 1630 ENDPROC
 1640 :
 1650 DEF PROCfsmap(buffer,size)
 1660 LOCAL count
 1670 FOR count=0 TO &1FF STEP 4
 1680 buffer!count=0
 1690 NEXT
 1700 ?buffer=7:buffer!&FC=size
 1710 buffer?&FF=FNchecksum(buffer)
 1720 buffer!&100=size-7
 1730 buffer!&1FB=RND(65536)-1
 1740 buffer?&1FE=3
 1750 buffer?&1FF=FNchecksum(buffer+&100
)
 1760 ENDPROC
 1770 :
 1780 DEF PROCroot(buffer)
 1790 LOCAL count,temp
 1800 DIM temp 4
 1810 FOR count=0 TO &4FF STEP 4
 1820 buffer!count=0
 1830 NEXT
 1840 $temp="Hugo"
 1850 buffer!1=!temp
 1860 buffer!&4FB=!temp
 1870 buffer?&4CC=ASC"$"
 1880 buffer?&4D9=ASC"$"
 1890 buffer?&4D6=2
 1900 ENDPROC
 1910 :
 1920 DEF FNchecksum(ptr)
 1930 !&70=ptr-1
 1940 =USR checksum
 1950 :
 1960 DEF PROCdcomm(block)
 1970 X%=block MOD &100
 1980 Y%=block DIV &100
 1990 A%=116:CALL &FFF1
 2000 IF ?block<>255 THEN ENDPROC
 2010 IF block?1=17 END
 2020 IF block?1=&40 PRINT''"Write Prote
cted":END
 2030 PRINT''"Disc error ";~block?1;" at
 track ";block?6
 2040 END
 2050 :
 2060 DEF PROCtdata(buffer)
 2070 DIM code 200
 2080 FOR pass=0 TO 2 STEP 2
 2090 P%=code
 2100 [OPT pass
 2110 LDA #&4E:LDY #&59
 2120 .loop1 STA (&70),Y
 2130 DEY:BPL loop1
 2140 LDA #16:STA &72
 2150 LDA &70:CLC:ADC #60:STA &70
 2160 LDA &71:ADC #0:STA &71
 2170 .loop2 LDX #0:LDY #0
 2180 .loop3 LDA data,X:STA &73
 2190 BMI 13d:LDA data+1,X
 2200 .loop4 STA (&70),Y:JSR inc
 2210 DEC &73:BNE loop4
 2220 INX:INX:BNE loop3
 2230 .13d DEC &72:BNE loop2
 2240 LDX #3:LDA #&4E
 2250 .loop5 STA (&70),Y:JSR inc
 2260 DEC &72:BNE loop5:DEX:BNE loop5
 2270 RTS
 2280 :
 2290 .inc INC &70:BNE inc2:INC &71
 2300 .inc2 RTS
 2310 :
 2320 .data
 2330 EQUB 12:EQUB 0:EQUB 3:EQUB &F5
 2340 EQUB 1:EQUB &FE:EQUB 1:EQUB 0
 2350 EQUB 1:EQUB 0:EQUB 1:EQUB 0
 2360 EQUB 1:EQUB 1:EQUB 1:EQUB &F7
 2370 EQUB 22:EQUB &4E:EQUB 12:EQUB 0
 2380 EQUB 3:EQUB &F5:EQUB 1:EQUB &FB
 2390 EQUB 0:EQUB &5A:EQUB 1:EQUB &F7
 2400 EQUB 24:EQUB &4E:EQUB &FF
 2410 :
 2420 .checksum LDA #&FF:LDY #&FF:CLC
 2430 .check2 STA &72:LDA (&70),Y
 2440 ADC &72:DEY:BNE check2:RTS
 2450 ]NEXT
 2460 !&70=buffer:CALL code
 2470 ENDPROC
```

Batch Byte Loading

Lance Vick shows how you can speed up programs that perform large amounts of file handling.

Programs which operate on large data files can frequently be slowed down dramatically by the time taken to read bytes of data from the file. To understand why this is, and how the situation can be improved, we need to look at how the filing systems access files. Data is stored on disc in chunks known as *sectors*. These sectors, which are 256 bytes long for both DFS and ADFS, are indivisible, and must be read in one operation. Therefore, when a file is opened and a byte read from it, the filing system in fact reads in 256 bytes. When a second byte is read, its value is already in memory, and so no further disc access is necessary. This is true for all of the first 256 bytes. However, when the 257th byte is read, the filing system must read in another complete sector.This process is called *buffering*.

A problem arises when the program that is reading the data (using the Basic function BGET#) takes a relatively long time to process the data. In this case, it is likely that by the time the filing system needs to read in a new sector, the disc drive has stopped spinning from the previous read. This means that the disc drive is stopped and re-started between each sector, and this is slow, noisy and unkind to the drive. One way around this problem is to read in as much data as is possible at a time. Therefore, instead of using the 256-byte chunks imposed by the sector length, try and read in several kilobytes in one go. In this way, the disc drive has to start and stop less, and the whole operation is faster and quieter.

Unfortunately, the filing systems do not provide a direct method of implementing a larger file buffer, but it is not too hard to do it ourselves by adding some simple routines to any program that reads data from a file. The following set of functions and procedures do just that.

```
1000 DEF FNbget
1005 F%=count% MOD max%
1010 IF F%=0 PROCload_bytes
1020 count%=count%+1
1030 =storage%?F%
1040:
1100 DEF PROCload_bytes
1110 A%=4
1115 REM OSGBPB-read bytes from current
position
1120 X%=block%MOD256
1125 REM LSB of parameter block address
1130 Y%=block%DIV256
1135 REM MSB of parameter block address
1140 ?block%=ch%:REM file handle
1150 block%!1=storage%
1155 REM where to store bytes read in
1160 number_left%=extent%-count%
1170IFnumber_left%>max%:amount%=max%:
ELSEamount%=number_left%
1180 block%!5=amount%
1185 REM number of bytes to be loaded
1190 CALL&FFD1
1200 ENDPROC
1210:
1220 DEF PROCsetupbuffer
1230 max%=&1000
1240 DIM storage% max%, block% 12
1250 extent%=EXT#ch%
1260 count%=0
1270 ENDPROC
```

The operation of this program fragment is relatively straightforward. The procedure *PROCsetupbuffer* reserves as large a block of memory as possible to act as a buffer, and sets up the variables *count%* as a pointer into the file, and *extent%* as the length of the file. The function *FNbget* then provides the equivalent of Basic's BGET function, except that data is read from our buffer, rather than directly from the filing system. If the desired byte is in memory then it is simply read from the buffer and returned. However, *if the byte is the first in the* buffer, then the function deduces that the buffer must first be loaded from disc, and calls the procedure *PROCload_bytes* to do this.

PROCload_bytes performs the function of loading our extended buffer from disc. This can be done by using the filing system routine OSGBPB which is covered in the Advanced User Guide, and which was also dealt with in part 11 of the *File Handling For All* series in BEEBUG Vol.8 No.1. The basic purpose of this call is to read and write blocks of data rather than single bytes (OSGBPB stands for Operating System Get Bytes - Put Bytes). By reading whole blocks in one go the filing system can optimise the process, thereby making it much faster than reading each byte individually. This is particularly true when using Econet on a model B, in which case OSGBPB is about twenty times faster than the equivalent series of calls to OSBGET.

USING THE ROUTINES IN YOUR OWN PROGRAMS

Incorporating the above routines into your own programs will usually be quite easy. An obvious requirement is that there is enough free memory to act as a buffer, because if a small buffer is used there will be little advantage. The best way to find out the available memory is to run the program in question, stop it and type:

```
   PRINT ~HIMEM-(!2 AND &FFFF)
```

This will print in hex the free memory. you should then round the value down by replacing the last two digits with zero (so for example, &1245 would become &1200), and then subtract &200 to allow for the extra routines and the Basic stack. Therefore, the value of &1200 would become &1000.

You can now add the two procedures and the function to the end of your program, substituting the value you have just calculated into line 1230. You should then modify your program by changing all occurrences of BGET to FNbget, and adding a call to *PROCsetupbuffer* immediately after the file has been opened. If the program uses the function EOF, it will also be necessary to change this to (count%=extent%).

To illustrate the changes needed, consider the following very simple program.

```
  10 ch%=OPENIN"Testfile"
  20 REPEAT
  30 PRINT BGET#ch%
  40 UNTIL EOF#ch%
  50 CLOSE#ch%
```

Using the buffering routines, this would need to be changed to:

```
  10 ch%=OPENIN"Testfile"
  15 PROCsetupbuffer
  20 REPEAT
  30 PRINT FNbget
  40 UNTIL (count%=extent%)
  50 CLOSE#ch%
```

Incidentally, the above example gains nothing from the extra buffering, because the disc will still be rotating when a new sector is read in, and the operation will not therefore be slowed down. However, had we performed some complex operation on each byte, rather than just printing its value, then the saving in execution time would be considerable.

EXTENDING THE TECHNIQUE

So far, we have only considered reading files. However, our buffering technique can be extended to writing or updating of files. For writing, we must build up a buffer in memory of all the data to be written to the file. When this buffer is full we write it out to disc using OSGBPB, and repeat the process for the next buffer full. For updating files (reading and writing), the situation is more involved, as we need to read the data from disc, and subsequently write it back again at a later stage.

A further major limitation is that the routines given will only work with a single file - the channel number of which is assumed to be in the variable *ch%*. A useful modification would be to allow more than one file to be buffered. If this was done, it would be necessary to pass the channel number to the routines, and to maintain separate buffers for each open file, which would clearly require more memory.

Another consideration is the implementation of the function PTR#. As it stands, our technique will only work with sequential access to files. To allow for random access it is necessary to write a procedure to handle the reading and writing of PTR#. Hopefully, though, this article has provided you with ideas for extending this buffering technique and tailoring it to your own needs.

Expanding the Music 5000

Ian Waugh investigates the the expansion of the Music 5000 with the Music 2000 and Music 3000 units.

Product	Music 3000 Expander
Price	£115.00 inc. VAT
Product	Music 2000 MIDI Interface
Price	£161.00 inc. VAT
Supplier	Hybrid Technology Limited, 273 The Science Park, Cambridge CB4 4WE. Tel. (0223) 420360

The Hybrid Music System has been in a continuous state of development ever since the launch of the Music 500 back in 1985. We reviewed the much friendlier Music 5000 in November 1986 (Vol.5 No.6) and the accompanying keyboard, the Music 4000, the following year in our Aug/Sept issue (Vol.6 No.4).

There is also the Music 1000, an 8 watt stereo amplifier with three headphone sockets (ideal for education), and suitable for powering bookshelf speakers. To complete the line-up, the Music 2000 was launched last year and the Music 3000 this year.

To recap, the Hybrid Music System is a fully-integrated, self-contained synthesiser and music composition system controlled entirely from software. At its heart is the AMPLE Nucleus ROM (AMPLE standing for Advanced Music Production Language and Environment). AMPLE is a programming language specially designed for music, and one of its main features is concurrency, the ability to do more than one thing at the same time allowing several music parts to play together.

The sounds come from the Music 5000, the basic module of the system, which can produce eight different voices at once. Music parts can be entered in traditional notation in the Staff Editor, in an MCL (Music Composition Language) in the Notepad text editor, or in real-time from the Music 4000 keyboard, and individual parts which make up a piece can be mixed together using a graphic-based computer Mixing Desk.

The Hybrid Music System is widely used in education and has a growing number of individual users.

The basic Music 5000 has two main limitations - it can only play a maximum of eight voices at once and it can't access external voices via MIDI, the standard music interface (see later). Enter the Music 3000 and Music 2000. Both are housed in the same BBC beige box and both connect to the Music 5000 (and hence the BBC micro) via a 1 MHz bus cable. Both, thoughtfully, have follow-on sockets to allow further daisy chaining, and both fully integrate into the existing Hybrid Music System through additional controlling software which loads from the system disc.

THE MUSIC 3000

If you have produced pieces with the Music 5000, you'll have no problem using the Music 3000. Its function is simply to add another eight voices to the system. This allows you to create bigger 'drum kits' and six or eight-note piano parts, while still leaving plenty of voices for other sounds. You could even use all 16 voices to create one gigantic instrument! The extra voices can also be used for special effects such as Echo which eats voices like laryngitis.

A single music part can contain up to 12 voices, and the number of parts the system can support rises to ten.

The unit is completely transparent in use. The only operational change to the software is the inclusion of an additional eight sets of channels in the Mixing Desk. If you try to move the cursor off the right of the screen these new channels appear numbered 9, A, B, C, D, E, F and G.

This increased power is terrific, but when you come to enter a score for a 16-piece ensemble you may run into a problem - lack of memory.

In these days of 16-bit and 32-bit computers, the poor old BBC is showing its limitations but, as of writing, Hybrid has firmly stated that it does not intend to produce an Archimedes version.

A second, minor problem may also become apparent if you do use more than eight voices - the 'mix' word may not fit into the Notepad. Hopefully this will act as a spur to Hybrid to release a more fully-featured text editor.

THE MUSIC 2000

The Music 2000 is powered from the BBC's power supply and this has a follow-on socket, too. It has three MIDI Out sockets, one MIDI In, and there are three LEDs on the front which flash when data is being transmitted. Useful.

The Music 2000 software adds new words to the AMPLE language (see figure 1). The number of voices available now rises dramatically from eight to 32, and an arrangement can consist of up to nine parts each with up to 12 voices (within the 32 voice limit).

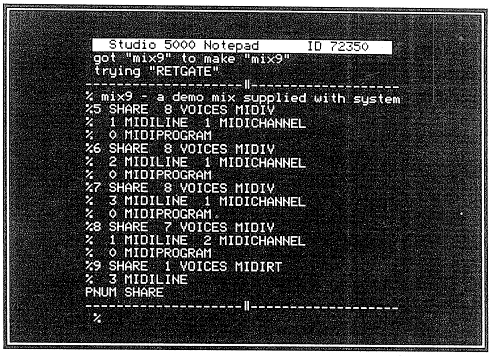
MIDIBEND	set pitch bend
MIDICHANNEL	set channel
MIDICHPRESSURE	set channel pressure
MIDICONTROL	set control
MIDILINE	set output line
MIDIOUT	send byte
MIDIPRESSURE	set pressure
MIDIPROGRAM	select program
MIDIRT	assign a real-time control voice
MIDIV	assign a voice
MIDIWOUT	send a word
The following are not part of the MIDI software but may often be used in word definitions.	
GATE	set gate
PITCH	set pitch
VEL	set velocity (dynamic level)

Figure 1. Music 2000 Words

This isn't the place for a MIDI tutorial but briefly, MIDI is an acronym for Musical Instrument Digital Interface. It is a sort of hi-tech musical instrument Esperanto which allows different instruments to communicate with each other. It is ideally suited to controlling instruments from a computer.

MIDI can handle a wide range of messages. The basic ones include Note On and Note Off information and the system can handle timing information too, to allow drum machines and sequencers to run in sync.

These basic control messages are built into the software, and voice assignments are carried out in the mix words. A sample mix, *mix9* (see figure 2), is supplied on disc. You can *EXEC this into a program and after making a few alterations to the mix you will be able to play a simple piece via MIDI.



```
 Studio 5000 Notepad       ID 72350
 got "mix9" to make "mix9"
 trying "RETGATE"
-------------------||-------------------
% mix9 - a demo mix supplied with system
%5 SHARE  8 VOICES MIDIV
%  1 MIDILINE  1 MIDICHANNEL
%  0 MIDIPROGRAM
%6 SHARE  8 VOICES MIDIV
%  2 MIDILINE  1 MIDICHANNEL
%  0 MIDIPROGRAM
%7 SHARE  8 VOICES MIDIV
%  3 MIDILINE  1 MIDICHANNEL
%  0 MIDIPROGRAM
%8 SHARE  7 VOICES MIDIV
%  1 MIDILINE  2 MIDICHANNEL
%  0 MIDIPROGRAM
%9 SHARE  1 VOICES MIDIRT
%  3 MIDILINE
PNUM SHARE
-------------------||-------------------
 %
```

Figure 2. Mix9 definition supplied with the Music 2000.

The four most important words are probably these:

MIDIV makes a voice a MIDI voice (as opposed to a Music 5000 voice).

MIDICHANNEL selects the MIDI channel number. MIDI can transmit different music parts on up to 16 different channels allowing you to select which sounds on your MIDI equipment each music line is heard with.

MIDILINE selects the MIDI Out socket to be used - 1, 2 or 3.

MIDIPROGRAM selects a new patch or sound on the receiving MIDI instrument.

Together, they might be used like this:

```
1 SHARE 3 VOICES MIDIV
1 MIDILINE 2 MIDICHANNEL
8 MIDIPROGRAM
```

This assigns three voices to part one and transmits them on channel two through MIDI socket one and selects patch number 8.

You can freely mix Music 5000 and MIDI voices, although the MIDI voices, obviously, do not appear in the Mixing Desk. Using AMPLE and the new MIDI words, you can construct and transmit virtually any MIDI message or combination of messages. Using the hierarchical nature of the language, you can do this from just one word, too.

It is easy to program changes in tempo on the Hybrid Music System, and this feature is carried over to MIDI control. Programmable tempo changes are not possible on all MIDI-based sequencers, and few drum machines can perform tempo changes on their own.

An alternative method of rhythm programming is to build a music part from the notes which trigger the sounds on a drum unit. The manual includes words which let you create drum patterns on a grid similar to that used by Roland drum machines and found in most books containing drum machine patterns.

Expanders are synthesisers without a keyboard. Most are multi-timbral which means they can play several different music lines using different voices at the same time (each would be tuned to a separate MIDI channel). Through the Music 2000 you can use the Music 4000 keyboard to play voices on a MIDI-compatible instrument. The Music 2000 reproduces all the elements of standard AMPLE notation on MIDI voices except one - the slur. This is the sounding of a sequence of notes without retriggering the sound's attack phase. The MIDI specification was basically designed for keyboards and does not have a slur command. The Music 5000 voices, of course, can play slurs.

One other point to bear in mind: loudness is expressed as key velocity so MIDI instruments must be velocity sensitive if dynamics are to be heard. Again, the Music 5000 voices are *velocity sensitive.*

The greatest disappointment, if you like, about the Music 2000 is the MIDI In socket. This is not supported by any high level AMPLE words, and it is doubtful whether Hybrid intends to make any available in the future. Oddly, the MIDI In socket was specially included late in the development stage. However, it is rumoured that some enterprising users have written their own machine code patches to support MIDI In although I am not personally aware of any.

This lack of support means that programs such as voice editors - should some enterprising individual write some - would be limited to a one-way conversation, and voice data could not be saved to disc - one of the great benefits of computer-based voice editors. It also means that a MIDI keyboard cannot be used for recording.

The Music 2000, therefore, must be the controlling instrument in any MIDI set-up. It has no sync to tape facility either, so you cannot overdub onto tape. However, I suspect that won't concern the vast majority of users.

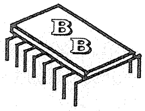
It's rather a shame that the 2000 isn't the same price as the 3000, but remember that if you are interested in using the system purely for MIDI, you also need to the Music 5000.

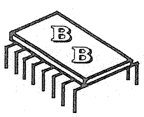
SUMMARY

If you already have a Music 5000 and want more voices for your System, the Music 3000 represents unbeatable value, breaking as it does Hybrid's traditional £161 price point. It is by far the cheapest music expander on the market.

If you like the power and flexibility of AMPLE but would like to apply it to a greater variety of sounds and instruments, the Music 2000 is for you. Its ability to transmit any MIDI message makes it among the most powerful MIDI controllers on the market. In spite of some operational restrictions, it is an extremely exciting addition to the Hybrid Music System in particular and the world of computer-based music in general.

Basic Booster





The Best of BEEBUG

The Basic Booster ROM from BEEBUG is a selection of useful utilities to 'boost ' the Basic in your computer and give more power to your programming.

Some of the utilities are enhanced versions of programs previously published in BEEBUG magazine, while others are totally new. For those with no spare ROM sockets, Basic Booster is also available on disc to load into Sideways RAM.

Partial Renumber

A very useful utility which renumbers a selected block of lines. Ideal for adding extra lines to the middle of a lengthy program.

SPECIAL SHOW OFFER! The Basic Booster will be available at the BBC Acorn User Show for just £6.00 (members only)!!

Super Squeeze

A program compressor which can remove REMs, blank lines, spaces, and compress variable names.

Textload and Textsave

Save and load a Basic program as text. Saves the hassle of using *SPOOL and *EXEC.

Program Lister

List any program (including Archimedes Basic) direct from a file. Formatting can be controlled using the same options as are available on the Archimedes, including splitting multi-statement lines, omitting line numbers and listing keywords in lower case.

Resequencer

Rearrange the lines in a Basic program. Any line, or block of lines, can be moved or copied to any place in the program. Line numbering is automatically adjusted as blocks are moved.

Smart Renumber

Renumber a program to a standard format with procedures starting at a particular line number

Please rush me my Basic Booster ROM/Disc:

Basic Booster ROM	*Code 1403A*	£10.95	Basic Booster Disc	*Code 1402A*	£9.95
Basic Booster ROM	*(non-members)*	£19.95	Basic Booster Disc	*(non-members)*	£18.95

Name ______________________

Address ______________________

____

Memership No ______________________

ROM/Disc *(delete as necessary)*	£______
Postage	£ 0.60
Total	£______

I enclose a cheque for £__________ OR please debit my Access, Visa or Connect account,
Card No ______/______/______/______ Expiry_____/_____ Signed______________________

Return to BEEBUG Ltd, Dolphin Place, Holywell Hill, St Albans, Herts AL1 1EX. Telephone (0727) 40303.

BEEBUG Education

As part of a long term plan it is the intention that BEEBUG Education should look at available software and its use on the Acorn computer family in major curriculum areas. This month Mark Sealey turns his attention to the subject of history.

A LESSON IN HISTORY

At any level, there are traditionally considered to be several difficulties associated with the teaching of history. Not least is the fact that many pupils will find it hard to relate directly to the remoteness or unfamiliarity of much of the material: lifestyles and events may well be outside their experiences, for example.

The best educational software attempts to overcome this in a number of ways: by taking a single event and rendering it as vivid and immediate as possible; by personalising the past and dealing, say, with family histories; and by examining an activity, job or 'snapshot' of the past in great detail, and treating it as representative of wider themes and issues. Or by providing extended scope for related work (visits, drawings, modelling, reconstruction etc.) away from the computer.

When choosing software for this area of work, it is important to bear these approaches in mind. All the software mentioned this month employs one or more of them.

Exciting and innovative packages exploring time periods from Bronze Age to 20th Century Britain (and beyond) include Time Traveller (Sulis £11.70 or £12.50 excl. VAT for 40 or 80 track discs) which takes the form of a maze game with factual quiz questions. ISCA and Settlement (ESM - £30 and £15 respectively, excl. VAT) offer chances for role-play and drama as pupils engage in a simulation of life in Roman Britain and an 18th Century colony. Once again, where these packages score is by providing much 'off the computer' material and activities.

Two games and multi-media packs from the ever enterprising RESOURCE which deal with the Industrial Revolution and its impact on Midlands life are High Peak Railway (£23.95 excl. VAT) and Forge (£44.95 - same price for Master Compact and Archimedes). By taking the seemingly quite specific and exploring it in a child-centred way, these excellent program suites allow pupils of 10 to 14 to become immersed in the issues and flavour of the period as much, perhaps, as any other school activity except visits, which the packs would certainly not preclude.

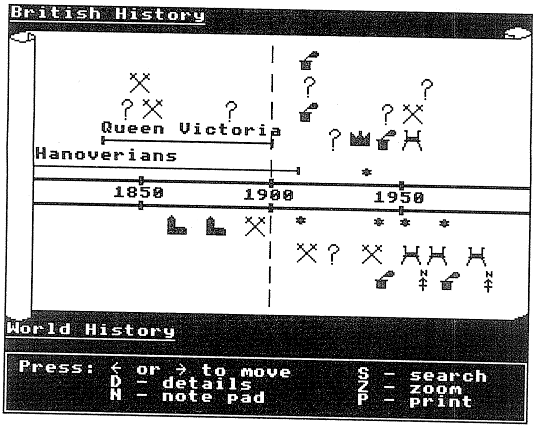
FEARNLEAF

Some excellent material which you might consider are those in the Fearnleaf History Packs series. Often they use more than one medium (books, videos etc), are usually aimed at the 8 to 13 age range and contain a wealth of resources centred around a precise location (like York) or an object (like a castle or ship). Fletcher's Castle and Castle Under Attack (£14.95 and £17.95 excl. VAT) are excellent examples, and include opportunities for role play and historical gaming.

TIME LINES

An altogether different approach is that taken by Time Lines (from Soft-Teach, price £21.50 excl. VAT, p&p 80p). As the name suggests, the suite allows you to store events on a sort of 'linear database'. Simply put, dates and their associated events scroll across the screen continuously. A simple idea, too, but there is much more to it. Symbols representing battles, literature, exploration, building and so on can also be placed, and greater detail about any one era or year etc. gone into by means of a card catalogue.

Although searching, editing and printing facilities are excellent, Time Lines is undoubtedly open to the criticism of presenting



```
Press: ← or → to move      S - search
       D - details         Z - zoom
       N - note pad        P - print
```

Scrolling Time Line screen

too mechanistic a view of history. Events and their influences do not fit precisely with finite or even known beginnings and endings. The suite's facilities for cross-referencing and showing parallel trends, though, offset this to some degree. Similarly the scrolling nature of the display should help many children to develop understanding of historical perspective and continuity. Documentation is crisp and clear with just the right amount of helpful guidance and information. A sample database is supplied, and network and Archimedes versions will be available shortly too.

KINSHIP

A more personal approach is taken by the latest offering from BBC Soft. The author of Ancestry for the Archimedes (published by Minerva) has written a Master 128 only database called Kinship (£29.95 plus 1.30 p&p inc. VAT). It is a complex suite of programs, but very easy to use with documentation well up to BBC Soft standards. Again there is a sample database - two in fact: the British Royal Family from Henry VII onwards and that of one of the best known popularisers of the subject of genealogy, Gordon Honeycombe.

Movement around the database is intuitive (cursor and via specific on-screen instructions etc). You toggle between the individual personal biographies of the members of each branch of the families and the relationships (parental or sibling) between them all.

Full saving, loading and printing facilities are all well described in the very approachable manual, which includes a clear 16 page tutorial. This program is noteworthy because it makes full use of the Master's sideways RAM for program overlays. Nevertheless only 250 records can be held in memory at once - fewer if you incorporate your own 'User File', a means of supplementing the data by adding fields with details of your own about each member of the family.

At any event, Kinship is a very good example of this sort of program and ought to be investigated where this sort of record keeping is part of a history course or project.

SUPPLIERS

Fearnleaf Educational Software Ltd.,
Fearnleaf House,
31 Old Road West,
Gravesend,
Kent DA11 0LH.

Sulis Software
Wiley, Baffins Lane,
Chichester PO19 1UD.

ESM
Duke Street,
Wisbech,
Cambs PE13 2AE.

RESOURCE
Exeter Road,
Off Coventry Grove,
Doncaster DN2 4PY.

Soft-Teach Educational
Sturgess Farmhouse,
Longbridge Deverill,
Warminster,
Wilts BA12 7EA.

BBC Soft
PO Box 234,
Wetherby,
West Yorkshire LS23 7EU.

1st course

Investigating Teletext Mode (4)

Mike Williams returns to basics, and also describes some of the special effects possible in mode 7.

I want to look at some further aspects of Teletext colour control this month, but first of all a correction to last month's article. The right-most column of figure 1 was shown containing code 157 from top to bottom. If you had read the text carefully, you will probably have realised that this should be code 156. Code 157, following a colour code, specifies a new background colour, while code 156 switches off the background colour (i.e. restores it to its default of black). My apologies for that error.

MORE WAYS OF GENERATING COLOURED TEXT

Let's start this time by looking at some quite basic ideas. Normally, when we print any Teletext mode listings in the magazine, we have to use the CHR$ function to specify any control codes. Thus:

```
   PRINT CHR$(129)"Hello"
```

will result in the word 'Hello' appearing in red text. However, because of the initial control character, the word will appear to be indented by one space from the side of the screen.

Now the control character, although invisible, is a character like any other, and may therefore be copied with the Copy key. Try the following, assuming that you have entered the PRINT statement above (in mode 7 of course). Type:

```
   PRINT"
```

and then without pressing Return move the flashing cursor (using the cursor keys) to the blank position in front of the word 'Hello' that resulted from the PRINT statement. Use Copy to copy that control character and the following word, ending with quotes (") and pressing Return. What you have typed will look like this:

```
   PRINT" Hello"
```

but the blank in front of 'Hello' is really the control character and so the word will be printed in red as before.

As a further experiment, use the cursor keys to copy the new PRINT statement up to and including the control character, but then complete it by typing in some new word (Goodbye, for example) before finishing with quotes and pressing Return. This time what you have typed will look like this:

```
   PRINT" Goodbye"
```

but again the resulting word will be displayed in red. All the Teletext control codes can be copied and used directly in programs in this way. This does save memory, if space is critical (unlikely in mode 7 which only uses 1K for the screen display). It does have one major disadvantage though. Control codes, at best, only appear as spaces on the screen, and if you print out a program listing you will probably find that they have disappeared all together, which is why we never use this approach with magazine listings.

There are some advantages though, as well as more efficient use of memory. For example, if say a warning message was always to be displayed in red, then not only the text but the control code as well could be part of a string assigned to a variable and used when required. For example:

```
   text$=CHR$(129)+"Warning"
```

Although I have used the CHR$ function, it is the corresponding Teletext code followed by the text which is assigned to the variable *text$*.

Another use for the direct code technique is to allow coloured REM statements in programs, which does make them stand out. Suppose you decide upon green as the colour (code 130). You will first need to create a green control code which can then be copied, so type:

```
   PRINT CHR$(130)"GREEN"
```

The reason for including the word 'GREEN' is just so that the position of the control code is clearly marked at the edge of the screen.

Now create a REM statement, but copy the green control code immediately after the REM itself. The result will look something like this:

```
   100 REM Any comment
```

but you should find the words 'Any comment' now appear in green, although the line number and the REM itself are still in white. A word of caution is due though before you get carried away by thoughts of colour. Remember that Teletext control codes only work in mode 7, not always the best mode for program listings.

USING THE FUNCTION KEYS

There is another and more convenient way of directly generating Teletext codes. You just use the Shift key in combination with the function keys f1 to f7 (see table 1).

Shift-f1	RED
Shift-f2	GREEN
Shift-f3	YELLOW
Shift-f4	BLUE
Shift-f5	MAGENTA
Shift-f6	CYAN
Shift-f7	WHITE

Table 1. Using function keys for text colours

In exactly the same way, using the Ctrl key with the same set of function keys will generate the equivalent graphics colour control codes. This makes life a lot easier, and avoids the need to use CHR$ to create the control code in the first place, and the need to use the Copy key to incorporate it in other statements.

COLOURED GRAPHICS

Control codes 129 to 135 generate the text colours, and codes 145 to 151 generate the graphics colours. Remember though that the choice of a text or graphics colour control code determines not only the colour, but whether the following character is displayed as a letter or as a graphics character.

The tables in your User Guide will show two different sets of codes for the graphics characters, 32 to 63 and 95 to 127 for one, and 160 to 191 and 223 to 255 for the other. Most of the time either set will produce the same result, but to be sure of always getting the correct result use the codes from 160 and above. The VDU drivers interpret a few codes differently (code 127 is treated as Delete, for example).

If you find it difficult to determine from the tables which code you need to specify for a particular graphics character, there is another approach. Each graphics character is made up of six pixels arranged 2 by 3 vertically. Each of these pixels can be given a weight as shown in figure 1.

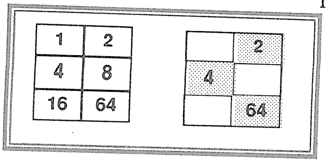


Fig.1 Weighting of graphics pixels

Thus if you want to determine the graphics character as shown in figure 1, add up the corresponding values (2 + 4 + 64 = 70), add on the base value of 32, and a further 128 to select the upper range of codes. In the example this gives the result of 230, so the character we require is character 230 (160 + 70), which is equivalent to 'f'. So this graphics character could be displayed in cyan, say, by putting:

```
PRINT CHR$150;CHR$230
```

You may also find it helpful to refer to the article and short program in Vol.7 No.8 which allows the Teletext graphics characters to be displayed in a rather more logical order than appears to be the case with the User Guides.

SPECIAL EFFECTS

As well as the text and graphics colour control codes, there are a number of other codes used in mode 7 which have certain special effects. We have already seen how code 141 may be used to produce double height text, and codes 157 and 156 to specify a new background colour, and to return to the default black background. There are a number of other special codes, and all of these are listed in table 2.

	CODE	MEANING
	136	flash
*	137	steady
*	140	normal height
	141	double height
	152	conceal display
*	153	contiguous graphics
	154	separated graphics
*	156	black background
	157	new background
	158	hold graphics
*	159	release graphics

Table 2. Codes for special effects

The entries in the table marked with an asterisk (*) are the default settings for each line of a mode 7 display. Each code affects what follows on the same line until another code cancels or changes the effect.

Thus code 136 causes subsequent characters to flash, while code 137 turns this effect off. Try typing:

```
PRINT"This is"CHR$136"flashing"CHR$137"text"
```

Notice how the two control codes are conveniently disguised as the spaces between words. The corresponding control characters can also be generated with Shift-f8 and Shift-f9.

The code for double height text (code 141) we have seen before, but note also the code for restoring normal height text (code 140). Code 152, *conceal display,* can be quite useful. Placed in front of any text or graphics it appears to make it invisible. This could be used to prevent the display of a password, for example, or the answer to a question could be 'shown' initially hidden, to be subsequently revealed by overwriting the conceal code with a space or some other control character.

By default, all graphics characters are *contiguous*, that is each character fully occupies the character position on the screen, and adjacent graphics characters join together to form continuous areas of colour. If *separated* graphics is selected, each graphics pixel occupies some 90% or so of the full size, and graphics characters have a characteristic mosaic appearance, which can be quite pleasing.

Codes 156 and 157 controlling the use of background colours we dealt with last time, so that just leaves us with *hold graphics* (code 158) and *release graphics* (code 159). These are probably the least understood, and thus least used of all the Teletext control codes.

We have already seen that Teletext control characters appear as spaces on the screen because they exist as characters just like any other. If we want to display two graphics characters side by side but in different colours, then the new colour control character will appear as a space separating the two graphics characters.

For example try typing (in mode 7):

```
  PRINT CHR$145;STRING$(15,CHR$175)
```

This will display a solid bar in red. Now change the instruction to introduce a colour change:

```
  PRINT CHR$145;STRING$(15,CHR$175);
CHR$147;STRING$(15,CHR$175)
```

You should now see two bars, one red and one yellow, separated by a space (the control code for graphics yellow). To avoid the gap, we can use code 158. This 'holds' the existing graphics character (code 175 here) and uses this to represent any subsequent control character on the screen. So type:

```
  PRINT CHR$145;STRING$(15,CHR$175);
CHR$158;CHR$147;STRING$(15,CHR$175)
```

and the gap should now disappear, but the red bar is now two characters longer than before as the control codes 158 and 147 are both now represented on screen by red graphics characters code 175. To achieve the desired result, the length of the first string will need to be reduced by two to compensate:

```
  PRINT CHR$145;STRING$(13,CHR$175);
CHR$158;CHR$147;STRING$(15,CHR$175)
```

As a further example try typing:

```
  PRINT CHR$145;CHR$152;CHR$255;CHR$158;
CHR$146;CHR$147;CHR$148;CHR$149;CHR$150;
CHR$151;CHR$159
```

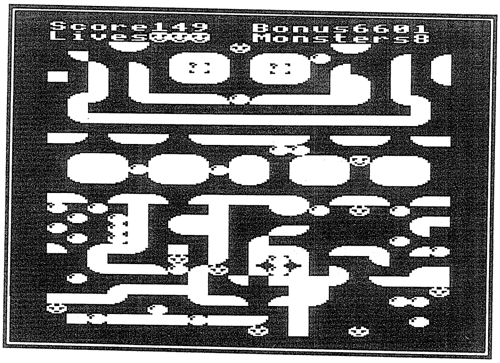
You should see a sequence of six solid blocks each in a different colour. Remember that each colour change only affects following characters, not itself. Code 152 conceals characters until the next colour change occurs. Code 158 'holds' the specified graphics character (code 255). Each colour change character then appears a solid block in the preceding colour. The last code (159) appears as a white block, but 'releases' the graphics character.

That's all I have room for this time. Next month I hope to conclude this short series of articles on the Teletext mode by describing a number of useful mode 7 functions and procedures.

Egghead

John McFarlane introduces his well-planned arcade style game program. But to be successful you will need brains as well as brawn.

Egg-Head is a Bone-Crusher style game in which Eddie Egg-Head roams through the caverns crushing monsters with boulders. Instructions and a high score table are provided. Eddie has three lives and loses one if he meets a monster or runs out of oxygen.



To kill the monsters Eddie must trap them in a confined space using the boulders until they suffocate. But be careful because it is easy to make mistakes which could jeopardize the rest of the game. Also, once you die, you go back to the beginning of the screen and may be cut off! When you kill all the monsters on one level, you go on to the next. There are three levels altogether.

If you want time to think ahead or leave the game for a while, pressing 'P' will pause the game during which time if you don't want sound you can turn it on or off using the usual 'S' and 'Q'. When you are ready to restart simply press 'R'.

If you make a mistake and get trapped, pressing 'E' will let you quietly die. If you know you have made a mistake which cannot be undone or if you wish to read the instructions press Escape. Should you wish to exit from the game completely press Escape twice.

Take care typing the program in, particularly the screen definitions in data statements at the end. For safety, make sure you save a copy of the program before starting to play.

```
   10 REM Program EggHead
   20 REM Version B1.3
   30 REM Author  J.McFarlane
   40 REM BEEBUG  July 1989
   50 REM Program subject to copyright
   60 :
  100 ON ERROR GOTO300
  110 PROCinit
  120 MODE7:PROCinstr
  130 REPEAT
  140 REPEAT:UNTIL INKEY-99
  150 MODE5:PROCbegin
  160 REPEAT
  170 Bonus%=Bonus%-Monster%DIV3-1
  180 IF INKEY-56 PROCpause
  190 PROCman:PROCmonster
  200 IF Monster%=0 PROCbonus:IF Level%<
4 PROCscreen
  210 PROCinfo
  220 IF Dead% OR Bonus%<1 PROCouch
  230 UNTIL Lives%=0 OR Level%=4
  240 IF Level%=4 PROCcongrats ELSE PROC
end
  250 MODE4
  260 IF Score%>Hisc%(8) PROChigh
  270 PROCtable
  280 UNTIL FALSE
  290 :
  300 IF ERR=17 AND ERL<>140 GOTO 120
  310 MODE7:REPORT:PRINT" at line ";ERL
  320 END
  330 :
 1000 DEF PROCinit
 1010 VDU23,128,0;0;0;0;
 1020 VDU23,129,&1F07;&7F3F;&FF7F;&FFFF;
 1030 VDU23,130,&F8E0;&FEFC;&FFFE;&FFFF;
 1040 VDU23,131,&FFFF;&FEFF;&FCFE;&E0F8;
 1050 VDU23,132,&FFFF;&7FFF;&3F7F;&71F;
 1060 VDU23,133,&FFFF;&FFFF;&FFFF;&FFFF;
 1070 VDU23,134,&5E3C;&BFBF;&FFFF;&3C7E;
 1080 VDU23,135,&6624;&99FF;&FFDB;&3C66;
 1090 VDU23,136,&FF7E;&BB99;&DBFF;&3C66;
 1100 ENVELOPE1,3,0,0,0,0,0,0,121,-10,-5
,-2,120,120
 1110 DIM Hisc$(8),Hisc%(8)
 1120 FOR A%=1TO8:Hisc$(A%)="JOHN"
 1130 Hisc%(A%)=900-A%*100:NEXT
 1140 ENDPROC
 1150 :
```

```
 1160 DEF PROCbegin
 1170 RESTORE
 1180 VDU23,1,0;0;0;0;
 1190 VDU19,2,6,0,0,0
 1200 Dead%=FALSE
 1210 PRINT"Score";TAB(10)"Bonus";TAB(0,
1)"Lives";TAB(10,1)"Monsters"
 1220 Level%=1:Score%=0:Lives%=3
 1230 PROCscreen
 1240 ENDPROC
 1250 :
 1260 DEF PROCscreen
 1270 COLOUR3:PRINTTAB(0,2);
 1280 Monster%=0:Bonus%=9999
 1290 Z%=&5500:M%=&900
 1300 FOR Y%=2TO30:READA$
 1310 FOR X%=1TO20:COLOUR2
 1320 ?Z%=VAL(MID$(A$,X%,1))
 1330 IF ?Z%=7 ?M%=X%-1:M%?1=Y%:M%?2=RND
(4)-1:M%?3=0:M%=M%+4:Monster%=Monster%+1
:COLOUR1 ELSE IF?Z%=6 COLOUR3
 1340 VDU?Z%+128
 1350 Z%=Z%+1
 1360 NEXT,:COLOUR3
 1370 ManX%=9:ManY%=2:PRINTTAB(ManX%,Man
Y%)CHR$136
 1380 M%?4=&FF:PROCinfo:M%=&900:Max%=&90
0+Monster%*4
 1390 ENDPROC
 1400 :
 1410 DEF PROCinfo
 1420 COLOUR3:PRINTTAB(5,0);Score%;TAB(1
5,0);Bonus%;" ";TAB(5,1)STRING$(Lives%,C
HR$136);" ";TAB(18,1)"  ";TAB(18,1);Mons
ter%
 1430 ENDPROC
 1440 :
 1450 DEF PROCman
 1460 IF FNblock(ManX%,ManY%)=7 OR INKEY
-35 Dead%=TRUE:ENDPROC
 1470 X%=INKEY(-98)-INKEY(-67)
 1480 IF FNgoahead(ManX%+X%,ManY%)=0X%=0
 1490 IF X% IF FNblock(ManX%+X%,ManY%)=6
 IF FNgoahead(ManX%+X%*2,ManY%)<>-1X%=0
 1500 Y%=INKEY(-73)-INKEY(-105)
 1510 IF FNgoahead(ManX%,ManY%+Y%)=0Y%=0
 1520 IF Y% IF FNblock(ManX%,ManY%+Y%)=6
 AND FNgoahead(ManX%,ManY%+Y%*2)<>-1Y%=0
 1530 IF (X%ORY%)=0 ENDPROC
 1540 IF X% AND Y% X%=0
 1550 COLOUR3:PRINTTAB(ManX%,ManY%)" ";
 1560 ManX%=ManX%+X%:ManY%=ManY%+Y%
 1570 PRINTTAB(ManX%,ManY%)CHR$136;
 1580 IF FNblock(ManX%,ManY%)=6 PRINTTAB
(ManX%+X%,ManY%+Y%)CHR$134;:PROCchange(M
anX%+X%,ManY%+Y%,6):PROCchange(ManX%,Man
Y%,0):SOUND0,-10,5,2
 1590 ENDPROC
 1600 :
 1610 DEF PROCmonster:COLOUR1
 1620 M%=M%+4:IF M%=Max%M%=&900
 1630 IF M%?3=1 GOTO1620
 1640 IF FNgoahead(?M%,M%?1-1)<>-1 IF FN
goahead(?M%+1,M%?1)<>-1 IF FNgoahead(?M%
,M%?1+1)<>-1 IF FNgoahead(?M%-1,M%?1)<>-
1 PROCgot:ENDPROC
 1650 IF M%?2=0 Y%=-1:X%=0 ELSE IF M%?2=
1 X%=1:Y%=0 ELSE IF M%?2=2Y%=1:X%=0 ELSE
 IF M%?2=3 X%=-1:Y%=0
 1660 IF FNgoahead(?M%+X%,M%?1+Y%)<>-1 M
%?2=(M%?2+SGN(RND(2)-1.5)+4)MOD4:GOTO165
0 ELSE PROCchange(?M%,M%?1,0):VDU31?M%,M
%?1,32:?M%=(?M%)+X%:M%?1=(M%?1)+Y%:VDU31
,?M%,M%?1,135:PROCchange(?M%,M%?1,7)
 1670 ENDPROC
 1680 :
 1690 DEF PROCgot
 1700 Monster%=Monster%-1:M%?3=1
 1710 Score%=Score%+Bonus%DIV100
 1720 PROCchange(?M%,M%?1,0)
 1730 PRINTTAB(?M%,M%?1)" ";:SOUND0,1,6,
1
 1740 ENDPROC
 1750 :
 1760 DEF PROCouch
 1770 COLOUR3:A%=255
 1780 REPEAT
 1790 SOUND1,-10,A%,1
 1800 A%=A%-RND(32)
 1810 UNTIL A%<15
 1820 PRINTTAB(ManX%,ManY%)" "
 1830 Lives%=Lives%-1:Dead%=FALSE
 1840 ManX%=9:ManY%=2
 1850 PRINTTAB(ManX%,ManY%)CHR$136:Bonus
%=9999
 1860 PROCinfo
 1870 ENDPROC
 1880 :
 1890 DEF PROCend
 1900 A$="GAME OVER":VDU18,0,0,5
 1910 MOVE360,492:PRINT A$
 1920 Y%=496
 1930 FOR A%=1TOLENA$:GCOL4,0
 1940 B$=MID$(A$,A%,1)
 1950 X%=288+A%*64:X=0:Y=0
 1960 x=(X%-X)/10:y=(Y%-Y)/10
 1970 MOVE X,Y:PRINT B$
 1980 FOR B%=1TO10
```

```
1990 *FX15
2000 SOUND1,-10,B%*4,1
2010 MOVE X,Y:PRINT B$
2020 X=X+x:Y=Y+y
2030 MOVE X,Y:PRINT B$
2040 NEXT
2050 GCOL0,3:MOVE X,Y:PRINT B$
2060 NEXT
2070 *FX15
2080 IF INKEY100
2090 ENDPROC
2100 :
2110 DEF PROCtable
2120 VDU23,1,0;0;0;0;
2130 RESTORE3820:READ A$
2140 FOR A%=1TO199:VDU128+VAL(MID$(A$,A
%,1)):NEXT
2150 PRINT''TAB(14)"HIGH SCORES"
2160 FOR A%=1TO8
2170 PRINTTAB(0,7+A%*2)A%;".";STR$Hisc%
(A%);TAB(30-LEN(Hisc$(A%)),7+A%*2)Hisc$(
A%)
2180 NEXT
2190 PRINTTAB(14,28)"Press <SPACE>"
2200 ENDPROC
2210 :
2220 DEF PROCinstr
2230 VDU23,1,0;0;0;0;
2240 FOR A%=0TO1
2250 PRINTTAB(12,A%)CHR$141;CHR$132;CHR
$157;CHR$(133+A%);"EGG HEAD  ";CHR$156
2260 NEXT
2270 PRINTTAB(12,A%)CHR$129"By J.McFarl
ane"
2280 PRINT'CHR$134;"   Eddie Egg Head,
beastie slayer"
2290 PRINTCHR$134;"unlimited has been a
ssigned the task"
2300 PRINTCHR$134;"of destroying all th
e baddies in a"
2310 PRINTCHR$134;"local underground ca
ve."
2320 PRINT'CHR$134;"   Kill the little
nastie thingies by"
2330 PRINTCHR$134;"crushing them with t
he conveniently"
2340 PRINTCHR$134;"placed boulders befo
re the oxygen"
2350 PRINTCHR$134;"runs out."
2360 PRINT'CHR$134;"The keys are:-"
2370 PRINT'TAB(5)CHR$134;"Z........LEF
T  E.........DIE"
2380 PRINTTAB(5)CHR$134;"X.......RIGHT
 P.......PAUSE"
2390 PRINTTAB(5)CHR$134;"*...........UP
 R......RESTART"
2400 PRINTTAB(5)CHR$134;"?........DOWN
 Q....SOUND.OFF"
2410 PRINTTAB(5)CHR$134;"ESCAPE.....END
 S.....SOUND.ON"
2420 PRINT'TAB(13)"<SPACE> to play"
2430 ENDPROC
2440 :
2450 DEF PROCbonus
2460 FOR A%=53 TO 77 STEP 8
2470 FOR B%=A% TO A%+48 STEP 8
2480 SOUND1,-10,B%,1
2490 NEXT,
2500 Level%=Level%+1
2510 ENDPROC
2520 :
2530 DEF PROChigh
2540 FOR A%=0 TO 255 STEP RND(16)+16
2550 FOR B%=A% TO 255 STEP RND(16)+16
2560 SOUND1,-10,B%,1:NEXT,
2570 PRINTTAB(14)"CONGRATULATIONS!"''TA
B(9)"YOU ARE IN THE TOP EIGHT!"''"Please
 enter your name":INPUT'"?"A$
2580 A$=LEFT$(A$,14):A%=8
2590 IF A%>1 IF Score%>Hisc%(A%-1) Hisc
%(A%)=Hisc%(A%-1):Hisc$(A%)=Hisc$(A%-1):
A%=A%-1:GOTO2590
2600 Hisc$(A%)=A$:Hisc%(A%)=Score%:CLS
2610 ENDPROC
2620 :
2630 DEF FNblock(P%,Q%)=?(&54D8+P%+Q%*2
0)
2640 :
2650 DEF FNgoahead(P%,Q%)
2660 IF P%<0 OR P%>19 OR Q%<2 OR Q%>30
THEN =0
2670 A%=FNblock(P%,Q%):IF A%>0 AND A%<6
 THEN =0 ELSE IF A%>5 THEN =-2 ELSE =-1
2680 :
2690 DEF PROCchange(P%,Q%,R%)
2700 ?(&54D8+P%+Q%*20)=R%
2710 ENDPROC
2720 :
2730 DEF PROCpause
2740 REPEAT
2750 IF ?&262 A$="OFF" ELSE A$="ON "
2760 PRINTTAB(2,31)"PAUSE, SOUND "A$;
2770 IF INKEY-82 THEN *FX210
2780 IF INKEY-17 THEN *FX210,1
2790 UNTIL INKEY-52
2800 PRINTTAB(2,31)SPC(16);
2810 ENDPROC
2820 :
 2830 DEF PROCcongrats
 2840 VDU12,19,1,8;0;:VDU19,2,15;0;5
 2850 GCOL3,1:MOVE128,1000:PRINT"C N R T
 L T O S ":MOVE128,1012:PRINT" O G A U A
 I N !"
 2860 GCOL3,2:MOVE128,1000:PRINT" O G A
U A I N !":MOVE128,1012:PRINT"C N R T L
T O S "
 2870 VDU4
 2880 PRINTTAB(3,3)"You have won!!"
 2890 PRINTTAB(3,7)"Press SPACE..."
 2900 REPEAT:UNTIL:INKEY-99:ENDPROC
 2910 :
 2920 DATA"0004300000000000430000"
 2930 DATA"04000015201520060101"
 2940 DATA"00202056505650100505"
 2950 DATA"50505045304530520504"
 2960 DATA"00505007000000550500"
 2970 DATA"01304555261555530420"
 2980 DATA"05000000000000000050"
 2990 DATA"04555555555555555530"
 3000 DATA""
 3010 DATA"53045304555530453045"
 3020 DATA"70000000000000000007"
 3030 DATA"01520152012015201520"
 3040 DATA"05556555655655565550"
 3050 DATA"04530453043045304530"
 3060 DATA""
 3070 DATA"52015201555520152015"
 3080 DATA"60605000500436000600"
 3090 DATA"06065015500000000015"
 3100 DATA"60065045500130000040"
 3110 DATA"06065000501700000600"
 3120 DATA"20153050505520152016"
 3130 DATA"00500007303640004000"
 3140 DATA"00455261072610200015"
 3150 DATA"06000005040555304555"
 3160 DATA"00015555000055000000"
 3170 DATA"55530004520153000660"
 3180 DATA"00000600000750000070"
 3190 DATA"05655556553057000060"
 3200 DATA"700000000000050000000"
 3210 :
 3220 DATA"74606000506040500000"
 3230 DATA"00060600565000550520"
 3240 DATA"01550530503020500050"
 3250 DATA"00000000500750000050"
 3260 DATA"55555520501555570050"
 3270 DATA"73006050553060456530"
 3280 DATA"00000050500006000000"
 3290 DATA"05520050500600155555"
 3300 DATA"05536130420155530007"
 3310 DATA"05600500600000000105"
 3320 DATA"05006420155555555304"
 3330 DATA"05006045300000000000"
 3340 DATA"05552000001555555555"
 3350 DATA"00005555553000006000"
 3360 DATA"00605500550605066670"
 3370 DATA"01053000050605555555"
 3380 DATA"550000000050605000007"
 3390 DATA"55555000550604201565"
 3400 DATA"53064505530600505000"
 3410 DATA"00006500000601305606"
 3420 DATA"02060571555200005000"
 3430 DATA"05601553060455205066"
 3440 DATA"05065000660000553660"
 3450 DATA"05605201060000500060"
 3460 DATA"05060005006100060600"
 3470 DATA"05555555555555555555"
 3480 DATA"04300060006005006006"
 3490 DATA"06000000650706060060"
 3500 DATA"00676050600600605007"
 3510 :
 3520 DATA"70000000500030400000"
 3530 DATA"50555520505060001520"
 3540 DATA"06555750505550503040"
 3550 DATA"60045050505700500000"
 3560 DATA"06003050504201305064"
 3570 DATA"01010050500005005007"
 3580 DATA"53050650555553005520"
 3590 DATA"50050030450000000050"
 3600 DATA"00150106000555552045"
 3610 DATA"01530452067570005000"
 3620 DATA"05506005055501205050"
 3630 DATA"00000165050006005530"
 3640 DATA"55555306000504306000"
 3650 DATA"00000005055500075555"
 3660 DATA"01555203000565553000"
 3670 DATA"03000550601060600020"
 3680 DATA"00050550005200000050"
 3690 DATA"52050555555555553050"
 3700 DATA"55700600500073000650"
 3710 DATA"55555201302010015050"
 3720 DATA"55530660605050050050"
 3730 DATA"55500606605050042050"
 3740 DATA"55520201003042005050"
 3750 DATA"00060555537004205050"
 3760 DATA"06000500000000505050"
 3770 DATA"04555301620001305050"
 3780 DATA"00000006052013005050"
 3790 DATA"55555550005030004030"
 3800 DATA"700000000005000170000"
 3810 :
 3820 DATA"01552015520155200000500501552
01552055520050000500005000000000500505000
0500505005005550050420504204520555505550
05555050050050000500505005000000500505000
0500505005004553045530455300000500504553
0300405553"
```

POSTBAG

ASTAAD ON THE MODEL B

With reference to David Wilson's letter in BEEBUG Vol.8 No.1 I would say that the original ASTAAD is very wasteful of memory and a lot of space can be made available simply by writing bits of it more economically. Also more space can be saved by using a separate initialisation program to set up the function keys etc., and then chain the main program. If sideways RAM is fitted, quite a lot of space can be saved by calling a routine from there to carry out the large text printing (and also speed it up a lot).

It's surprising how much the humble Beeb can achieve.

C.J.Collins

Mr Collins is quite correct in what he says with regard to more economical use of memory, but ASTAAD3 on the Master is able to exploit both sideways and shadow RAM, take advantage of the lower value for PAGE on disc systems (&E00), and benefit from the additional graphics and other facilities available through Basic IV. ASTAAD3 on a model B may not be impossible, but the amount of time and effort would be considerable. However, if anyone wants to take up this challenge we would always be prepared to consider publication on a future magazine disc/tape.

MAGAZINE STANDARDS

I feel I really ought to write and protest at the generally poor standard of information printed in your (and other) magazines - for example the useless Wordwise Plus hint on page 61 of Vol.7 No.10). Doesn't anyone check the copy before you print thousands of magazines and send them out?

To make matters worse, I bought another well known BBC computer magazine, and that has completely left out a program, and also got the index to the magazine wrong.

I have been concerned with computing for over 25 years and have watched standards fall continuously. I would not be in my job long if I installed programs like those printed in magazines these days. Can we all try a little harder?

R.A.Winter

We have always been very much concerned with the standards of programs and other information published in both BEEBUG and RISC user. All articles and programs are read and checked for accuracy by four people, and we have taken steps recently to improve accuracy yet further.

With the best intent, that does not prevent errors from occurring. We would always suggest that any reader who suspects an error and has thoroughly checked their copy of a program to the best of their ability should contact us by phone. And we do publish any updates (under the heading of 'Points Arising') as soon as information is available.

Having said that, in all honesty the majority of problems encountered by readers trying to get magazine programs to run do arise from mis-typing of the program listing on entry, or failing to read sufficiently the accompanying article. We also get queries from readers who have failed to identify the vertical bar character (|) which appears in listings from time to time. This is the shift character to the left of the cursor keys.

And yes we will try harder.

MAGAZINE TAPES AND DISCS

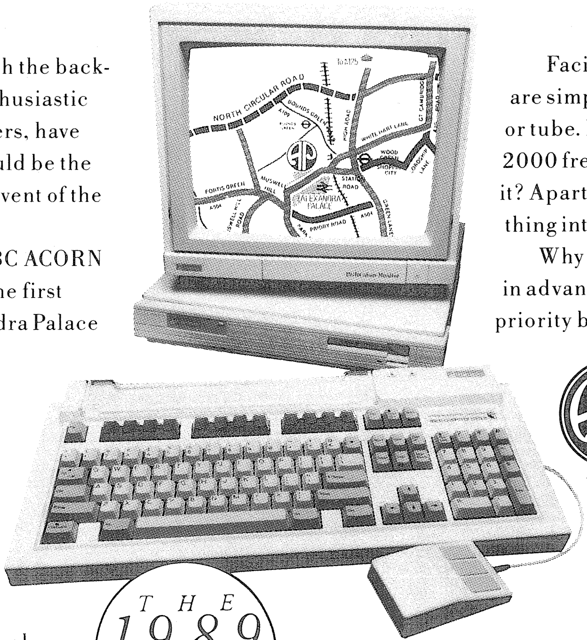
As a member from the beginning of BEEBUG, may I voice my protest at your recent Editor's Jottings regarding "more programs". It has generally been accepted that the magazine is the core of BEEBUG, and that such things as the monthly discs or tapes are something of a convenience for those with neither the time nor the inclination to type in the programs from the magazine. Now we are informed that the recent Video Cataloguer has been extended or improved, but only if you buy the disc!

May I suggest you reconsider this new 'sell a disc' policy, and keep the magazine subscribers as the centre of BEEBUG.

W.S.Turner

The magazine will certainly remain the core of BEEBUG. However, if the quoted program had not been published on the magazine disc or tape it would not have been published at all. There are a variety of reasons why it may be desirable to publish items only on the disc or tape, and we shall not refrain from doing so where it seems appropriate.

BBC Acorn User Show



BBC ACORN USER, with the backing of the BBC and the enthusiastic support of Acorn Computers, have together planned what could be the most exciting computing event of the year.

The new, all-action BBC ACORN USER SHOW is back, at the first home of BBC TV – Alexandra Palace featuring:-

- new product launches
- new software
- massive computing exhibition
- informative seminars
- technical clinics
- workshops
- demonstrations

All this and more will make the new BBC ACORN USER SHOW a real must for everyone interested in computers and their applications.

There are daily competitions, free draws with fabulous prizes to be won and celebrity guest appearances.

THE 1989 BBC ACORN USER show

Facilities at the tastefully restored Alexandra Palace are simply superb and it's so easy to get to by road, rail or tube. If you're driving you'll be glad to know there are 2000 free car parking spaces. So why not make a day of it? Apart from the beautiful grounds you'll find something interesting and exciting around every corner.

Why not save time and money by booking your ticket in advance? Cut the coupon now and ensure your priority booking.

ALEXANDRA PALACE · LONDON
Friday July 21st 3pm-9pm
Saturday July 22nd 10am-6pm
Sunday July 23rd 10am-6pm

B O O K   N O W   A N D   S A V E

Advance Tickets: Adults £2.50 Under 16s £1.50

At the door: Adults £3.50 Under 16s £2.50

Please send me ______ Adult Tickets at £2.50 each

Please send me ______ Under 16s Tickets at £1.50 each.

I enclose cheque/PO for £________ payable to SAFESELL Ltd., Market House, Cross Road, Tadworth, Surrey KT20 5SR.

DON'T SEND CASH.

Name ____________________

Address ____________________

RUI

This month's hints features a number of procedures. If you have any useful short procedures (or functions) then we would be pleased to consider them for our Hints & Tips page. We pay £5 for each hint published and £15 for the star hint.

***** STAR HINT *****

TELETEXT CODES IN MASTER EDIT

David Stevens

When using the Master's EDIT program you can include graphics colour codes by moving to the required position, selecting cursor editing (press Shift-Copy), and then typing Shift-f1 for red (code 145), Shift-f2 for green (code 146) and so on. This also enables Shift-f8 (conceal graphics) and Shift-f9 (join graphics). In Basic, these codes are normally generated using the Ctrl key and a function key.

To permit text colour codes to be entered from within Edit, first type:

```
<f1>            Command Line
*FX228,128
Escape
```

Then, whenever you want a text colour code to be entered, proceed as follows:

```
Shift/Copy      Cursor Edit Mode
Ctrl-Shift-<fkey>
Escape
```

Ctrl-Shift-f1 will generate red text (code 129), with f2 giving green text (code 130) and so on. For example, to enter the code for cyan use:

```
Ctrl-Shift-f6
```

To insert the double height character (code 141) substitute *FX228,140 in the first part and then use Ctrl-Shift-f1 for each instance that it is needed. In each case, the value following *FX228 sets the base value for key f1.

MORE ON BOOT FILES FOR THE MASTER

G.Cooley

Further to the hint in Vol.7 No 10 on the construction of conditional boot files for loading SWR images, an alternative method of loading ROM images is as follows. Before constructing the boot file you will need to determine for each ROM the value of its type byte (the sixth byte). To do this, use *DUMP to dump the ROM image to the screen, and note the hex value of the sixth byte. Then substitute this value in the corresponding line of the boot file. The address for this value is &2A1 plus the socket number.

```
 1 *BASIC
 2 *DIR <drive and directory as required>
 3 *SRLOAD ROM1 8000 4 Q
 4 *SRLOAD ROM1 8000 5 Q
 5 *SRLOAD ROM1 8000 6 Q
 6 *SRLOAD ROM1 8000 7 Q
 7 ?&2A5=<value1>
 8 ?&2A6=<value2>
 9 ?&2A7=<value3>
10 ?&2A8=<value4>
11 CH."AnyProg" or <any star command>
```

With this method, language ROMs will initialise themselves, but service ROMs which require workspace will still need to be initialised with Ctrl-Break.

COLOUR MIXING WITH A MASTER

David Stevens

The following procedure can be included in any mode 2 program to allow the four extended colour fill patterns to consist of a new colour mix.

```
1000 DEF PROCset_ecf(pat_no,col1,col2)
1010 A0=col1 MOD2:A1=col2 MOD2
1020 A=col1 DIV2:B=col2 DIV2
1030 A2=A MOD2:A3=B MOD2
1040 A4=A DIV2:A5=B DIV2
1050 A=32*A5+16*A4+8*A3+4*A2+2*A1+A0
1060 B=32*A4+16*A5+8*A2+4*A3+2*A0+A1
1070 C=pat_no+1
1080 VDU23,C,A,B,A,B,A,B,A,B
1090 ENDPROC
```

There are three parameters, *pat_no* the pattern number in the range 1 to 4, and the two colours to mix, *col1* and *col2*, both in the range 0 to 15.

Personal Ads

BEEBUG members may advertise unwanted computer hardware and software through personal ads (including 'wants') in BEEBUG. These are completely free of charge but please keep your ad as short as possible. Although we will try to include all ads received, we reserve the right to edit or reject any if necessary. Any ads which cannot be accommodated in one issue will be held over to the next, so please advise us if you do not wish us to do this. We will accept adverts for software, but prospective purchasers should ensure that they always receive original copies including documentation to avoid any abuse of this facility.

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

Printer: Radio Shack DMP-110 dot matrix printer, sheet and tractor feed, in original packaging complete with manual, three printer ribbon cartridges and BBC micro lead £70. Tel. (0604) 51778.

Cumana 40T S/S disc drive boxed, utilities and manual £35. ADFS ROM and manual (unused) £18. Tel. 01-318 3995.

Sanyo 12" green screen monitor £30. Tel. 01-942 1766.

View 3.0 ROM complete with manual £35. Tel. (0273) 559593.

BEEBUG magazine cassettes 62 complete to May 1989, all with indexed inlays (value £140) £55, can be transferred to disc. More than 100 other cassettes for sale for £1 each (list available), computer magazines in binders; Micro User Vols 1-4 complete, 48 issues in 4 binders £30, Acorn User 1982-1986, 32 issues + 3 supplements in 3 binders £25. Tel. (0727) 61835.

Acorn User back numbers 1 to 62 (October 87), 18 and 49 missing, only £25 the lot plus postage and packing. Tel. (0256) 24880 after 7pm.

AMX Stop Press 2 ROMs plus system disc, font disc, clip art £25, Watford ROM Manager plus manual £9, various games; Elite, Revs, Repton, Play It Again Sam 2,6, £8-10 each. Tel. (0652) 650324.

Software for sale, all original cassettes including educational, business and leisure. Send S.A.E for list to; Mr R G Gill, Demmal Cottage, Newton Arlosh, Kirkbride, Cumbria, CA5 5HE or Tel. (06973) 51445.

Elite Zipstick joystick for BBC B, BBC B+ or M128, excellent condition, less than 6mths old £7. Tel. 01-698 0215.

WANTED: Microsoft's Flight Simulator, must be version 2.12. Tel. (0252) 713077.

Assorted BEEBUG magazine tapes; 33 magazine cassettes (vols 2-7) including most of BEEBUG's best programs (ASTAAD 3, Filer, Business Graphics etc) all originals, cased as new. Original cost over £80, will accept offers around £40 for the lot. Tel. (0279) 813463 after 6pm.

Business Ad

Software in BBC BASIC

Full-time position in small business in Beaconsfield for someone able to develop efficient programs for process monitoring. Some knowledge of op amps and TTL essential. Training available.

Phone: Ian McFarlane
Beaconsfield Instrument Co.
0494 677833

Brother EP-44 Personal printer with BBC lead, in original packaging with user manual, doubles as LQ typewriter and second keyboard £150 o.n.o. Tel. (0722) 29341.

Master 128, incl. MOSplus, ADT, Exmon II, Hyperdriver ROMs £275. 512 board + Dabs Shareware, user guide £55. BBC B OS1.2 issue 7 + DFS, Wordwise plus £175. Morley Teletext adapter + ATS ROMs £45. Taxan Supervision 620 RGB monitor £175. 80T DS disc drive (mains powered) £60. Super Art for Master and BBC £25. Mouse £10. PMS Genie £35. Mega 3 ROM £40. View Professional £40. BEEBUG Master ROM £15. All manuals included. Tel. (08444) 4633.

Archimedes 310 + colour monitor + podule backplane and some games £675 o.n.o. Tel. (0727) 67354 after 6pm.

Master 512 co processor DOS+ 2.1, c/w Acorn mouse and GEM £75. Clares Art-Room (80T ADFS) + Nidd Valley Digimouse for Master 128 £25. Watford "Solderless Sideways ROM board" c/w battery back up, 2X6264 CMOS RAM, read/write inhibit toggle switches for BBC B £25. WWPlus ROM + manuals £10. Tel. 031 449 3869.

Archimedes 310M with colour monitor plus Fortran £950 o.n.o. Tel. (0902) 751762.

BBC Master 128k, Turbo Co-processor, Music 500, 3.5" D/D, light pen, WYSIWYG+, Quickshot joystick, Advanced User Guide and Reference manual part 2, double plinth, quad cartridge, Trivial Pursuit, Starquake, Office Mate & Office Master, 20 3.5" discs, dust cover. All fully boxed, all in immaculate condition £600. Tel. (0924) 848909 eves.

BBC B issue 4, Acorn DFS £190. 5.25" single sided Cumana D/D £30. 5.25" double sided Cumana dual D/D £135. Kaga Taxan printer £130. Panasonic printer KX-P1081 £100, all equipment in excellent condition. Tel. (02407) 5670.

Printer Epson RX80, boxed with instructions etc £90, D/D Cumana CS400E D/S 80T and CS100 S/S 40T, wired as a double drive, both with own PSU £90, original software; Computer Concepts Printmaster ROM £10; Beebug Toolkit ROM £7, Wizard Joysticks ROM (converts keyboard only games to play on joystick) £5. In original packaging with documentation. Tel. (0786) 61501.

For BBC B Stop Press Desktop Publishing software, boxed brand new (without mouse) £22. Tel. (0223) 321128.

BBC B issue 7 DDFS and joystick £260, Viglen PC case and ROM cartridge system £25, 2xD/S D/D 40T £80, med. res. monitor (Cub 1451) £185, ATPL ROM board with 16k RAM, printer buffer and utilities £30, Watford Shadow RAM £40, Interword £30, Interbase £40, Spellmaster £35, Teletext adaptor, latest ROM £50, Solidisk 32k RAM £20, Advanced user Guide £5, EPROMs 21V, 16k £3.50 each, Acorn Atom £10, many discs (most full of software). Any reasonable offers considered. Write to: 2, Buttermere Drive, Allestree, Derby.

Acorn DD SS 40T with system disc, excellent condition £25 o.n.o. BCPL chip, manual, keystrip, maths package and discs £10. Tel. (0702) 586536.

BBC B issue 7 £150, Watford dual DSDD 40/80 drives £150, Wordwise £12, discs 30 for £6, double joystick interface £3, all for £310. Tel. (05806) 2068.

Microlink Multispeed intelligent modem. V21,V23 and V22 full duplex. Auto-dial, auto answer, fully hayes compatible, brand new. Original packaging never used £150. Tel. (0742) 376982.

WANTED: Advanced Disc User Guide. Tel. (0344) 776674.

View Guide and Utilities disc, 34 page guide and tutorial, includes 7 printer drivers + printer driver generator on 80T 5.25 DFS disc. Includes template files and 64k printer buffer. Beebug Astaad CAD, 80T DFS enhanced version, joystick or mouse control sample files and 8 page booklet. Max Desktop ROM 16k ROM giving full icon control of DFS or ADFS files, complete with 28 page manual and reference card. Chaffeur Mouse Dnr 40T DFS allowing adaption of non mouse software to run under mouse or joystick control, ROM images on disc and 10 page instruction manual. Arcade game creator 40/80T DFS disc allows arcade games to be created, 10 page manual. Acorn User 1987 compilation disc, Best of Acorn User from 1987 over 25 quality programs and 16 page booklet. All above costs £75 new, will accept £35. Tel. (0705) 527957 after 6pm.

Aries B20 shadow RAM board £20. Tel. 031 334 8723.

Solve your compatibility problems using **DOS+ Problem Solver**

It enables the 512 board to run most IBM-PC programs that otherwise wouldn't run, emulates the missing keys that BBC model B users don't have and allows the saving of high-res screens from any program.

PRICE: £24.95 (including program, user manual and 1 year free user support)

"What a great program Problem Solver turned out to be.
... if you have a program that you use at work on the office PC and you want to get it up and running on your model B or Master, it will seem worth it to you.
And if you want to play PC games, then you really will notice a difference."
BBC Acorn User , May 1989, pages 129-130 (editorial review)

Lots of programs like Cat, Jet, Digger, Artwork's Strip Poker, ELECTRONIC ART's Golf, Driller, Dark Side, Impact, Charlie Chaplin, Test Drive, Infiltrator, StarQuake, Bushido, Tennis, Frogger, all versions of MicroSoft's Flight Simulator, Osbit, 688 Atack Sub, Defender of the Crown, Quadralian, Yes Chancellor, Ancient Art of War, Adventure Writter, Dream House, AFT, Droege, Dream, Delux Paint 2, Fontasy 2.08, News, PC Tutor, Lotus 123 2.0, Turbo Calc, Mandelbrot Generator, Prospero Pascal, Turbo Pascal 4.0, Turbo C 1.5, Turbo Prolog, Prolog2, PC File+, Galaxy, Trendtex/2, Homebase 2.15, Mindreader, DBASE III plus, Pipedream, and many more will run like on an ordinary IBM PC.

Available from Shibumi Soft, R. Prof. Camara Sinval 138, 4100 PORTO - PORTUGAL

Payment can be made with an ordinary English cheque. Please add £3 for postage and packing.
Faulty disks will be replaced.

BBC Master 128, Phillips colour monitor CM8833 with TV tuner 7300, HCR external ROM expansion box (24 ROMs, 64k battery backed RAM) all with original boxes, 2 Teac DS 40/80T disc drives one with PSU, Marconi Tracker Ball, computer dust cover, v.g. double shelf plinth, C.C. Interword, Spellmaster, Watford - Quest Paint/Conquest/Quest Font, BEEBUG - Masterfile 2, Quickcalc, Printwise, Hershey Characters, Technomatic - Novacad, Reference manuals Pt 1&2 £750. Tel. (0243) 263645 eves.

2 40T SS DD inc. PSU's, printer lead, handbook, and utilities disc £100. 6502 second processor c/w handbook, DNFS & Hi-Basic ROMs, boxed as new £90. Acornsoft GXR-B ROM c/w handbook, demo tape etc £14, Beebugsoft Toolkit ROM £5, Wordwise plus ROM c/w demo tape and all instruction manuals £25. Tel. (0384) 373928.

BBC B OS 1.2, DNFS, Speech System, ATPL Board, 32k Sideways RAM, Write protect switch, Volume control, Discs, ROMs Mags £250. Tel. (0324) 558692.

512 Board, 4 DOS discs, Manuals £100, Dumpout 3 £20, AMX art program with mouse £60, Lord of The Rings £15, All good condition, will split. Tel. (0206) 230255.

WANTED: Century Course on Micro Computing for the BBC. Title may not be totally acurate but beleived correct. Tel. (0777) 818298 eves.

Wordwise plus £25, System Delta £35, System Delta Reference Guide £10, Monitor ROM £10, 65CO2 £5, Small C v1.5 (80T) £30, Offers invited for books; A Book on C (Berry & Meekings), 6502 Assembly Language Programming (Leventhal), Assembly Language for BBC (Birnbaum), Forth on BBC (de Grandis-Harrison), Functional Forth on BBC (Allen), Forth Programming (Scanlon), Adv. Basic ROM User Guide (Pharo), Mastering the Disc Drive (Snee), Disc Systems for BBC (Sinclair). Tel. (0535) 602960 after 8pm.

Archimedes 310 entry with RISC OS, Manuals, discs, £650. Write to: Lorcan Mongey, 56 Salisbury Court, Belfast, BT7 1DD.

WANTED: 5.25" DD, Cumana CS400S/E preffered (already have CS400S), but will consider any single D/S 40/80 unit with PSU. Also need an Epson, or compatible printer - 80col, with tractor unit and sheet feeder. Tel. (09295) 51450 day 3670 eves.

M128, Micro Vitec med. res. monitor, Twin DS DD Cumana drives with PSU, Epson LX80 printer, manuals Vol 1&2 £650 o.n.o. Tel. (09904) 3025.

Torch Z80 2nd processor, Perfect Writer, Calc, Spell, and File, Sage Professional accounts (worth £350 alone) all manuals. £250 o.n.o. Tel. 041 332 6183.

Solidisk 128k RAM disc plus 32k SWRAM plus 5 discs software unused £100. Tel. (0903) 203858.

BEEBUG MEMBERSHIP

Send applications for membership renewals, membership queries and orders for back issues to the address below. All membership fees, including overseas, should be in pounds sterling drawn (for cheques) on a UK bank. Members may also subscribe to RISC User at a special reduced rate.

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

BACK ISSUE PRICES (per issue)

Volume	Magazine	Tape	5"Disc	3.5"Disc
1	£0.40	£1.00	-	-
2	£0.50	£1.00	-	-
3	£0.70	£1.50	£3.50	-
4	£0.90	£2.00	£4.00	[illegible]
5	£1.20	£2.50	[illegible]	[illegible]
6	£1.30	£3.00	[illegible]	[illegible]
7	£1.30	[illegible]	[illegible]	[illegible]

See enclosed leaflet for Special Summer Offer on Back Issue magazines, tapes and discs!

All overseas items are sent airmail. We [illegible] accept [illegible] for official [illegible] back [illegible] that [illegible] £1 handling [illegible] orders under £10 which require an invoice. Note that there is no VAT in magazines.

Dest[illegible]	[illegible]	[illegible]ond Item
[illegible]	[illegible]	30p
[illegible]	60p	50p
E[illegible]	£1	£1
El[illegible]	£2	[illegible]

POST AND PACKING

Please add the cost of p&p as shown opposite.

BEEBUG
Dolphin Place, Holywell Hill, St.Albans, Herts AL1 1EX
Tel. St.Albans (0727) 40303, FAX: (0727) 60263
Manned Mon-Fri 9am-5pm
(24hr Answerphone for Connect/Access/Visa orders and subscriptions)

BEEBUG MAGAZINE is produced by BEEBUG Ltd.

Editor: Mike Williams
Assistant Editor: Kristina Lucas
Technical Editor: David Spencer
Technical Assistant: Glynn Clements
Advertising: Sarah Shrive
Production Assistant: Sheila Stoneman
Membership secretary: Mandy Mileham
Managing Editor: Sheridan Williams

CONTRIBUTING TO BEEBUG PROGRAMS AND ARTICLES

We are always seeking good quality articles and programs for publication in BEEBUG. All contributions used are paid for at up to £50 per page, but please give us warning of anything substantial that you intend to write. A leaflet 'Notes to Contributors' is available on receipt of an A5 (or larger) SAE.

Please submit your contributions on disc or cassette in machine readable form, using "View", "Wordwise" or other means, but please ensure an adequate written description is also included. If you use cassette, please include a backup copy at 300 baud.

In all communication, please quote your membership number.

BEEBUG Ltd (c) 1989
ISSN - 0263 - 7561

Printed by Newnorth-Burt Ltd (0234) 41111

Magazine Disc/Cassette

JULY 1989 DISC/CASSETTE CONTENTS

MAGIC LANTERN SLIDES - two demonstration programs showing computer implementations of old fashioned lantern slides.

RUNNING FOUR TEMPERATURES - a calibration program and a plotting program for use with this month's four-channel temperature probe project.

FOREIGN LANGUAGE TESTER (Part 2) - monitor your progress in foreign languages with this month's program, plus a utility to print out the redefined keyboard layout, new character definers for Portuguese, Italian and the Scandinavian languages, and a repeat of all the programs from part one.

AN ADFS DISC SECTOR EDITOR - use this utility with your Master to edit disc sectors directly.

MATHEMATICAL TRANSFORMATIONS (Part 1) - the latest program in our educational series deals with mathematical transformations.

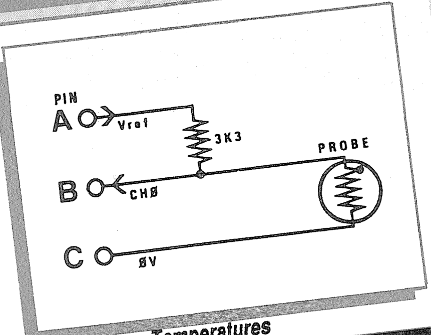
NIMBLE TYPER - this program allows the user to define a range of abbreviations which will be automatically expanded in full on entry.

WORKSHOP - a fast disc formatter for the ADFS based on the principles described in recent Workshops.

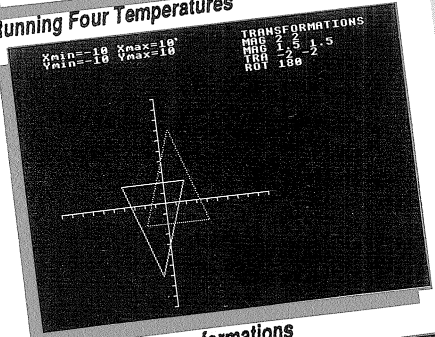
BATCH BYTE LOADING - streamline data transfers from disc with this handy technique.

EGGHEAD - this month's disc/tape includes a challenging arcade-style game. Crush the nasties with the moveable boulders to win.

MAGSCAN - bibliography for this issue (Vol.8 No.3).

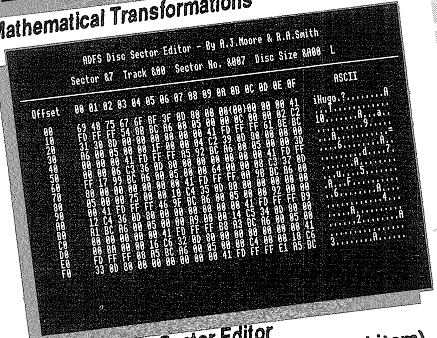


Running Four Temperatures



```
Xmin=-10 Xmax=10          TRANSFORMATIONS
Ymin=-10 Ymax=10          MAG 2 2
                          MAG 1.5 1.5
                          TRA -2 -2
                          ROT 180
```

Mathematical Transformations



```
        ADFS Disc Sector Editor - By A.J.Moore & R.A.Smith
   Sector &7  Track &00  Sector No. &007  Disc Size &A00  L

                 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F        ASCII
Offset
  00             69 48 75 67 6F BF 3F 0D 80 00 00{00}00 00 00 41   iHugo.?........A
  10             FD FF FF 54 8B BC A6 00 05 00 00 0C 00 00 02 C2   ...T............
  20             31 30 8D 00 00 00 00 00 00 41 FD FF FF 61 8E BC   10.......A...a..
  30             A6 00 05 00 00 1F 00 00 04 C2 39 0D 80 00 00 00   ..........9.....
  40             00 00 00 41 FD FF FF A5 92 BC A6 00 05 00 00 3D   ...A...........=
  50             00 00 06 C3 36 0D 80 00 00 00 00 00 00 41 FD FF   ....6........A..
  60             FF 17 99 BC A6 00 05 00 00 64 00 00 08 C3 37 0D   .........d....7.
  70             80 00 00 00 00 00 00 41 FD FF FF 0A 9B BC A6 00   .......A........
  80             05 00 00 75 00 00 10 C4 35 0D 80 00 00 00 00 00   ...u....5.......
  90             00 41 FD FF FF 46 9F BC A6 00 05 00 00 92 00 00   .A...F..........
  A0             12 C4 36 0D 80 00 00 00 00 00 00 41 FD FF FF B9   ..6........A....
  B0             A1 BC A6 00 05 00 00 A9 00 00 14 C5 34 0D 80 00   ............4...
  C0             00 00 00 00 00 41 FD FF FF B8 A3 BC A6 00 05 00   .....A..........
  D0             00 BA 00 00 16 C6 32 0D 80 00 00 00 00 00 00 41   ......2........A
  E0             FD FF FF 08 A5 BC A6 00 05 00 00 C4 00 00 18 C6   ................
  F0             33 0D 80 00 00 00 00 00 00 41 FD FF FF E1 A5 BC   3........A......
```

An ADFS Disc Sector Editor

All this for £3. 50 (cassette), £4.75 (5" & 3.5" disc) + 60p p&p (30p for each additional item).
Back issues (5.25" disc since Vol.3 No.1, 3.5" disc since Vol.5 No.1, tapes since Vol.1 No.10) available at the same prices.

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

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

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

THE Archimedes A3000

The New BBC Micro A3000

Beebug are pleased to offer the new BBC Micro, from Acorn Computers. It is ideal for home, education and business use and includes RISC OS, the new multi-tasking operating system. With the PC Emulator it will also run over 90% of DOS Software, including MS Word, Wordperfect, Dbase III, Lotus, etc.

AVAILABLE July 21st

The A3000 will be available in limited quantities from July 21st, when it will be launched at the Acorn User Exhibition, Alexandra Palace. Demand for the A3000 is very high, so if you wish to be one of the first to own one, you should reserve it now.

GUARANTEE

Beebug have over 7 years experience of providing support on the original BBC Micro. If you need help, you can rely on us, in addition to your full 1 year warranty on the A3000 itself. Our prices are competitive, but if you can get a better deal on the A3000 we will match it.

2 FREE SHOW TICKETS

If you wish to collet your A3000 at the Acorn User Exhibition reserve it now and we will send you two admission tickets free of charge. If you cannot get to the show, your computer will be despatched by courier on Monday 24th July.

0% FINANCE

We are also able to offer 0% finance on the A3000, with a deposit of only £71.35. If you choose this option the members special offer is 10 x 3.5" Discs and a Lockable Disc Storage Box.

TRADE IN

If you have a BBC Micro, Master or Archimedes and wish to upgrade to an A3000, we will take in your old computer in part exchange.

MEMBERS SPECIAL OFFER

If you are not yet a member, subscribe now to RISC User, The Archimedes magazine (UK £14.50) and when purchasing your A3000 you will then also receive Artisan, Pacmania, 4-Way Spike Protection Socket Strip, 10 x 3.5" Discs and a Lockable Disc Storage Box. Alternatively you may choose any other items free of charge up to 10% of the cost of the computer.

A3000 PRIORITY ORDER FORM

I wish to order an A3000 Base/Colour System @ £746.35/£999.35 (inc VAT) ☐

I will collect it at the Acorn User Show, please send me 2 tickets free of charge ☐

I wish it to be despatched and enclose £7.00 (UK) carriage/I will collect it from the showroom. ☐

BEEBUG/RISC USER membership number

I wish to subscribe to RISC USER at £14.50 (UK). ☐

A cheque for £________ is enclosed.

Please debit my Access/Visa/Connect account with £________

Card No. ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐ Expiry Date______/______/

Please send further details on the A3000 ☐ 0% Finance ☐

12/24/36 Months Finance ☐ Trade In ☐

Name________________________________

Address________________________________

________________________________ Post Code ________________

Send to:
BEEBUG LTD, Dolphin Place, Holywell Hill, St. Albans, Herts AL1 1EX Tel : 0727 40303 Fax: 0727 60263

BEEBUG LIMITED

Dolphin Place, Holywell Hill, St. Albans, Herts. AL1 1EX ☎ 0727 40303 Fax: 0727 60263