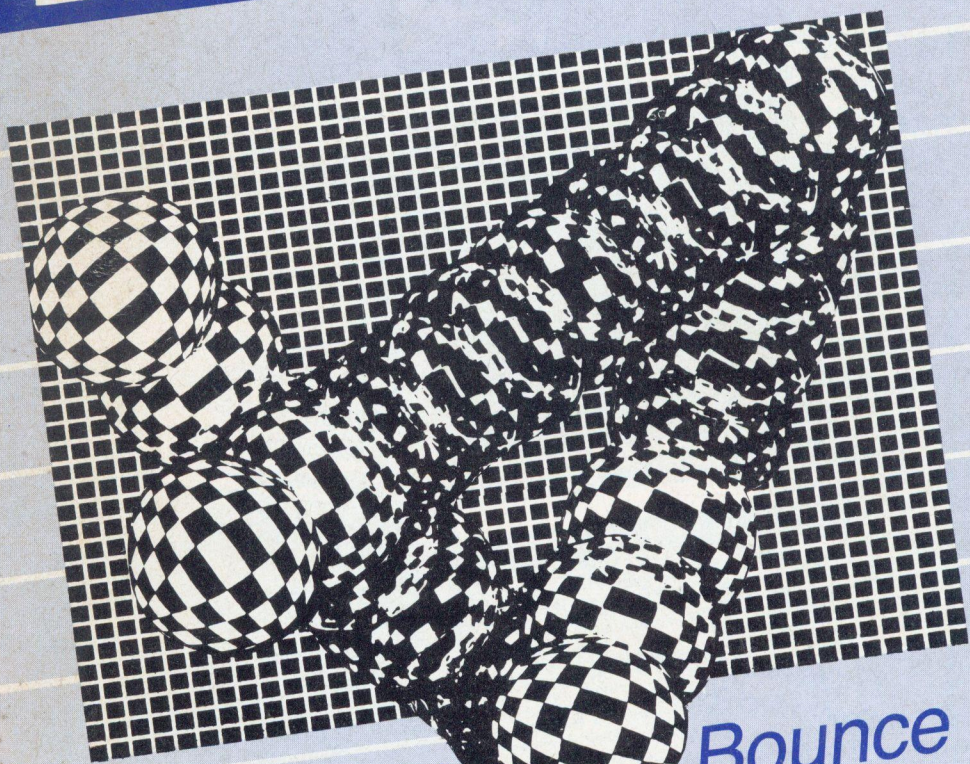


Vol.6 · No.5 · October 1987

BEEBUG

FOR THE
BBC MICRO &
MASTER SERIES



Giant Bounce

- DYNAMIC MEMORY DISPLAY FOR THE MASTER
- ACORNSOFT 'C' REVIEWED ● CATALOGUING DISCS

FEATURES

The Ghost Host	9
Cataloguing Discs With Filer	12
Super Function Keys	16
Archimedes Product Update	18
Barry Christie Visuals - Giant Bounce	20
BEEBUG launches RISC User	23
BEEBUG Education	24
Exploring Assembler (Part 4)	26
Workshop - Precision Arithmetic	30
The Master Pages - Dynamic Memory Editor	41
Master Hints	46
First Course - What's The Function of a Function?	50
Regression Analysis	53
Spooks And Spirits	60

REVIEWS

Acornsoft 'C'	6
Adventure Games	47
System Delta Applications	48
Pearl Data Logger	58
Modem Master	65
ADU ROM from Pineapple	66

REGULAR ITEMS

Editor's Jottings	4
News	4
Supplement	33-40
Master Hints	46
Postbag	67
Hints and Tips	68
BEEBUG Technical	69
Subscriptions & Back Issues	70
Magazine Disc & Cassette	71

HINTS & TIPS

GENERAL

InterWord Multi-File
Giant Scroller
Making Your REMs Stand Out
Quick ADFS & DFS Enable
Magscan Data Files
One-Line Metronome
InterWord and Long Commands

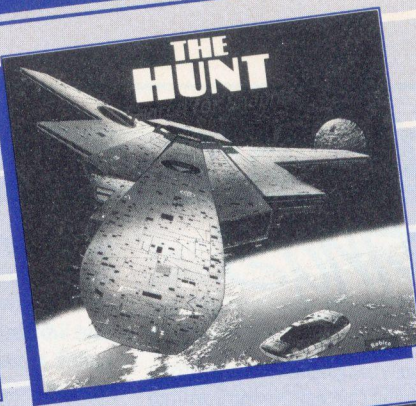
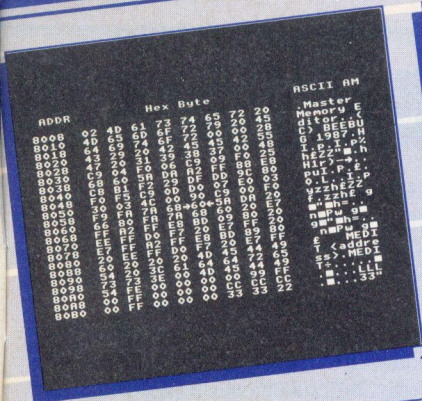
MASTER

Screen Save to Sideways RAM
CMOS RAM Antitheft Device
Editor Comment Stripper
Definition Lister

PROGRAM INFORMATION

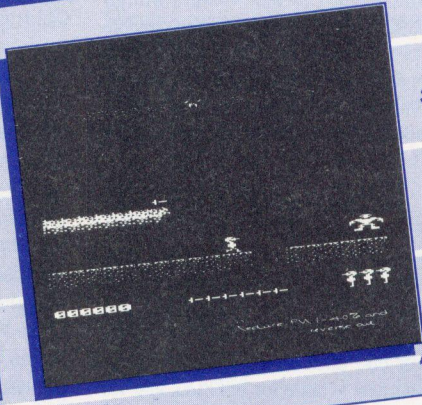
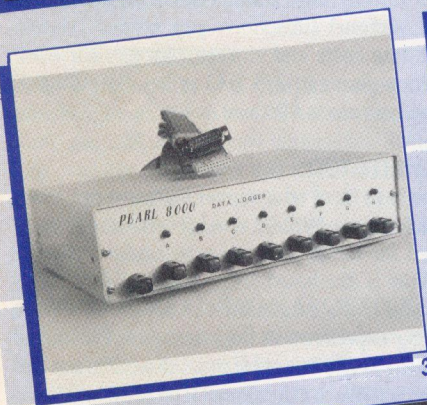
All programs listed in BEEBUG magazine are produced direct from working programs. They are listed in LISTO1 format with a line length of 40. However, you do not need to enter the space after the line number when typing in programs, as this is only included to aid readability. The line length of 40 will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

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



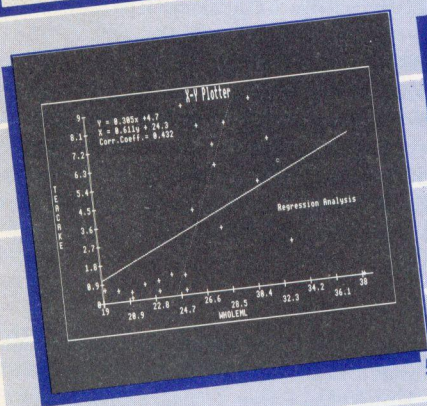
1. Master Memory Editor screen

2. REVIEW: The Hunt - Search for Shaun



3. REVIEW: Pearl 8000 Data Logger

4. Spooks and Spirits - Arcade Game



5. Regression Analysis

6. Magazine Disc Menu screen






5.





6.

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

Computer System

Filing System

- Master (Basic IV) 
- Compact (Basic VI) 
- Model B (Basic II) 
- Model B (Basic I) 
- Electron 

- ADFS 
- DFS 
- Cassette 
- Tube Compatibility**
- Tube 

Editor's Jottings

BEEBUG SUPPORT FOR ARCHIMEDES

BEEBUG is launching a new magazine and supporting group for Archimedes users. This new venture will go under the name of RISC User and will, we believe, enable us to offer the best possible support in the future to both existing BBC micro and Master Series users, and to the new Archimedes. Full details of RISC User, and special rates for joint or converted membership for BEEBUG members, are given on page 23.

BEEBUG magazine, which was increased by eight pages from the August/September issue to provide extra coverage of Archimedes, will remain at this new larger size. We shall continue to provide essential news coverage of Archimedes, but the majority of the extra pages will henceforth be devoted, like the rest of the magazine, to the needs and interests of BBC micro users. RISC User is an important and exciting new venture for BEEBUG, and we hope that many BEEBUG members will take advantage of the special joint subscription rates available to them and help us to make RISC User a real success.

COMPUTER GAMES

As a result of feedback from BEEBUG members we have further decreased our coverage of the computer games market in BEEBUG over recent issues. In future, games reviews will appear only at the most appropriate times (e.g. pre-Christmas), and games programs will be published in BEEBUG less often, and only when we can offer games of the highest quality (within the constraints of the magazine). Some members have more recently expressed a view that they would like to see more games content than in recent issues. What is your view? Have we made the right decisions?

NEW LOOK TO BEEBUG

We have introduced a desktop publishing system for originating BEEBUG (using a Macintosh SE with laser printer), and with this issue we have also provided a new look for the inside and outside covers. We believe that this will make the magazine easier to read, while at the same time providing a better looking magazine. All this has taken some time and effort and we are sorry if BEEBUG is arriving a little later than usual.

Mike Williams

News..News..New

MICRONET GOES FOR GOLD

Telecom Gold, BT's message service, is now available to all Micronet subscribers via a Prestel Gateway called Interlink. Interlink users get a Telecom Gold mailbox giving access to all Telecom Gold services. Micronet subscribers can also choose between accessing Telecom Gold via the Interlink Gateway in 40 column Prestel mode, or directly in 80 column mode. Registration is free but Micronet subscribers pay normal Telecom Gold on-line costs and storage charges. For more information phone 01-278 3143 or key *INTERLINK when you next access Micronet.

QUITE WRITE TOO

Latest release from educational software producers Soft-Teach is Quite-Write, a word processor specifically designed for junior and lower secondary schools. Quite-Write is disc-based, using a mode 7 WYSIWYG screen, and is the culmination of seven years word processing experience in the classroom. The cost of Quite-Write is £20 but substantial discounts are also available. Soft-Teach Educational is on (0985) 40329.

MORE MASTER CARTRIDGES

'Romboard 4' from Vine Micros provides yet another option for Master users who have more ROMs than sockets. Romboard 4 will house up to four ROMs including the Computer Concepts Inter-Link series. A read/write switch is fitted as standard and the unit costs just £29.95 inc. VAT. Vine Micros are on (0304) 812276.

SOFT IN THE ED

Parents and Teachers looking for good educational software to interest and stretch their children need look no further than Edsoft. Operating like a book club, Edsoft offers selected educational software six times a year to Edsoft members at attractive prices. There are some titles of interest to adults as well. For more information contact Edsoft on (0442) 41221.

SAVE YOUR BEEB FROM DROWNING

If your micro suffers from frequent spillage of tea or coffee or even worse, then Seal'n Type Spillcovers from

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

Kador could be the answer. The flexible vacuum formed PVC seal completely covers the keyboard, and protects your machine from dirt and spillage of all kinds. Currently available for the BBC B and B+ (and Amstrad machines) the Seal'n Type costs £6.50 inc. V.A.T. and postage. Kador also manufacture acoustic printer box/stands, screen filters and mouse pads. For more information ring (0784) 252662.

TWO WAY STRETCH

A & G Electronics is a company specialising in switching boxes. A two way User Port switch costs £24, three way £28, all complete with cables. Switches are also available for printers and RS423 connections. For more information contact A & G at 14 Hurstwood Avenue, Bexley, Kent DA5 3PH, or 322522315 on Prestel.

TURN YOUR BEEB INTO AN ARCHIMEDES

Tubelink, the popular BBC micro database on Prestel, has announced its own Advanced Basic for the BBC micro and Master series. Advanced Basic, previously called Archie Basic, provides a language that is as close as possible to Basic V as used on the latest Archimedes. Advanced Basic is an extension to Hi-Basic and as such needs a system with a 6502 second processor (co-processor) fitted. Tubelink claims that programs written using Advanced Basic are fully compatible with Archimedes. Advanced Basic costs £29.95 on disc, £34.95 on EPROM from Tubelink, P.O.Box 641, London NW9 8TF. In both cases Acorn's Hi-Basic is supplied under licence.

GAMES UPDATE

Games may feature less than they used to, but the games scene is far from dead. Topologika has released Peter Killworth's classic sci-fi adventure, Countdown to Doom, on disc. This version has been greatly expanded with new puzzles and extra locations. Philosopher's Quest, Kingdom of Hamil and Acheton all highly acclaimed adventure games from Acornsoft have received similar treatment and have been re-released by Topologika. All these games cost £17.50 inc VAT and p&p. Contact Topologika on (0733) 244682.

Bug-Byte is another company remaining faithful to the games market. Recent releases include Megarok and Squeakaliser, both at just £2.99. Contact Bug-Byte on telephone 01-439-0666.

Superior Software has released Palace of Magic which Superior confidently expect to be one of the best selling games it has ever produced. Palace of Magic is a vast arcade adventure featuring over 100 screens, and is available on disc or tape for all machines including the Acorn Electron. Prices range from £9.95 for a cassette version, £11.95 for 5.25" disc and £14.95 for 3.5" disc. For more information contact Superior on (0532) 459453.

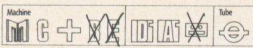
Meanwhile, Domark (of Trivial Pursuit fame) has taken the recent James Bond film, The Living Daylights, as the theme for their latest game of the same name. The game, on cassette only, costs £9.95.

For avid adventure game programmers, newcomer to the games scene, Summit Software, has climbed the heights with ALPS, the Adventure Language and Programming System. ALPS is in the form of an EPROM for the BBC micro and Master series and is designed to make the writing of text-only adventure games as easy as possible. No price yet but Summit software are on (0762) 42510.

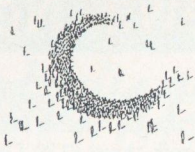
Incentive Software promises two graphic adventure games for the price of one with the release of The Alien from Outer Space and Dragon's Tooth, on a single cassette for just £7.95. Contact Incentive on (07356) 77288.

MORE FOR ARCHIMEDES

The trickle of software and other products for use with Archimedes is beginning to grow rapidly. Acorn has already released ARM 'C' and Pascal, together with text editor Twin; Minerva Systems are in there with Deltabase, and Clares Micro Supplies has released Toolkit. More on these and other products in this month's special feature on releases for the Archimedes market. **B**



ACORNSOFT



'C' is a highly regarded computer language, but until now no full implementation has been available for the Beeb. Ray Hughes has been putting Acornsoft's new version to the test. Benchmarks and additional material provided by BEEBUG.

Product: Acornsoft C
Supplier: Acorn Computers Ltd
Cambridge Techno Park,
645 Newmarket Road,
Cambridge CB5 8PD.
Tel. (0223) 214411
Price: £89.70 inc VAT

For years they said that it couldn't be done. People argued that it was not possible to write a full implementation of the 'C' language for the 6502 processor. But suddenly three separate implementations have been announced, all for the BBC micro:

Acornsoft C
Beebugsoft C
Brasscourt C

In this review I shall concentrate on Acornsoft's version, making occasional comparisons to Beebugsoft's, which, like the former, implements a full Kernighan and Ritchie standard. The Brasscourt version was not available at the time of writing.

SOME PRELIMINARIES

I want to begin, however, with a few words about 'C' and related matters. 'C' is a language in vogue at the moment. Languages come and go, but for the moment 'C' is IN. A major reason for this is that it compiles well, producing reasonably efficient run-time code. This is extremely useful for professional programmers and software houses. Instead of writing applications directly in machine code for a given host processor, they can be written in 'C',

and compiled to produce a stand-alone application package in machine code. Programming time is less, because it is easier to debug 'C' than it would be to debug machine code. And to make a version for a different processor, you need only to recompile the source code on a suitable compiler for the new processor.

The portability of 'C' is greatly increased by the way that the language is organised. For while 'C' is a fully structured language with FOR loops, DO loops and CASE statements, it contains no code for handling input/output functions such as printing and file handling, or even string handling. All these functions are provided by machine-specific disc-based *library routines*. These are *linked* in to the object code during the compiling process. The same 'C' source code program can therefore work equally on an IBM mainframe, a Cray 2 or a BBC micro - they just use different libraries.

COMPILERS AND INTERPRETERS

Programs in high level languages like Basic, 'C' or Pascal must be translated into a series of machine code instruction before they can be executed by a computer's processor. The difference between interpreters and compilers is that interpreters perform this translation one instruction at a time, executing the code that has just been translated, before moving on to translate the next bit. Compilers, on the other hand, translate the whole of a high level language program into a complete machine code program (called the object code). This object code may then be run directly.

Compiled languages generally run a lot faster than interpreted ones, but suffer the inconvenience that you need to compile a program before it can be run - though you only need to compile any given version once, providing that it is bug free.

ACORNSOFT 'C'

Acornsoft's 'C' is no different in this respect, and comes with a set of BBC specific library routines on disc. Both 5.25 inch (DFS format) and 3.5 inch (ADFS format) discs are supplied in the pack, which also contains a very competent 250 page manual, a keystrip (for the editor) and a shortlist of known bugs. What is

important to note about Acornsoft's product, however, is that it will *not* work across the full range of Acorn micros. If you want to run it on a straight model B, you will also need a 6502 second processor. Master and Compact users, with all that extra RAM, have no need of a second processor though. The Beebugsoft version, by contrast, will work on all machine configurations, including a bare B.

Whichever version of 'C' you are using, there are four major stages to the process of programming. First you must write your program using a text editor, and save the result to disc as an ASCII file. You then *compile* this text source into *intermediate code*. You then *link* this code into what Acorn call *object code*. Finally this object code may now be executed.

In the case of both Acornsoft's 'C' and Beebug's, this final code is not full stand-alone object code. To run it, you must have 'C' installed in the machine. It is to be hoped that someone will produce a 'converter' program which will convert the so-called 'object code' to fully stand-alone code. But even if this does not happen, there is still the major advantage that versions of 'C' for other micros - including the Archimedes - will be able to generate stand-alone code from source code created in 'C' on the BBC micro or Master.

SETTING UP

Let us look in a little more detail at the various processes involved in using Acornsoft's 'C'. Before you can do any programming at all, there is a setting up and configuring process to be performed. How this is accomplished depends on which machine configuration that you are using. The Master and Compact version of the software, for example, requires you to run a special file which loads up all four banks of sideways RAM. You must then execute a hard Break to initialise the code, and can then type:

```
*C
```

to invoke the language. To signify that you are within 'C', you will then get the following prompt:

```
C>
```

and can proceed to use any one of a number of commands.

Once you have reached this stage, the next job is to create a program source code using the editor - so type:

```
EDIT
```

Second processor users must use *EDIT to load in the editor, so overwriting the compiler, because the second processor is a bit tight on RAM! And Master users are advised to *UNPLUG the Master's resident editor because of a command name conflict.

THE EDITOR

This looks and behaves almost identically to the Master's Editor, except that for some reason I could not get the Change Mode key to work, and it uses a virtually identical keystrip. It is a nice piece of software, and is very easy to use. Once you have typed in your program it should be saved to disc in the C directory. You then use Shift-F4 to take you directly back to the C> prompt. Second processor users need to perform a *TC to reinstall the compiler. In the Beebug 'C' package, by contrast, there is no built-in editor; and users are recommended to use View, Wordwise, the Master Editor or some other word processor for the purpose.

THE COMPILER

To compile your program, just type:

```
COMPILE filename
```

at the C> prompt. Acornsoft's compiler operates in three passes, performing the following tasks:

- 1st pass - Preprocessing
- 2nd pass - Syntax check & code generation
- 3rd pass - Post processing

If this all proceeds without error, the newly created intermediate code is automatically saved away in the L directory, and you may proceed to link your program.

THE LINKER

It is at this stage that the final code is produced, and any required library files are automatically appended to the program. User-created library functions may also be appended at this point. At the end of this process you should have on disc in directory O a file which may be directly executed. To run it, simply type the filename (without the directory) at the prompt. If it doesn't run correctly, you must, as with all

compiled systems, go all the way back to the editor to correct it; and then repeat the succeeding stages until you get it right.

PERFORMANCE TESTS

To check out the performance of Acornsoft's 'C' we have run it against two PCW benchmarks (converted to 'C' from Basic), and given comparative figures for the rival Beebugsoft package. To give you an idea of what a 'C' program looks like, we have included the listing for INTMATH. As you can see from the results, the compile time for the Beebug version is considerably quicker than Acornsoft's, though on INTMATH, Acornsoft's run time is a good bit better. These figures seem to be reasonably representative of the two packages, over other code on which we have tested them.

PCW BENCHMARK 1 (INTMATH)

```

/* program intmath */ |comment
#include <h.stdio> |library file
main() |program start
{
int i,x,y; |integer variables
i = 0;
x = 0;
y = 9;
puts("start"); |basic print
while(i < 1000) |main loop
{
x = x + (y * y - y) / y;
i++; |increment i
}
printf("Finish %d\n",x); |formatted
} |print

```

You will also notice that the Beebugsoft code is much shorter in length than Acornsoft's. This is because the linker in the Acornsoft package includes all library routines, even though only a small subset may be required. Acornsoft do supply some subsets of their library to reduce this overhead; and of course in those situations where you need large parts of the library, the code lengths of the Acornsoft and the Beebug source code will be more comparable.

It should be said that neither the Acornsoft nor the Beebug compilers produce particularly fast code. Generally speaking, while the object code from both compilers executes faster than BBC

BENCHMARKS

	Compile Time	Link Time	Run Time	Size (bytes)
INTMATH:				
Acornsoft	40.4	36.8	1.3	8876
Beebugsoft	8.6	9.0	2.5	230
REALMATH:				
Acornsoft	42.1	41.5	4.9	8885
Beebugsoft	8.7	8.8	4.9	238

The benchtests were performed on a standard Master 128 fitted with DFS v2.29 and 5.25" floppy disc. All times are in seconds.

Basic, their speed is much closer to Basic than it is to normal (hand produced) 6502 machine code. Just for the sake of comparison, we give times for INTMATH and REALMATH for the Borland Turbo C running on an Amstrad PC, which note, uses a 16 bit processor.

BORLAND C TIMES

	Run Time	Size (Bytes)
INTMATH	0.04	5484
REALMATH	3.70	19288

CONCLUSION

Acornsoft have produced a creditable implementation of 'C'. It follows the full Kernighan and Ritchie standard and is easy to operate, if a little slow to compile. The documentation is first rate, though be warned that the guide supplied is a guide to the product and not to the language. To use this package, you will need an additional text on 'C'.

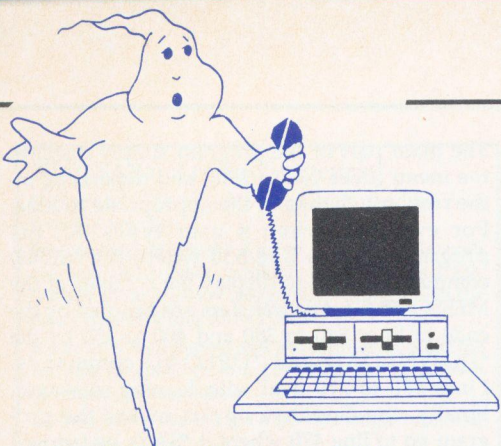
The recommended book for both tutorial and reference is of course:

The C Programming Language
by B. Kernighan and D. Ritchie
Prentice Hall, £22.75 (members)

B

PRICES (inc VAT)

Code Product	Members	Non-Members
1046C Acornsoft C	£85.22	£89.70
0075B Beebugsoft C	£44.25	£59.00
1289B K & R Book	£22.75	£23.95



The Ghost Host

If you have a modem and use your computer to communicate with Miconet and the many bulletin boards that are accessible, you might well have wondered how a bulletin board is set up. David Pilling, author of BEEBUG's Command ROM, has written a program that you can use to do just that.

Anyone who uses their Beeb for communications, will be familiar with big databases like Prestel and Telecom Gold. In addition, there are many hundreds of private bulletin boards (see BEEBUG Vol.6 Nos.4 & 5 for details). Usually, these consist of a micro-computer running a bulletin board program (for instance the Fido package for the IBM-PC).

THE GHOST HOST DESCRIBED

The Ghost Host, is a host or bulletin board system for the BBC micro. It can be used as an answering machine, as your own private bulletin board, or simply to allow you to transfer files between machines without repeatedly having to run across the room. To use the Ghost Host program, you must have a Magic (or compatible) modem and have the BEEBUG Command ROM installed in your machine. This ROM has over 50 star commands and five OSWORD calls which provide extensive support for writing customised communications software like Ghost. In fact Ghost only just begins to exploit all the facilities provided by Command.

A bulletin board, should provide at least downloading and uploading of programs, and sending and receiving messages. These are exactly the options provided by the Ghost Host.

SETTING UP YOUR OWN BULLETIN BOARD

If you fancy running your own bulletin board, begin by typing in the accompanying listing for the Ghost Host, and save the program to disc. Then, connect your Magic modem to the phone line and to your Beeb, setting the switch on the front to '1200 ans'. Next, create an ASCII file called MESSOUT containing any information or messages that you want people logging on to your system to read (you can use most word processors for this). In addition, you may want to put other files on your discs for downloading. It would be sensible to lock all the files (on all drives) to stop them from being accidentally (or otherwise) erased.

Running the program will set things off. What happens next, is that Ghost waits until a call is received. When this happens, Ghost will answer the phone and, assuming that there is a computer at the other end, begin talking to it. You will see the transmitted text from Ghost and the responses from the other end appear on your Beeb's screen.

Looking at things from another point of view, what happens if you call up the Ghost Host? After dialling the correct number, and going on line you will be presented with a welcome message and asked to enter the password for the particular host (pre-defined in line 1080). You have three chances at this; if you fail to get it right, you are automatically logged off. After a successful log-on, you will see the Ghost Host menu:

- D)ownload R)ead messages
- U)pload L)eave message
- G)oodbye

Each option is selected by typing the appropriate letter followed by Return. Of these, Uploading and Downloading are fairly obvious. Files are transferred using Xmodem, and you must remember to set off your uploader/downloader after telling Ghost what you are going to do. If you opt to read the messages file, you will have the contents of MESSOUT typed out on your screen. The 'Leave a message' option, lets you type in lines of text which are appended to the end of the file MESSIN. Pressing Return twice, terminates this

process. The owner of the host, can *TYPE out this file later to see your message. Finally 'Goodbye', terminates your session on Ghost, causing it to log you off and drop the line.

CUSTOMISING YOUR BULLETIN BOARD

There are a few obvious modifications that can be made. Firstly, it would be nice to have a way of stopping and starting the output of the message file. This can be achieved by putting *XON in line 545. The user can then halt things by typing Ctrl-S, and restart output with Ctrl-Q, inserting his own text between the two.

One traditional way of swapping files between different computers, BBC and PC etc., is to connect them together via their serial ports and run a communications program on each machine. However, this can lead to a lot of running around. Instead, you can run Ghost on the Beeb and type all the necessary upload/download commands from the other machine. Such a server-like function may be familiar to users of Kermit. If you want to use Ghost in this way, it is necessary to remove *ANSWER from line 140 and replace it with *CONNECT. In addition, you may like to replace *V23A in line 1030 with a couple of *RXRATE and *TXRATE commands to set the fastest baud rate that you can get away with.

UNDERSTANDING THE PROGRAM

I hope that the way in which Ghost works is obvious from the listing. The key piece is probably its only obscure bit. This is the use of the Command ROM OSWORD call to input strings from the remote computer; which is packaged up in the function FNin (line 1160). This function allows you to get strings from the remote computer in exactly the same way that INPUT is used to get strings from the keyboard in a conventional program. Similarly, instead of PRINT to put messages on the screen, there is the *SAY command to print messages on the distant screen. Using these two simple tools, you can easily construct what appears to be a sophisticated on-line system. For example, look at the way the password checking is implemented in lines 150 to 230. Also notice how the bar M bar J sequence is used in the *SAY strings to send Carriage-Return/Line-Feed sequences at the end of each line of output.

The main part of the program simply displays the menu (lines 260 to 290), and depending on the response, jumps to the appropriate routine. For example, when a user types 'R', the program goes to line 530 where the *SEND command is used to display the contents of the MESSOUT file. File transfers are handled by the code in lines 340 to 380 and 400 to 450 by the *DOWNLOAD and *UPLOAD commands. Lines 580 to 710 deal with leaving messages. Finally, the Goodbye option makes the program go to line 470 where it *SAYS its farewell message and *DISCONNECTs. The % in the *SAY in line 500 gives a time delay for the text to be sent before the modem drops the line.

The worst thing that can happen in a program like this, is that it should hold the phone line for some reason long after the remote computer has gone away. I have made great efforts to prevent this situation occurring. The program's natural flow of control repeatedly goes through line 250 where FNOnline (again using a Command OSWORD call) checks if the system is still on line, and if this is not the case, restarts the program. The section of code between lines 730 and 800, handles errors. Again, this code is designed to prevent the program holding the phone line. In line 750, a check is made to see if the string input function has timed out, and if it has the program is restarted.

PROCSETUP is the initialisation routine for Ghost. Line 1040, *DISCONNECT, ensures that whenever the program is started up, the modem drops the phone line. You can replace *V23A in line 1030 with one of *V230, V210 or V21A, as long as you adjust the baud rate of your modem accordingly. V23A gives a transmit rate of 1200 baud and a receive rate of 75 baud, which is just the same as Prestel.

The password for the host is set to 'PASSWORD' in line 1080. Obviously, this can be changed to whatever is desired. Similarly, the variable pass% in this line controls the number of attempts allowed at the password when logging on.

The program presented here is only a skeleton; with a little imagination, you should be able to add lots more features. Here are some ideas

GLOSSARY of computing terms

Download - To transfer a file from the host to a remote computer.

File Transfer Protocol - A set of rules agreed by both the host and remote computer for swapping a file. Usually these rules are designed to spot errors in the data transferred and correct them.

Kermit - A well known and increasingly popular file transfer protocol. Also a very nice public domain communications program.

Upload - To transfer a file from the remote computer to the host.

Xmodem - The most widely used file transfer protocol. Not as sophisticated as Kermit.

Xon/Xoff - A method of regulating the flow of data between two computers so that one does not send information quicker than the other can receive it.

that you might wish to explore:

Call logging. By using the real time clock in the Master it should be easy to write to a file the time the host was accessed and how for long.

Change the logon system so that anyone can become a user by leaving their name. You can then use call logging to see who uses the system longest, and what they download.

Extend the message system, so that

individual users have separate mail files, and can send messages to each other.

Provide different special interest areas, each with its own message files and downloadable programs.

Add an option to the main menu to display a catalogue of the programs available for downloading.

Add teletext colour to the messages sent by the host.

Use the real time clock to give messages like the current time, and how long someone has been connected.

Continuing along these lines, you could easily write a full scale bulletin board. Finally, Command also provides many of the facilities needed to run your own Viewdata host.

If you set-up a regular bulletin board, then let us have the details and we will include your board in our list (see supplement).

If you are using this program with Basic I you will need to substitute a suitable replacement OSCLI procedure in lines 370 and 440.

```

10 REM Program GHOST
20 REM Version B1.0
30 REM Author David Pilling
40 REM BEEBUG October 1987
50 REM copyright BEEBUG
60 :
100 PROCSETUP
110 ON ERROR GOTO 730
120 MODE7
130 PRINT"waiting..."
140 *ANSWER
150 *SAY"%1|M|J|J"
160 *SAY"Welcome to the Ghost
Host"
170 *SAY"%M|J"
180 *SAY"Enter password:"
190 IF FNin=password$ GOTO
230
200 *SAY"Wrong
password|M|J|J"
210 pass%=pass%-1:IF pass%
GOTO 150
220 GOTO 470
230 *SAY"Password accepted"
240 :
250 IF FNonline ELSE RUN
260 *SAY"%M|J"
270 *SAY"D)ownload U)pload
G)oodbye "
280 *SAY"R)ead messages
L)ave "
290 *SAY"message|M|J"
300 *FX21,1
310 A$=FNin:IF A$="" GOTO 250
320 ON INSTR("DUGRL",A$)+1
GOTO 250,340,400,470,530,580
330 :
340 REM Download
350 *SAY"File to Download:"
360 A$=FNin:IF A$="" GOTO 380
370 OSCLI("UPLOAD "+A$)
380 GOTO 250
390 :
400 REM Upload
410 *SAY"File to Upload:"
420 A$=FNin:IF A$="" GOTO 450
430 *FX138,0,78
440 OSCLI("DOWNLOAD "+A$)
450 GOTO 250
460 :
470 REM Goodbye
480 *SAY"Goodbye from the
Ghost Host"
490 *SAY"%M|J|Jthanks for
calling."
500 *SAY"%M|J%5"
510 RUN
520 :
530 REM Read messages
540 *SAY"Today's
messages:|M|J|J"
550 *SEND MESSOUT
560 GOTO 250
570 :
580 REM Leave messages
590 *SAY"Enter your
message|M|J"
600 *SAY"Terminate with two
RETURNS"
610 *SAY"%M|J"
620 C%=OPENUP("MESSIN")
630 IF C%=0
C%=OPENOUT("MESSIN")
640 PTR#C%=EXT#C%
650 A$=FNin
660 IF A$="" GOTO 710
670 FOR I%=1 TO LEN A$
680 BPUT#C%,ASC(MID$
(A$,I%,1)):NEXT
690 BPUT#C%,13:BPUT#C%,10
700 GOTO 650
710 CLOSE#C%:GOTO 250
720 :
730 REM Errors
740 IF ERR=106 THEN *SAY"Not
Found|M|J"
750 IF ERR=103 OR ERR=100 RUN
760 IF ERR=102 THEN
*SAY"Download Aborted|M|J"
770 CLOSE#0:IF ERR=17 GOTO800
780 IF ERR<100 OR ERR>110
OSCLI("SAY Oops! Error numb
er "+STR$ERR+"|M|J")

```

Continued on page 32

Cataloguing Discs with Filer

Many disc users find that some form of catalogue is essential if they are to keep track of a multitude of files stored on different discs. Jan Stuurman describes a program to automate the process of disc cataloguing, and uses the Filer Database system to transform this into a slick display.

FILERD is another program in the BEEBUG Filer database series (see BEEBUG Vol.5 No.9). The purpose of this utility is to compile a catalogue of all your disc files so that any one may be easily located. It does this quite automatically, reading all the necessary information from disc. The resulting data file, in Filer format, can then be sorted, displayed and printed out using all the facilities of Filer. There is even a remarks line to describe each file, and the catalogue can be updated as needed. The program could also be adapted to create data files for most other database systems, or even Wordwise text files.

The records created by FILERD contain 8 fields: disc code, filename, directory, load address, execute address, length of file, start sector on disc and remarks. The disc code consists of a (unique) three character/digit disc identification code of your choice, plus either a 0 or a 2 (1 or 3) specifying the side of the disc. The other fields, with the exception of the remarks, contain the same information as obtained by the *INFO command.

Inclusion of the remarks field is optional (see program notes); it provides for additional information on each file. For Basic programs which have a first line starting with a REM statement (no space between line number and REM), the remarks field will contain the first 50 characters of the text message following the REM (for more information see the article on extended disc catalogues in BEEBUG Vol.4

No.2). FILERD provides some remarks field information for non-Basic files and for files with bad filenames (see later).

Before FILERD can be used for the first time, an empty data file has to be created with BEEBUG Filer. The size will depend on the number of files to be catalogued. When specifying the file format, the following eight fields must be included with the field widths shown. The names of these fields may be changed and additional fields may be added.

Field Name	Field Width
DISCCODE	5
FILENAME	7
DIRECTORY	1
LOAD ADDRESS	4
EXEC ADDRESS	4
FILE LENGTH	5
START SECTOR	3
REMARKS	50

Again, the REMARKS field may be omitted (see program notes), but otherwise its width should be equal to the value of rem in line 1080. If you intend to sort the resulting file with FILERS, the maximum number of records should be set to 999 or less (see BEEBUG Vol.5 No.7).

```

BEEBUG DISC FILER
Options:
  1 - Add new disc catalogue record
  2 - Update disc record
  3 - Delete disc record
  4 - Quit program
Choose option (1-4) : 1
Memory left for 247 catalogue entries
Insert disc to be catalogued in drive
Enter disc code :
or RETURN to finish :
Insert FILER data
file disc in drive : 0
Enter FILER filename: DiscCat
    
```

After the file has been created and closed, FILERD can be run, but make sure that you have saved a copy of the program first. PAGE should be set to &1400, as with the other Filer programs, unless running on a Master, Compact or with OPUS DDOS, where PAGE is at &E00 anyway.

USING FILERD

The main menu offers three operations that may be performed on the data file:

1. **ADD** - Append disc catalogue entries to the existing file.
2. **UPDATE** - Delete disc catalogue entries with given disc code from the file and append current catalogue entries.
3. **DELETE** - Delete disc catalogue entries with given disc code.

The UPDATE operation is simply a combination of the other two, DELETE and ADD.

Each operation may be performed on many discs (disc codes) before the data file is actually updated. The number of discs that can be handled depends on the type of operation, the amount of memory available and the number of files on each disc.

For ADD and UPDATE operations the disc to be catalogued should be inserted in the disc drive as prompted. For all operations you will need to key in the disc code and drive (side) number. This process is repeated until either the user presses Return when asked to enter the disc code, or when the maximum number of disc codes (maxd=20 at line 1070) has been

recorded. The user is then asked to insert the disc with the Filer data file in the drive. The drive number and filename must then be entered.

After the data file has been updated, control is passed back to the main menu until option 4 'Quit program' is chosen. Escape may be used to abort any operation, and pressing the space bar will then return you to the main menu.

USING YOUR CATALOGUE

Once you have created your catalogue file, or whenever you have updated it, you may use all the facilities of Filer as usual. This means that you can sort the entries, search for and display entries, and by using a suitable print format, print out your catalogue (see illustration). You can of course, also use Filer to amend or delete any of the entries in the catalogue, without re-running FILERD.

PROGRAM NOTES

Inclusion of the remarks field depends on the value of the variable rem as set in line 1080. If rem=0 no remarks field will be stored, otherwise the value of rem determines the width of this field.

The data statement in line 1100 gives the length of each field. Note that variable names, e.g. rem, may be put in a data statement. READ L% assigns their value to the variable L%.

3DLETRR	9	7.0B.D	1900	0021	0070	3D LETTERS FOR PRINTING: 0010/021
3DMAZE	9	5.0D.J	0000	0023	1901	3 DIMENSIONAL MAZE PRAC.T. COMP. AIR/03
3DMH2E	9	4.0D.B	0000	0073	10CF	0010/03 2D MAZE
3DFEIT	E	7.0B.D	0000	0073	00AF	3-D TEXT Demo program 004/3-4
EDITOR	9	1.0E.C	1900	0023	00FF	TELETYPE EDITOR PCW NOV 84
A	9	5.0D.C	FFFF	FFFF	0101	*RFD000 text file
ACCENT	9	2.0E.A	1900	0023	0170	ACCENT LETTERS a,e,i,o,u
ALC-IMP	D	2.0B.I	0000	0023	0200	DEF PROGRAM ROUTINE
ALFRAM	9	2.0F.C	0000	0023	0270	ABS.LANG. OUTLINE 4 TEST ROUTINE
ALIN	9	5.0E.B	0000	0023	0300	0010/02
ALIN	9	E.0D.I	0000	0020	0400	NO REM statement
ALIN2	9	E.0B.W	0000	0023	0100	NO REM statement
ALFRM	9	0B1.2A.C	0000	0023	0E00	UNIDENTIFIED FLYING LETTERS SINGLE KEY ENTRY
ALLCOD	U	1.0B.A	1900	0023	0F70	NO REM statement
ALTUDR	9	0.0B.W	1900	0023	0D4D	ASSEMBLY LANGUAGE TUTOR
ANAB	9	0B1.2A.N	0000	0023	2D00	ANABAME LEVEL 1-12
ANAB	9	1.0B.B	1300	0023	01C0	ANABAS PART 3 OF ANALYSER
ANALHC	9	1.0B.D	1300	0023	0536	ANALHC PART 2 OF ANALYSER 002/3
ANALYZE	9	1.0B.C	0000	0023	0300	ANALYZE BASIC PROGRAM PERFORMANCE ANALYSER 003/5
ARCAB	9	4.0C.C	0000	0023	0007	...APCADIAN... VJ
ARCDEF	9	4.0C.B	1900	0023	4E00	Machine code program
ASKEYT	K	1.0D.D	0000	0000	00FF	*KEY definitions
ARBEV2	H	1.0B.B	0000	0000	00FF	KEY definitions
ASMAZE	9	0.0E.F	1900	0023	1520	PROGRAM ASMAZE
ASTARD1	9	9.0D.B	0000	0021	0051	ASTARD 002/2-4
ASTARDC	9	9.0D.C	0000	0023	1114	ASTARDC 002/9-5
ATTACK	9	5.0E.E	0000	0023	7AA7	PCN 4/25/85
ATTACKR	D	5.0E.H	1900	0023	0752	*RFD000 text file
B	9	0.0F.F	FFFF	FFFF	0101	BACH'S CHAVIRA NO.147 REEBUD/9-10
BACH	9	7.0F.I	1900	0023	0500	REEBUD VOL. 3 NO 5 BACH2
BACH2	9	7.0F.H	1900	0021	15C0	REEBUD VOL. 3 NO 5 BACH2
BACH3	9	1.0B.D	1900	0023	005F	BACH'S CHAVIRA NO.147 REEBUD/9-10
BALLOON	9	7.0C.N	0000	0023	064F	FLICKER FREE GRAPHICS DEMO2 BALLOON
BARRE	D	5.0E.C	0000	0023	0000	BARREL DUMP BY CANCEDE
BARRETT	9	1.0B.D	0000	0023	1E03	BASIC Extensions "Do it yourself BASIC" 005/1
BASICID	9	1.0B.A	1900	0023	00DF	NO REM statement
BASKEV1	9	1.0B.C	0000	0000	00FF	KEY definitions
BASMAZE	9	0.0E.F	1900	0023	0040	PROGRAM BASMAZE
BBDD	9	0.0B.D	1900	0023	0000	KEY definitions
BFIDTR	9	1.0B.I	7000	0023	0070	UTILITY EDITOR FOR PROGRAM DEVELOPMENT DEF REM 2-9
BFIDTR	9	1.0B.I	1900	0023	0070	UTILITY EDITOR FOR PROGRAM DEVELOPMENT DEF REM 2-9
BIDRAW1	9	0.0B.B	1900	0023	1722	PROGRAM BIDRAW2
BIDRAW2	9	0.0B.B	1900	0023	13E1	PROGRAM BIDRAW2
BIDRMND	9	7.0B.H	1900	0023	0746	PRINT BIN NUMPBR IN HDGS 0,1,2,4 OR 5
BIDTFT	9	2.0B.W	0000	0000	0000	DEF PROGRAM routine
BILL	9	0D.1.0B.Z	1900	0023	0007	REEBUD DEMO PROGRAM
BIR02.1	9	0.0B.E	1900	0023	0126	BIRNAUM LIGHTING 2.1 21/03/84
BIR02.2	9	0.0B.F	1900	0023	000F	BIRNAUM LIGHTING 2.2 21/03/84
BIR02.3	9	0.0B.G	1900	0023	0157	BIRNAUM LIGHTING 2.1 20/03/86

Sample catalogue printed with Filer

Bebug October 1987

On some commercial discs, the names of files contain non-printable ASCII values (<32 or 127). FILERD replaces these codes by the "#" sign. In this case and when a filename contains a DFS wildcard character ("#" or "*"), a BAD FILENAME message is put in the remarks field. All ASCII codes greater than 127, in both filenames and REM statements, are ANDed with &7F to clear the MSB which would upset most printers if left unchanged.

```

10 REM Program FILERD
20 REM Version B1.2
30 REM Author Jan Stuurman
40 REM BEEBUG October 1987
50 REM Program subject to copyright
60 :
100 MODE7:HIMEM=PAGE+&1700
110 ON ERROR GOTO 250
120 PROCtitle
130 PROCinit
140 ON ERROR GOTO 250
150 REPEAT
160 B%=HIMEM:cat=0:disc=0:del$=""
170 opt=FNmenu:IF opt=4 GOTO220
180 IF opt<>3 PROCreadcat ELSE PROCdel
ete
190 PROCfiledisc
200 IF opt<>1 PROCremove
210 IF opt<>3 PROCwritecat
220 UNTIL opt=4
230 MODE7:END
240 :
250 ONERROROFF:CLOSE#0
260 IF ERR=17 PROCmessage("DISC FILER
ABORTED"):GOTO140
270 REPORT:PRINT" at line ";ERL
280 END
290 :
1000 DEFPROCtitle:FOR I=0 TO1
1010 PRINTTAB(8,I)CHR$141CHR$129CHR$157
CHR$131"BEEBUG DISC FILER "CHR$156
1020 NEXT:ENDPROC
1030 :
1040 DEFPROCinit
1050 FDR=256:buf%=&900
1060 maxf=12:DIMrec$(maxf)
1070 maxd=20:del$=STRING$(5*maxd," ")
1080 rem=50:rem$=STRING$(rem," ")
1090 len%=30+rem:fn$=STRING$(8," ")
1100 DATA 5,7,1,4,4,5,3,rem
1110 ENDPROC
1120 :
1130 DEFFNmenu:LOCALI,N,opt,opt$
1140 VDU28,0,24,39,3,12
1150 PRINTCHR$134"Options:""

```

```

1160 RESTORE1240:READN
1170 FORI=1TON:READopt$
1180 PRINTTAB(4)CHR$134;I;CHR$133"- "op
t$
1190 NEXT
1200 PRINT'CHR$134"Choose option (1-";N
;") : "CHR$131;
1210 REPEAT opt=GET-48:UNTIL opt>0ANDop
t<=N:PRINT;opt
1220 =opt
1230 :
1240 DATA 4
1250 DATA Add new disc catalogue record
,Update disc record
1260 DATA Delete disc record,Quit progr
am
1270 :
1280 DEFPROCreadcat:LOCALN%
1290 REPEAT VDU28,0,24,39,12,12
1300 room%=(&7C00-B%)/len%
1310 PRINTCHR$133"Memory left for"CHR$1
34;room%;CHR$133"catalogue entries"
1320 PRINT'CHR$133"Insert disc to be ca
talogued in drive"
1330 PROCdiscode:IF code$="" GOTO1380
1340 N%=FNnooffiles(DR%,0,2)
1350 IF N%>room% PROCmessage("NOT ENOUGH
MEMORY LEFT"):GOTO1380
1360 VDU28,0,22,39,19,12
1370 FORF%=1TON%:PROCfile(F%):NEXT
1380 UNTIL code$=""ORDisc=maxd
1390 ENDPROC
1400 :
1410 DEFPROCdelete
1420 REPEAT PROCdiscode
1430 UNTIL code$=""ORDisc=maxd
1440 ENDPROC
1450 :
1460 DEFPROCfiledisc:LOCALDR%
1470 REPEAT VDU28,0,24,39,19,12
1480 PRINTCHR$134"Insert FILER data'"CH
R$134"file disc in drive : "CHR$131;
1490 REPEAT DR%=GET-48:UNTIL DR%>=0ANDD
R%<=3:PRINT;DR%
1500 PRINT'CHR$134"Enter FILER filename
:"CHR$131;:INPUT""file$
1510 file$=":"+STR$DR%+." "+file$:PROCOp
en(file$):UNTIL ch%<0
1520 CLOSE#ch%:ENDPROC
1530 :
1540 DEFPROCremove:LOCALI%,J%
1550 IF LENdel$=0 ENDPROC ELSE PROCopen
(file$)
1560 IF rec=1 GOTO1610
1570 J%=1:FORI%=1TOrec-1:PROCread(I%)
1580 IF INSTR(del$,rec$(I))=0 J%=J%+1:I
F I%>J%-1 PROCwrite(J%-1)
1590 NEXT:rec=J%
1600 PTR#ch%=0:PRINT#ch%,rec

```



```

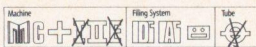
1610 CLOSE#ch%:ENDPROC
1620 :
1630 DEFPROCwritecat:LOCALI%,J%,K%,L%
1640 IF cat=0 ENDPROC ELSE PROCopen(fil
e$)
1650 IF rec+cat>recn PROCmessage("NOT E
NOUGH FILE SPACE LEFT"):GOTO1710
1660 B%=HIMEM:FORI%=1TOcat:RESTORE1100
1670 J%=1:FORK%=1TO8+(rem=0):READL%
1680 rec$(K%)=MID$(B%,J%,L%):J%=J%+L%:
NEXT
1690 PROCwrite(rec):rec=rec+1:B%=B%+len
%:NEXT
1700 PTR#ch%=0:PRINT#ch%,rec
1710 CLOSE#ch%:ENDPROC
1720 :
1730 DEFFNnooffiles(DR%,SC%,NS%)
1740 ?&80=DR%:!&81=buf%:!&85=&5303
1750 ?&88=SC%:?&89=&20+NS%
1760 X%=&80:Y%=0:A%=&7F:CALL&FFF1
1770 IF ?&8A<>0 PROCmessage("READ/WRITE
ERROR "&+STR$~(?&8A)):END
1780 =(buf%?261)DIV8
1790 :
1800 DEFPROCdiscode
1810 REPEAT:VDU28,0,24,39,16,12
1820 PRINTCHR$134"Enter disc code :
"CHR$131
1830 PRINTCHR$134"or"CHR$133"RETURN"CHR
$134"to finish "CHR$11;
1840 INPUT"code$:IF LENcode$>3 PROCmes
sage("MAXIMUM 3 CHARACTERS")
1850 UNTIL LENcode$<=3:IF code$="" ENDP
ROC
1860 PRINTCHR$134"Side/drive number :
"CHR$131;
1870 REPEAT DR%=GET-48:UNTIL DR%>=0ANDD
R%<=3:PRINT;DR%
1880 code$=RIGHT$(" "+code$,3)+"."+STR
$DR%
1890 IF opt<>1 del$=del$+code$:disc=dis
c+1
1900 IF disc=maxd PROCmessage("MAXIMUM
NUMBER OF DISC CODES RECORDED")
1910 ENDPROC
1920 :
1930 DEFPROCfile(F%):LOCALC%,P%,Q%
1940 P%=buf%+F%*8:Q%=P%+256
1950 fn$="":FORC%=0TO7
1960 fn$=fn$+FNalf(P%?C%):NEXT
1970 lo$=FNhex(Q%?0+&100*Q%?1+&10000*((
Q%?6AND&0C)DIV 4),4)
1980 ex$=FNhex(Q%?2+&100*Q%?3+&10000*((
Q%?6AND&C0)DIV64),4)
1990 le$=FNhex(Q%?4+&100*Q%?5+&10000*((
Q%?6AND&30)DIV16),5)
2000 ss$=FNhex(Q%?7+&100*(Q%?6AND&03),3
)
2010 IF rem>0 re$=FNrem ELSE re$=""

```

```

2020 PRINTCHR$131code$CHR$130fn$TAB(18)
CHR$133lo$" "ex$" "le$" "ss$
2030 PRINTCHR$130LEFT$(re$,39)
2040 $B%=code$+fn$+lo$+ex$+le$+ss$+re$
2050 B%=B%+len%:cat=cat+1
2060 ENDPROC
2070 :
2080 DEFFNalf(A%):A%=A%AND&7F
2090 IF A%<32ORA%=127 ="#" ELSE =CHR$A%
2100 :
2110 DEFFNhex(A%,L%)
2120 =RIGHT$("0000"+STR$~A%,L%)
2130 :
2140 DEFFNrem:LOCALF%,I%,CF%
2150 IF fn$="!BOOT $" rem$="BOOT-up fi
le":GOTO2290
2160 IF ex$="0000"ORex$="FFFF" rem$="Te
xt,SPOOL or data file":GOTO2290
2170 IF ex$<"8000" rem$="Machine code f
ile":GOTO2290
2180 IF INSTR(fn$,"#")+INSTR(fn$,"*")<>
0 rem$="> BAD FILENAME <":GOTO2290
2190 F$="":+STR$DR%+"."+RIGHT$(fn$,1)+"
."+LEFT$(fn$,7)
2200 rem$="BASIC program file"
2210 CF%=OPENUP(F%):IF CF%=0 GOTO2280
2220 PTR#CF%=4:F%=BGET#CF%:IF F%<>244 G
OTO2280
2230 REPEAT F%=BGET#CF%:UNTIL F%<>32
2240 PTR#CF%=PTR#CF%-1:rem$=""
2250 REPEAT F%=BGET#CF%
2260 IF F%<>13 rem$=rem$+FNalf(F%)
2270 UNTIL F%=13ORLENrem$=rem
2280 CLOSE#CF%
2290 =LEFT$(rem$+STRING$(rem, " "),rem)
2300 :
2310 DEFPROCopen(p$)
2320 IF MID$(p$,5)="" PROCmessage("NO F
ILE GIVEN"):ch%=0:ENDPROC
2330 ch%=OPENUP(p$):IF ch%=0 PROCmessag
e("NO SUCH FILE"):ENDPROC
2340 PTR#ch%=0:INPUT#ch%,rec,recn,recs,
f
2350 ENDPROC
2360 :
2370 DEFPROCread(n):LOCALI
2380 PTR#ch%=FDR+recs*(n-1)
2390 FORI=1TOf:INPUT#ch%,rec$(I):NEXT
2400 ENDPROC
2410 :
2420 DEFPROCwrite(n):LOCALI
2430 PTR#ch%=FDR+recs*(n-1)
2440 FORI=1TOf:PRINT#ch%,rec$(I):NEXT
2450 ENDPROC
2460 :
2470 DEFPROCmessage(m$):LOCALI
2480 PRINT"TAB(19-LENm$/2)CHR$130;m$
2490 VDU7:I=INKEY400
2500 ENDPROC

```

SUPER FUNCTION KEYS

If you never have enough function keys, or find it difficult to remember what each function key does, John Cole's utility offers a novel solution. Now every key on your keyboard can be a function key.

The red function keys are an invaluable feature of the BBC Micro, especially when you realise that whole sets of definitions can be loaded at once from a Basic program or !BOOT file. There are drawbacks though. Most of us find that there are simply too few function keys for everything we want to do. Also, a key legend like 'f7' gives no clue as to the string it has been programmed to produce. The two problems conspire together to generate numerous strips of cardboard, inscribed with mnemonics for alternative definitions. Somehow the right strip is never to hand when needed, and the facility doesn't get used as much as it might.

Simple as it is, the accompanying program helps considerably with both problems. It enables any of the ASCII codes to be programmed with a string, using the normal *KEY conventions. The string is invoked by preceding the corresponding keyboard character with Tab. In other words, the Tab key becomes a 'super' function key which delivers a different string according to the next key to be pressed, (the normal Tab action is not lost, but requires the key to be pressed twice). The result is that there is no longer any shortage of keys to be programmed, and the characters can be chosen to provide mnemonics for their own definitions. When programming, for instance, you might wish to use upper case letters for Basic keywords, lower case letters for operating and filing system commands, and Ctrl codes for mode 7 teletext codes:

Tab-D = DELETE
Tab-d = *delete
Tab-Ctrl-D = !!M, the 'double height' character

The facility is no less useful for writing common words or phrases when word processing. View and Wordwise usually preempt the normal function key action with their own commands, leaving the awkward Shift-Ctrl-Fkey combinations to invoke the user's own definitions.

USING THE PROGRAM

The program should be typed in and saved to disc or tape before proceeding. The Basic program when run assembles a short piece of machine code, together with a look-up table of your definitions, in two pages beginning at the value of M% set at line 100. The two pages are *SAVED by the program as the file "Keys". To load your definitions it is not necessary to run the Basic program each time: entering *Keys loads the file from disc and runs the code to reset the operating system read character vector RDCHV. The super function key facility will then be operative until Break is pressed or RDCHV otherwise reset. As it stands the program uses pages 9 and 10, normally used for RS423, cassette, sound, and speech buffers. If you have a BBC Master and do not use Econet you may prefer to change M% to &B00 to allow these other facilities to be used at the same time. Similarly, cassette users should make space available elsewhere and change M% accordingly.

The listing here doesn't contain a long list of definitions; its essence is that you can easily add your own. Just two are included to show how it is done, each as a single string in a DATA statement. To be recognised as a definition the string must contain the "=" character. To the left of the equality is the character code being programmed, in the range 0 to 175. To the right is the string to be generated. In both cases the normal operating system conventions can be used, such as !M for Return and !! in characters with ASCII codes above 127. As usual with strings in DATA statements, the definition needs to be enclosed

in quotes if there is a quote (") or comma within it, and quotes are denoted by two quote characters together ("").

POSSIBLE MODIFICATIONS

The Tab key is a suitable choice for the super function action because it is physically big, not used very often, and gives the same code whether or not Shift is pressed at the same time. The @ key shares the latter two attributes, and you might prefer it for use with software which either re-defines Tab with *FX219, or commandeers it for some other use (as the BBC Master's Editor does). If so, the changes are simple. Lines 150 and 480 become:

```
150 T$="@=@@"
and 480 CMP#64:BEQ startzipping
```

One final point, the red function keys themselves are associated with character codes, normally 128 to 137. Such characters will be 'expanded' into their corresponding string if the code is inserted into the keyboard buffer, but not if it simply occurs within a string which itself is generated by a red key. Similarly, super function keys can be programmed with the codes of either red or super function keys without their being expanded. But a super function code will be expanded if it occurs within the definition of a red function key, with Tab represented by |I (or @ just by itself). This fact will be useful if you want to program a red function key to generate a list of your current super function keys.

If you are using this program with Basic I you will need to substitute a suitable replacement OSCLI procedure in lines 580 and 585.

```
10 REM Program SFKeys
20 REM Versiom B1.0
30 REM Author John Cole
40 REM BEEBUG October 1987
50 REM Program subject to copyright
60 :
100 M%=&900
110 N%=M%+&100
120 PRINT"SUPER FUNCTION KEYS:"
130 FOR T=M% TO N%:T=0:NEXT
```

```
140 pointer=N%
150 T$="|I=|I"
160 REPEAT
170 OSCLI(" *KEY9 "+T$)
180 *FX21
190 *FX138,0,137
200?(M%+GET)=pointer-N%+1
210 REPEAT:UNTIL GET=ASC("=")
220 REPEAT
230 pointer=pointer+1
240?pointer=INKEY(0)
250 UNTIL?pointer=255
260?pointer=0
270 READ T$:PRINTT$
280 UNTIL INSTR(T$,"")=0
290 FOR pass=0 TO 2 STEP 2
300 P%=M%+&B0
310 [OPT pass
320 PHP
330 LDA #entry DIV 256:STA &211
340 LDA #entry MOD 256:STA &210
350 PLP
360 RTS
370 .readchar
380 JSR !&210 MOD &10000:RTS
390 .startzipping
400 JSR readchar:BCS return
410 CMP#&B0:BCS startzipping
420 STA lookitup+1
430 .lookitup
440 LDA M%:STA entry+1
450 .entry
460 LDA N%:BNE zip
470 JSR readchar:BCS return
480 CMP#9:BEQ startzipping
490 CLC
500 .return
510 RTS
520 .zip
530 INC entry+1
540 CLC:RTS
550 ]
560 NEXT pass
570 PRINT"Definitions use ";pointer-N
%:" bytes out of 255"
580 IF pointer<N%+&100 OSCLI(" *SAVE Ke
ys "+STR$~M%+" +200 "+STR$~(M%+&B0)) ELS
E PRINT" . . too many!"
585 OSCLI("keys")
590 END
600 :
610 DATA T=try this key
620 DATA t= and this (it beeps!)|G
630 DATA End of definitions
```

B

ARCHIMEDES PRODUCT UPDATE

We have investigated the latest situation regarding the availability of software for the Archimedes, and report below for the benefit of potential purchasers. This list is by no means exhaustive, but does serve to indicate that the Archimedes market is off to a flying start. All prices quoted below include VAT, and are followed by the month when the relevant suppliers expect their products to be available.

COMPUTER CONCEPTS

INTERWORD, INTERSHEET INTERCHART. Emulator only. ROM only. None is yet available. No release date is known.

WORDWISE PLUS. An emulator version may be released soon on disc.

INTERBASE. Not to be released for Archimedes.

DISC DOCTOR II. All you'd expect from a disc utility. No release date planned.

"WORDPROCESSING PACKAGE". Available December. Written specifically for the Archimedes. This will probably be on ROM. Similar, but better than MacAuthor, with WIMPS, WYSIWYG, Magnify/Reduce, Auto-Indent, Multi-document, Multi-windows, Fonts, Spelling checker at 60,000 words per minute (an improved Spellmaster), Thesaurus, Multi Printer Drivers including Laser.

"DRAWING PACKAGE". Available on ROM for Christmas. Links to CC's wordprocessor and future products, will be fast, WIMPS, will include textual definitions of drawings which may be altered to change the drawing.

CLARES

TOOLKIT MODULE. £39.95. Available NOW. Disassembler, 5 memory editor formats. 4 search commands, compares, shifts, and memory dumps to printer. Single line assembler. ADFS utilities.

ALPHA BASE. £49.95. End of October. Compatible with the Model B's Betabase, free format input, window/icon driven, 400 fields, 27600 characters per record, range check, different access levels, passwords, multiple pages/screens per record.

ARTISAN. £39.95. End of October. Total

Graphics (Art) package featuring Magic Pen and Magic Rubber, uses WIMPS, and includes ECF designer, Sprite designer brush designer, user-controlled palette, Zooms with up to eight times magnification.

IMAGE WRITER. £29.95. Middle of October. A wordprocessor providing on-screen features reproducible on the Epson FX80, standard machine font, control window based, document (not page) based, graphics areas with graphics module (draw/load).

MINERVA SYSTEMS

DELTABASE. £29.95. Available NOW. A powerful, simple-to-use, database. Menu operation/pull-down Help panels. Over 8000 records, 255 fields. Records up to 8000 characters. Uses 40/80/132 column modes. Searching, maths and sort facilities.

SYSTEM DELTA PLUS. £69.96. Early November. Relational Database Management System. WIMP database allows search selection from within windows. Fully programmable using System Delta Commands. Programmer's Reference Guide available at extra £29.95.

SUPER DELTA. £199.95. First half of 1988.

REPORTER. £24.95. Early November. User definable reports, for use with Deltabase and System Delta Plus.

SYSTEM SIGMA. Price T.B.A. November. Spreadsheet Management System.

GAMMAPLOT. £34.95. Early November. General graphics program allowing the production of graphics and charts such as bar, pie, scatter, line and histogram.

PERSONAL ACCOUNTANT. Early November. Requires either System Delta, or System Delta Plus to run. Personal Accountant consists of the following individual packages: Order Processing, Sales Ledger, Purchase Ledger, Nominal Ledger, Stock Management. £64.95 each.

SPECIALIST APPLICATIONS. Requires either System Delta, or System Delta Plus to run. School Administrator £79.95, Newsagent £69.95. Ancestry, Timetabling prices T.B.A.

BBC SOFT

ARCCOMM. £24.95. January 1988.

ARCWORD. Price T.B.A. No release date. This wordprocessor is to be aimed at the every-day, rather than the advanced user.

BLACK QUEEN. Approx £15. November. Black Queen is a Contract Bridge program available for BBC/Master & Compact. The Compact version will work on the Archimedes. A specialist Archimedes version will be released in December.

ACORN

Without any warning Acorn announced on 24th September that from 1st October there will be a reduction of £115 (inc. VAT) on all 300 series Archimedes.

Acorn have now released the A310M, which incorporates the PC emulator at an extra £69 (inc. VAT) on top of the A310 price.

Acorn report "Masters are selling very well with between 80,000 and 100,000 sold. Anticipated sales are 130,000 by end of 1988. There is no predicted date for the "out-of-house" produced Advanced Master Manual.

I/O PODULES. Code:AKA10. £90.85. In stock.

ROM PODULE. Code:AKA05. £44.85. December. Acorn are still looking at what resident operational software to include.

MIDI ADD-ON I/O PODULE. Code AKA15. £33.35. The User Guide is currently being printed, and will be available in November.

BACKPLANE. Code:AKA01. £44.85. October. The Backplane includes the fan, and is currently in stock.

WINCHESTER. Code:AKD50. £573.85. December. The Winchester upgrade includes backplane.

2ND DISC DRIVE. Code:AKD50. £143.75. October.

PROGRAMMERS REFERENCE MANUAL. £19.95. Currently running to 600 pages, the manual will be available early November.

MONO SYSTEMS. The Mono Monitor for use with the 305/310 systems will not be available until November. "Entry" systems will run equally well with any other standard mono monitor.

WATFORD ELECTRONICS

REAL TIME MONO DIGITISER. Approx £170. Available late October.

SERIAL KIT. £28.75. Available NOW. The kit enables Archimedes to be linked with a BBC Model B or Master computer.

5.25" DISC DRIVE INTERFACE. £23.00. This enables a 5.25" disc drive to be connected up as drive 1. Includes a cable and buffer. Available NOW.

DATABASE SOFTWARE. Price T.B.A. Mid November.

PAINT PACKAGE. Price T.B.A. Late November. This package is Icon-driven and (according to Watford Electronics) is far superior to the Quest paint package.

BEEBUG

"MODEM". Hayes-compatible modem podule especially for the Archimedes. Provides V21 & V23 baud rates as standard, with V22 & V22bis rates with tone dial. Full Hayes "AT" command set together with battery backed phone number and 'S' register store. Line monitoring through computer's speaker, V25 auto-answer and selectable bell tones. Price T.B.A. Available December.

COMMAND. Communications software using WIMPS, featuring a range of terminals - Viewdata, VT52, VT100, Tektronix 4010, Teletype etc. Supports range of modems, facility to customise modem drivers. Wide range of file transfer protocols including XMODEM, YMODEM, KERMIT, CET. Features disc manager, phone directory, host mode. Self-contained comms language to control all its features. Price T.B.A. Available December.

MONITOR CONVERSION KIT. £40-£50 November. Enables the Archimedes to be connected to Microvitec and similar monitors. Will consist of a small interface box and leads.

ACORNSOFT

VIEW, VIEWSHEET, VIEWSTORE, VIEWINDEX. Separate versions are available now which run under the emulator.

VIEW PROFESSIONAL. £113.85. Spring 1988. The full Archimedes version contains many extra features, and will possibly be re-named.

LISP, £113.85

PROLOG, £113.85

LOGISTIX, £113.85

TWIN, £33.35

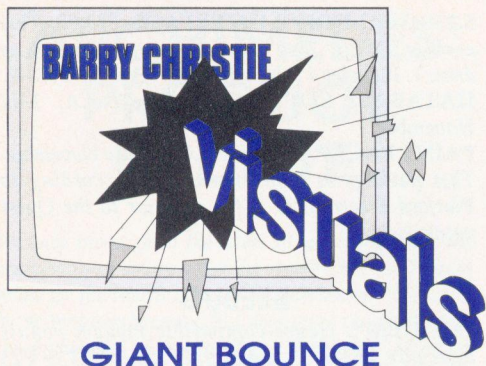
ZARCH, £19.95

All available now.

ANSI C, ISO PASCAL, FORTRAN 77. All at £113.85. Available now in pre-release versions. By completing the form in the pack you will get the complete production version when ready.

TERMULATOR. Available December.

B



GIANT BOUNCE

This month Barry Christie employs some tricks of the trade to create very fast animation effects.

Have you ever seen the giant bouncing ball demonstration on the Amiga? Well, this month we bring the Amiga to the BBC. At a time when the new Archimedes has cast the speed of the BBC in an unfavourable light, this program will show just what can be done on the faithful old Beeb, thanks to a few tricks.

In fact, as you can see, there are actually two programs accompanying this article. The first creates a large blue and yellow beach ball against a green background grid. The second program then uses a small bit of machine code to bounce this ball around the screen at very great speed. The ball twists and spins as it goes, making a 'blip' each time it hits the edge of the picture. The accompanying screen shot of course gives no impression of movement - the only way to see how well the Beeb performs is to try it out for yourself!

GETTING STARTED

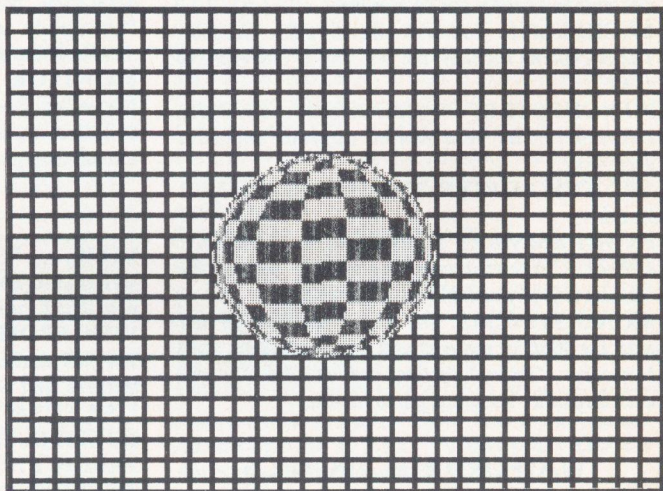
Begin by typing in listing 1, and save it away before running it. When you do run it, it should create a large blue and yellow beach ball in the

centre of a green grid; and should then automatically save away the resulting screen under the name *Bscrn*. If all is well, clear the machine, type in the second listing, and again save it away before running it. When you run the second program, it will load in the screen saved with program one, and then proceed to bounce and spin the ball around the screen.

HOW IT WORKS

If we had employed conventional techniques to move the ball around using the machine code equivalent of PLOT commands, the whole thing would have been extremely slow indeed. The speed is achieved here by moving not just the ball, but the whole screen. We do this by writing directly to the 6845 video controller chip. This creates what is described as a *hardware scroll*, and is almost instantaneous in its effect.

Listing 2 has the lines handling the scroll which are 690 to 710. They simply set the start address of the screen to a new value resulting from calculations carried out earlier in the program.



Bouncing ball

The program holds this address in locations *startL* and *startH*, and the major part of the program is concerned with updating this address in such a way as to represent the course

travelled by a bouncing ball. Each time the x and y co-ordinates of the ball are incremented or decremented, checks are carried out on the new co-ordinates to see if the ball would hit the edge of the screen. If so, the direction of travel is reversed as appropriate

THE 6845 REGISTERS

Screen start address:

Register 12 = Low byte

Register 13 = High byte

To alter these parameters, write to:

&FE00 to set the register number
and &FE01 to provide the data

Thus: LDA #12:STA &FE00
LDA #00:STA &FE01
will set the low byte screen start address to zero.

Of course, a hardware scroll would also shift the background too, but the grid has been drawn in such a way that each time that the screen is hardware-scrolled, the grid lines exactly overlay the previous grid. The background thus appears to stand still. To see how it's done, press Escape, engage mode 1, type some rubbish on the screen, then execute:

```
CALL &900
```

This invokes the scrolling software, and you will now see what is really happening: the whole screen is dancing around.

To add further to the effect of movement, the pattern on the ball is switched every hundredth of a second. To do this, we set one of the two component colours of the ball to flashing blue and yellow, and the other to flashing yellow and blue, then we change the flash rate to its fastest setting (lines 990 and 1000). As a result, the ball appears to spin very very quickly as it moves.

You may like to experiment with scrolling using the 6845 data provided in the accompanying table. If so, remember that to scroll the screen up or down by one line, you need to change the screen start address by 80 for modes 0-3, and by 40 for modes 4-6.

Listing 1

```
10 REM Program Beach Ball
20 REM Version B 0.2
30 REM Author Barry Christie
40 REM Beebug October 1987
50 REM Program subject to copyright
60 :
100 MODE 1
110 PROCinitialise
120 PROCbackground
130 PROCdrawball
140 MODE 7:END
150 :
160 DEF PROCdrawball
170 c%=1:radstp=1
180 radius=158:angstp=PI/14.5
190 REPEAT
200 PROCdrawface
210 radstp=2*radstp
220 radius=radius-radstp
230 UNTIL radius<=0
240 radius=0:radstp=32
250 PROCdrawface
260 *SAVE Bscrn 3000 8000 0000 3000
270 ENDPROC
280 :
290 DEF PROCdrawface
300 FOR angle=0 TO 2*PI STEP angstp
310 GCOL 0,c%:c%=(c% MOD 2)+1
320 PLOT 4,640+radius*SIN(angle),512+
58*COS(angle)
330 PLOT 4,640+(radius+radstp)*SIN(ang
le),512+158*COS(angle)
340 PLOT 85,640+radius*SIN(angle+angst
p),512+158*COS(angle+angstp)
350 PLOT 85,640+(radius+radstp)*SIN(an
gle+angstp),512+158*COS(angle+angstp)
360 NEXT angle
370 ENDPROC
380 :
390 DEF PROCinitialise
400 VDU 23,1,0;0;0;0;
410 *FX 9,0
420 *FX 10,0
430 VDU 19,1,11;0;
440 VDU 19,2,12;0;
450 VDU 19,3,2;0;
460 ENDPROC
470 :
480 DEF PROCbackground
490 FOR pass%=0 TO 2 STEP 2
500 P%=&900
510 [ OPT pass%
520 .clearscreen
530 LDY #&00
540 .loop1
550 LDX #&0F
560 .loop2:DEY:LDA screendata,X:
570 .inc:STA &3000,Y:DEX:BPL loop2
580 CPY #&00
```



```

590 BNE loop1
600 INC inc+&02
610 BPL clearscreen
620 RTS
630 .screendata EQU D &888888FF:EQU D &F
F888888:EQU D &111111FF:EQU D &FF111111
640 ]:NEXT pass%
650 CALL clearscreen
660 ENDPROC

```

* * * * *

Listing 2

```

10 REM Program Bounce
20 REM Version B 0.2
30 REM Author Barry Christie
40 REM Beebug October 1987
50 REM Program subject to copyright
60 :
100 MODE 1
110 ON ERROR GOTO 1040
120 VDU 19,1,11;0;
130 VDU 19,2,12;0;
140 VDU 19,3,2;0;
150 VDU 23,1,0;0;0;0;
160 *FX 9,0
170 *FX 10,0
180 startL=&80:startH=&81
190 stepsX=&82:stepsY=&83
200 coordX=&84:coordY=&85
210 incdec=&86
220 FOR pass%=0 TO 2 STEP 2
230 P%=&900
240 [ OPT pass%
250 :
260 LDA #&00:STA startL:LDA #&06
270 STA startH:LDA #&0F:STA coordX
280 LDA #&0B:STA coordY:LDA #&00
290 STA stepsX:LDA #&00:STA stepsY
300 :
310 .scrollball
320 LDA #&02:STA incdec
330 LDA stepsX:BMI subX
340 INC coordX:JSR increment:JMP skipX
350 :
360 .subX:DEC coordX:JSR decrement
370 :
380 .skipX:LDA #&50:STA incdec
390 LDA stepsY:BMI subY:INC coordY
400 JSR increment:JMP skipY
410 :
420 .subY:DEC coordY:JSR decrement
430 :
440 .skipY:LDA coordX:CMP #&1E
450 BNE Yokay1:LDA #&FF:STA stepsX
460 JSR blip
470 :
480 .Xokay1:LDA coordX:CMP #&00
490 BNE Yokay2:LDA #&00:STA stepsX

```

```

500 JSR blip
510 :
520 .Xokay2:LDA coordY:CMP #&16
530 BNE Yokay1:LDA #&FF:STA stepsY
540 JSR blip
550 :
560 .Yokay1:LDA coordY:CMP #&00
570 BNE Yokay2:LDA #&00:STA stepsY
580 JSR blip
590 :
600 .Yokay2:LDA startH:CMP #&05
610 BNE okays1:CLC:ADC #&0A
620 :
630 .okays1:STA startH
640 LDA startH:CMP #&10:BNE okays2
650 SEC:SBC #&0A:.okays2:STA startH
660 :
670 LDA #&13:JSR &FFF4
680 :
690 LDA #&0D:STA &FE00:LDA startL
700 STA &FE01:LDA #&0C:STA &FE00
710 LDA startH:STA &FE01
720 BIT &FF:BMI escapekey
730 JMP scrollball
740 .escapekey:RTS
750 :
760 .decrement
770 SEC:LDA startL:SBC incdec
780 STA startL:LDA startH:SBC #&00
790 STA startH:RTS
800 :
810 .increment
820 CLC:LDA startL:ADC incdec
830 STA startL:LDA startH:ADC #&00
840 STA startH:RTS
850 :
860 .blip
870 LDA #7:LDX # sound MOD 256
880 LDY #sound DIV 256
890 JSR &FFF1:RTS
900 :
910 .sound
920 EQU B &01:EQU B &00 \ Channel
930 EQU B &F7:EQU B &FF \ Volume
940 EQU B &01:EQU B &00 \ Pitch
950 EQU B &01:EQU B &00 \ Duration
960 :
970 ]:NEXT pass%
980 *LOAD Bscrn
990 *FX 9,1
1000 *FX 10,1
1010 CALL &900
1020 END
1030 :
1040 ON ERROR OFF
1050 *FX4,0
1060 MODE7:REPORT:PRINT" at line ";ERL
1070 END

```

B

BEEBUG LAUNCHES RISC USER

The Archimedes Support Group

After a very careful look at the Archimedes, we have decided that we cannot possibly do full justice to the vast potential of this machine within the existing Beebug format. We would either end up starving the Archimedes of vital coverage, or would be taking too many pages away from the model B and Master machines, whose own potential is very far from exhausted. We are therefore launching a sister user group and magazine for the new machine.

Existing Beebug members may either transfer their membership to the new magazine or, for only an extra 60p per issue, extend their subscription to include both magazines. A joint subscription will enable you to keep completely up-to-date with all innovations and the latest from Acorn and other suppliers on the complete range of BBC micros. **RISC User** has a massive amount to offer, particularly at this time while documentation about the Archimedes is of such a limited nature.

Here are some of the topics covered in the very first issue of **RISC User**, due out in a couple of weeks:

ARM ADFS MENU - Keep this short machine code program resident in your Archimedes for rapid access to all files and directories.

ARCHIE'S WHITE MOUSE - How to use the mouse from Basic, and how to incorporate mouse and pointer into your own applications.

VISUAL EFFECTS - Some short Basic routines producing stunning visual effects.

SOUND SET-UP - How to use the Archimedes sound system - with information on incorporating the extra instrument files on the Welcome Disc.

OFF-THE-SHELF - A look at some of the products available now for the Archimedes - including Wordwise Plus, Deltabase, ANSI C, Clare's Toolkit, and Acorn's Twin Editor.

ACORN HOTLINE - News of the new operating system release, and of software and hardware due for imminent release.

HINTS & TIPS - Dozens of vital hints and tips covering undocumented features of the new machine, short cuts, bugs, and errors in the manuals.

Plus much more including a look at Aliases, how to use the Basic Editor and Emulator, and articles on Welcome Paint, the new Basic structures, and using the Archimedes disc system.

Don't delay - Phone your instructions now on (0727) 40303

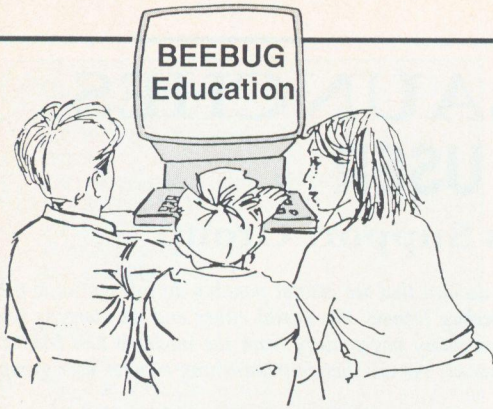
As a member of BEEBUG you may extend your subscription to include the first year of **RISC User** for only £6.00 (*overseas see below) - less than HALF PRICE! The initial introductory subscription to **RISC User** is £12.50 to the public, and the full rate from early 1988 will be £14.50.

Name:.....
 Memb No:.....
 Address:

SUBSCRIPTION DETAILS	
Destination	Additional cost
UK, BFPO & Ch Is	£6.00
Rest of Europe and Eire	£8.00
Middle East	£9.00
Americas and Africa	£9.50
Elsewhere	£10

I wish to receive both BEEBUG and RISC User. I enclose a cheque for £
 Alternatively I authorise you to debit my ACCESS/Visa/Connect account /_____/_____/_____/_____/_____
 Signed: Expiry Date /_____/_____/

Send to: **RISC User, Dolphin Place, Holywell Hill, St Albans, Herts AL1 1EX, or telephone (0727) 40303**



Six years on from the birth of the BBC Micro and the Government scheme (half funded by the Department of Trade and Industry) to put micros in schools, Mark Sealey takes a personal look at what has been achieved, and with the advent of the Archimedes, considers the future of the BBC family of computers in education.

ACHIEVEMENTS

From our point of view - as computer enthusiasts - the biggest success of the original machine is without doubt the proliferation of Acorn systems within schools with literally thousands of software titles to choose from. If the BBC models A and B had not been chosen for support in schools, Acorn might never have achieved the success it has.

For children, students and teachers, the sheer versatility of function has meant that, whatever figures you choose, the vast majority of British places of learning which use Acorn equipment have had a very good deal. This is evident, for instance, in the sphere of special education, or in schools and colleges which use modems; the choice of ports, peripherals and alternative input-output devices also outstrips the competition.

IMPENDING CRISIS

Yet there are three sour notes in assessing the impact of micros in the world of education - one of which is also of pressing concern to the hobbyist in general, namely the sobering thought that computers do not last forever. There are not many items in the Local

Education Authority budget that have required so heavy a capital outlay as the quarter of a million plus BBC computers sold within the education sector by Acorn since 1981/2. It is clear that these must reach the end of their useful, if not their natural life sooner or later.

Repairs can only be carried out as long as spares last. There were scare stories earlier this year about Acorn discontinuing its support for some parts. Michael Page at Acorn, however, has said categorically that "as long as there is a demand for spares, we will continue to supply them". Acorn sees education as one of two 'strategic markets' (the scientific market being the other), and will continue to promote and support sales here. However, not all is within Acorn's control - for example, the demise (at the cost of some anguish among users) of the 8271 Floppy Disc Controller chip.

The current crisis in education will not help either, as schools' budgets become more and more stretched, so the discrepancy will grow between the life-expectancy of micros on one hand and the slower rate at which other (say audio-visual) equipment is replaced on the other. The Secretary of State for Education himself admits that far too few children still have satisfactory access to computers. It has been estimated that a six-fold increase in spending would be necessary in order to provide adequate numbers of machines.

In the absence of any national policy on computers in schools, course designers and LEAs will be faced with a local choice, either to grit their teeth and update machines, or rely less heavily on all forms of computer assisted learning. I think most would agree that the latter course of action would be highly disadvantageous, not to say disruptive, to us all!

Moreover, the fact that the boom (which we all enjoyed some years ago) is over is evidenced by the decline in the volume of software production. Three or four of the major publishers are now no longer issuing or developing titles in anything like the same quantity. This should not matter significantly as

there is undoubtedly enough software to keep us all going for many years yet. But the loss of software writers could ultimately be more serious.

TRENDS IN COMPUTER USE

When the Microelectronics in Education Programme (MEP) was wound up last year, the Department of Education and Science published a formal assessment of its work. Some of the findings related directly to the uses to which computers are put. They do, I am afraid, make rather depressing reading. Although the main national agency was able to point to some impact in changing learning styles and dissemination of materials, it is admitted that this was often an uphill struggle. The uptake and employment of many excellent materials was (and still is) often slow. And given the importance attached to in-service training for teachers and lecturers it is desirable (but unlikely) that this should not be impeded in the future.

When you add to this the estimates that have been made of the ownership of peripherals (only about 60% of all Primary Schools, for instance, have disc-drives and still only 25% have printers), the picture is gloomy indeed. Two further factors are apparent: the computer is under-used and the programs often ill-chosen. Instead of the imaginative use of problem-solving and open-ended activities, all too often drill and practice predominate. Nor does very much software take full account of learning theory. I refer, for example, to children's own learning strategies, their response to the screen and how important it is to grade properly increases in the difficulty of tasks, or the order in which they are to be completed.

THE FUTURE

The enhanced facilities of the Master series mean that other areas of Information Technology such as Interactive Video can now be brought into the game. Such equipment could revolutionise some aspects of education in the future if their use was widespread enough. Beebug Education looked at the Domesday system in Vol.5 No.9. There is little

evidence, though, that schools are using this in any quantity or that software is appearing. This is sure to influence decisions on the purchase of new equipment. Why splash out on the extra facilities when the uses to which it could be put are so few? Imagine a video-disc tour written in Logo of, say, Bronze Age archaeological sites in Britain or places of entertainment in your area.

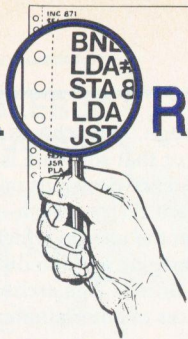
Which factors, then, will influence decisions about the future? There are, I believe, two futures: the short term, which is gloomy. Hardware, demoralised teachers and children, and software with some of the limitations alluded to, may not survive the damage currently being done to state education. If they can, then what the Archimedes range has to offer in increased power, compatibility and sheer ease of use should mean a very exciting period indeed in the long term.

The MEP survey makes the point that updating and converting software to run on later machines is vital. The 6502 emulation alone will not justify the purchase of Archimedes. If schools and colleges are to have the best, as they should, in each of the three broad areas of educational use (computer assisted learning, applications, and computer studies), the amount of software already commissioned and even completed prior to the launch of the 300 series last June makes it a very attractive buy. As the model B becomes obsolete, it may well be put to dedicated word-processing or Logo use, while those few LEAs now buying Masters may see them turned over, say, to desktop publishing and conventional CAL (Computer Assisted Learning). Music and CDT (Craft/Design/Technology) Departments could well expect Archimedes to gravitate to them.

Indeed, it may well turn out to be the twin bugbears of compatibility and teacher confidence that prove decisive in the long run. The software which has proved sturdiest, most used and most popular since 1981 has been the software with the greatest payoff between the results it can achieve and the effort needed to learn how to produce them. It follows that the systems on which this software will run must have a decisive advantage in the market.

B

EXPLORING



ASSEMBLER

Part 4

A series for complete beginners to machine code by Lee Calcraft

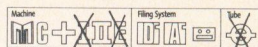
This month: Text Printing
Indirect Indexed Addressing
Mode 7 Screen Filling
and Generalised Memory Move Routine

In Part 3 we looked at the way in which indexed addressing could be used in a simple screen-fill routine. This month we will introduce a more powerful addressing mode called *indirect indexed addressing*, and will put this to work in a full mode 7 screen fill, and in a useful generalised memory move routine. The latter has a wide number of applications, from move-down routines to the implementation of multiple screens, as we shall see. But we begin with a final example of indexed addressing.

TEXT PRINTING

It is often useful to be able to print strings of text directly from machine code, as prompts, error messages or the like. And while this is not quite as easy as it is from Basic, it is nevertheless a relatively simple matter to create a generalised text printing routine in 6502 assembler using indexed addressing.

A routine to accomplish this is given in listing 1. It is complete in itself, and when run will display the text contained in lines 310 and 330. It will work on all machine configurations except Basic I and the Tube. To make it work across the Tube, simply replace the &900 in lines 150 and 370 with &3000. To make it work on Basic I, you should replace the EQU and EQU directives in lines 310 to 340 as outlined in the article in BEEBUG Vol.4 No.3 page 42;



though I would strongly recommend a Basic II upgrade for users interested in machine code.

To see how the routine works, you need to be conversant with the Basic II assembler directives EQU and EQU. These insert respectively an ASCII string, or a byte value into memory during assembly. Thus when the routine is assembled, the string "Any text less than 256 bytes in length." is stored one byte at a time starting at the location given by the label *text*. Immediately following this (i.e. at location *text+39*) the value &0D (carriage return) is stored, and so on.

Listing 1

```
100 REM Message printer
110 REM Version B 0.2
120 osascii=&FFFE3:osnewl=&FFFE7
130 MODE 7
140 FOR pass=0 TO 1
150 P%=&900
160 [
170 OPT pass*3
180 :
190 .start
200 LDY #&FF
210 LDA #&0D
220 .more
230 JSR osascii
240 INY
250 LDA text,Y
260 BNE more
270 JSR osnewl
280 RTS
290 :
300 .text
310 EQU "Any text less than 256 bytes
in length."
320 EQU &0D
330 EQU "Terminate with a zero byte."
340 EQU 0
350 ]
360 NEXT
370 CALL &900
```

When the machine code is run, it reads the characters found in this block of text one at a time, and sends them to the operating system to be written to the screen. As you may have noticed, we have used OSASCI rather than OSWRCH for this purpose. This is because the former, while identical to OSWRCH in all other respects, usefully sends a carriage return *and* a line feed when it is called with &0D in the accumulator.

A glance at the main loop, beginning on line 220, shows what is going on. The key instruction here is LDA text,Y at line 250. This loads the accumulator with the contents of the address found by adding the value of text to the contents of the Y register. The Y register starts at zero, so the accumulator is initially loaded with the byte at location text. This is ASCII "A", the first letter of the message. Then OSASCI is called, and the "A" is written. Y is incremented, and this process continues until a zero byte, marking the end of the string, is detected (BNE more). By using a zero rather than &0D as the terminator, we can easily print multi-line strings without having to call the routine separately for each line.

INDIRECT INDEXED ADDRESSING

In the last issue we looked by way of illustration at a mode 7 fill routine which filled the first 240 bytes of the screen with a character of the users choice. While the routine was both fast and efficient, it could not be readily extended beyond the 256 byte limit imposed by the maximum size of the Y register. The Y register, remember, was used as the index register in the so-called *indexed addressing mode*, and the screen was filled using the instruction:

```
STA 7BFF,Y
```

We can get around this 256 byte limit by using the much more powerful *indirect indexed addressing mode*.

To give an idea of the syntax, here is an accumulator load instruction using this addressing mode:

```
LDA (address),Y
```

This instruction has the following effect. Look up the *contents* of the two adjacent locations *address* and *address+1*, and use these to form a two-byte address (low byte first). This *indirect* address is then indexed by adding to it the contents of the Y register so as to form a new address. The accumulator is then loaded with the contents of this new address.

This is as powerful as it is involved. An example may help. Suppose that location &80 contains the value 0, &81 contains the value &7C, and the Y register contains the value 5. The result of performing the instruction:

```
LDA (&80),Y
```

will be to load the accumulator with the contents of location &7C00+5 (= &7C05). If we now increment Y, the load would be performed from address &7C06 (i.e. &7C00+6), rather as you might expect. But if you increment the contents of location &81 rather than the Y register, the load will be performed from location &7D05. In other words, from a point one page higher up in memory - thus immediately solving the problem of the 256 byte limit of the index register.

As you can see, this form of addressing will allow us to access any location in the 6502's memory map (&0000 - &FFFF), either by altering the value of Y, or by changing the base address (in this case the contents of &80 or &81). There are, however, a few provisos to bear in mind. Firstly, indirect indexed addressing may only be used in conjunction with the Y register. The X register may not be used as an index here, even though with indexed addressing both X and Y indices are permitted. Secondly, the base address must be held in a so-called *zero page location*; that is to say a location whose address is below &100. In the above example we used it with location &80, and in fact users of Basic and assembler on the BBC micro have at their disposal 32 unused zero page locations. These lie between &70 and &8F.

MODE 7 SCREEN FILL

Listing 2 makes use of indirect indexed addressing to implement a mode 7 fill routine capable of filling all 1000 bytes of the screen. The program is again self-contained, and when it is run it requests the ASCII value of a character to fill the screen. Any number between 32 and 126 may be entered, and on pressing Return, the whole screen will be immediately filled with the given character.

As you can see from the listing, the central routine has been broken down into three main sections preceded by a set of initialisation instructions. These latter load locations &80 and &81 with the low byte and high byte of the screen start address (i.e. 0 and &7C respectively), zero the Y register, and place the chosen character into the accumulator. The loop which begins on line 250 fills a 256 byte block of

the screen with the chosen character. On the first time around the loop, it stores the character at &7C00+0 (since Y=0 at the start), then &7C01 is filled, and so on until Y loops around to zero again. By this time a 256 byte block has been filled. The new instruction *INC* is then used in line 300 to increment the contents of location &81 (i.e. the high byte of the screen base address). Line 310 loads the new value into the X register, and line 320 compares this value to the constant &7F. On the first time around, location &81 will contain the value &7D, and the program will loop around to label *loop 1*, where a new 256 byte block will be filled.

You can see that the new block is immediately adjacent to the last because the instruction:

```
STA (&80),Y
```

on line 260 now refers to location &7D00+0 (since Y=0), while the last location to be filled by the first 256 byte block was &7CFF (i.e. &7C00+&FF), filled when Y contained the value &FF.

At the start of the second block, Y contains the value 0, and is incremented with each cycle of the loop until another block is complete. Then the contents of &81 are again incremented. This continues until location &81 contains the value &7F (line 320). The main fill then terminates, and the routine beginning at line 350 is engaged to mop up the remaining unfilled bytes. In fact there are 232 bytes still to fill because the target of 1000 is made up of $3 \times 256 + 232$. It is more efficient to fill these separately than to try to fill them from the main loop at line 260.

GENERAL PURPOSE MEMORY MOVE

Using indirect indexed addressing we can write an all-purpose memory move routine capable of moving a block of memory of any size between any two specified areas of RAM. The complete routine appears in listing 3, and as well as an assembler header, it contains a short demonstration in Basic. When the program is run, it assembles the code and then tries to load a mode 7 screen from disc. This is imaginatively called "screen", and on the magazine disc we have provided a copy of the BEEBUG Micronet front page for the purpose. With each press of the space bar, two alternate screens are now displayed - the Micronet screen, and one with a "Press the space bar" message on it. Each time that the Micronet screen is required, it is copied into screen RAM by the memory move routine, from a buffer (at &3000).

The routine has three parameters which must be supplied before it is called.

These are:

source	Address of source
dest	Address of destination
number	Number of bytes to be moved

The routine is called at two distinct points in the program, line 610 and 650. In the first case the parameters are set up to copy the mode 7 screen into the buffer area (at &3000), and in the

Listing 2

```
100 REM Full Mode 7 Fill
110 REM Version B 0.3
120 MODE 7
130 FOR pass=0 TO 1
140 P%=&900
150 [
160 OPT pass*2
170 : \**MODE 7 SCREEN FILL**
180 :
190 .start \ Initialise
200 LDA #0:STA &80
210 LDA #&7C:STA &81
220 LDA &70
230 LDY #0
240 :
250 .loop1 \ Fill 256 byte block
260 STA (&80),Y
270 INY
280 BNE loop1
290 :
300 INC &81 \ Repeat block 3x
310 LDX &81
320 CPX #&7F
330 BNE loop1
340 :
350 .loop2 \ Fill last 232 bytes
360 STA (&80),Y
370 INY
380 CPY #232
390 BNE loop2
400 RTS
410 ]
420 NEXT
430 :
440 REPEAT
450 INPUTTAB(5,15)" Enter Character no
"A
460 ?&70=A
470 CALL &900
480 UNTIL FALSE
```


Listing 3

```
100 REM Generalised Block Move
110 REM Version B 0.4
120 REM Call with source addr in &70
130 REM Destination address in &74
140 REM And size of block in &78
150 source=&70:dest=&74:number=&78
160 MODE 7
170 FOR pass=0 TO 1
180 P%=&900
190 [
200 OPT pass*3
210 :      \**GENERAL BLOCK MOVE**
220 :
230 .start
240 LDX number+1
250 BEQ loop2
260 LDY #0
270 :
280 .loop1      \Move 256 byte block
290 LDA (source),Y
300 STA (dest),Y
310 INY
320 BNE loop1
330 :
340 INC source+1 \Repeat block n times
350 INC dest+1
360 DEX
370 BNE loop1
380 LDX number
390 BEQ finish
400 :
410 .loop2      \Move last few bytes
420 LDA (source),Y
430 STA (dest),Y
440 INY
450 DEX
460 BNE loop2
470 .finish
480 RTS
490 ]
500 NEXT
510 :
520 *KEY0 OS. ("SAVE mover "+STR$~start
+" "+STR$~P%)|M
530 PRINT'"Press space between each s
equence"'
540 PRINT'"Alternatively, to save the
machine code"
550 PRINT"with the name MOVER"
560 PRINT'"Press Escape, then f0"'
570 A=GET:MODE 7
580 *LOAD screen 7C00
590 A=GET
600 !&70=&7C00: !&74=&3000: !&78=1000
610 CALL &900
620 REPEAT
630 CLS:PRINT'"Press space to restor
e screen"; :A=GET
640 CLS: !&70=&3000: !&74=&7C00
650 CALL &900
660 A=GET
670 UNTIL FALSE
```

second (which is called repeatedly) the routine copies from the buffer to the screen area. Note that the number of bytes, once set, does not need to be altered.

If you are going to use this routine in your own programs there are two points to note. Firstly the machine code may be saved by escaping from the demonstration program and pressing function key f0. Secondly, if the source and destination areas overlap in your application, you will get problems unless the destination is below the source in RAM (i.e. you must be moving down in memory). If the two areas overlap in an upward move, you will either need to use an intermediate buffer to avoid the clash, or modify the routine.

HOW IT WORKS

Just like the mode 7 screen fill routine, the generalised memory move uses indirect indexed addressing, and contains three main sections. The first (beginning at line 280) moves a 256 byte block of memory. It is quite straightforward, but now there is both a load and a store instruction. When the loop terminates (with Y=0), this part of the program will have moved a 256 byte block from the area designated by *source* to that given by *dest*. The instruction on line 340 then increments the value held at source +1 (i.e. the high byte of the source address), and line 350 does the same with the destination. You will see that X is then decremented. The reason for this is that X is being used to count the number of 256 byte blocks which are moved. Back at the start of the routine (at line 240) X was loaded with the high byte of the two-byte variable representing the number of bytes to be transferred (called *number*) If the high byte had been zero, then lines 280 to 390 would have been skipped (BEQ loop2 at line 250).

Once the required number of blocks has been transferred, a separate mopping up operation is engaged (again, just as in the previous program). This moves the number of bytes corresponding to the value held in the low byte of the two-byte variable number, and so completes the operation.

Next month we will introduce some of the 6502's arithmetical and logical instructions. **B**

BEEBUG WORKSHOP Precision Arithmetic

The Beeb's arithmetic is more precise (it works to more significant figures) than that of many other small computers. Although this is good enough for most purposes, you sometimes need greater precision. Surac explains how additional accuracy can be achieved, but at a price.

This month, I shall be looking at a way of performing arithmetic with an accuracy of up to 250 significant figures. It is slow compared to normal arithmetic, but it is still much faster than doing it by hand, and it works.

THE APPROACH

In this Workshop, I will confine myself to the basic tools of long precision addition and subtraction. I will cover multiplication and division in a later workshop.

Basically, the approach is to store numbers as strings, with each digit being represented by a separate character in the string.

For instance, the number 123.4567 is stored, precisely, as the string "123.4567". This is in direct contrast to the computer's normal method of storing numbers as binary patterns.

To make the computer's job easier, negative numbers are NOT stored as they look. Instead, they are held in "10s complement" form. This is equivalent to the "2s complement" form used with binary arithmetic, and makes addition and subtraction much easier without affecting multiplication or division

Unfortunately, I have not enough space to go into the detail of how it works, but briefly a number is converted to 10s complement form by subtracting each digit in turn from 9, and then adding 1 to the right-most digit. Thus, -123.4567 is represented as "9876.5433". The lefthand "9" is the clue that the number is negative.

BASIC TOOLS

The first listing (rather longer than usual for a Workshop) is a set of procedures and functions for basic long arithmetic. Let's look first at the tools which make it all possible.

PROCInit sets up the system and reserves a buffer in which the manipulation will be performed. This procedure needs a value (not more than 250 or the strings will overflow) to define the maximum number of digits to be handled.

FNLongEq converts a conventional number to the program's format (10s complement). It adds a decimal point to the right of integers, since the arithmetic routines require the point to be there. FN10comp carries out the 10s complementing operation, inverting each digit and adding 1. It does this manipulation in the buffer set up by PROCInit.

The final utility routine is FNLongPrint which is effectively the opposite of FNLongEq. It converts a number in the internal string format to a printable form; for instance, it re-inverts a negative number and puts a negative sign in front of it. It also removes any leading zeros (positive numbers) or nines (negative).

BASIC ARITHMETIC

With these routines out of the way, FNLongAdd adds together two numbers held as strings in the program's internal format. It actually does it in much the same way that you would probably do it by hand.

It first makes sure (line 30390) that the result won't be longer than the maximum length allowed by PROCInit. If necessary, it chops off the least significant decimals to make the string fit. It then aligns the decimal points (line 30400) and, at lines 30430-30480, adds matching digits, carrying if necessary, and puts the sum in the buffer. At the end of the exercise, the answer is in 'strbuff', which the function returns. That's it.

Once long addition is available, subtraction is easy to implement - see FNLongSub at line 30680. It uses the fact that (A-B) is just the same as (A+(-B)). It therefore takes the two numbers, uses FN10comp to negate the one which is to be taken away, and adds what is left together.

Long Arithmetic Procedures

```

30000 DEF PROCInit(SigDigs)
30010 maxlen=SigDigs+2
30020 DIM strbuff maxlen
30030 AscDOT%=ASC(".")
30040 Asc0%=ASC("0")
30050 Asc9%=ASC("9")
30060 Asc10%=Asc9%+1
30070 ENDPROC
30080 :
30090 DEF FNLongEq(str$)
30100 IF INSTR(str$,".")=0
    THEN str$=str$+"."
30110 IF LEFT$(str$,1)="+"
    THEN str$=RIGHT$(str$,
        LEN(str$)-1)
30120 IF LEFT$(str$,1)="-"
    THEN str$=FN10comp(RIGHT$(
        str$,LEN(str$)-1))
    ELSE str$="0"+str$
30130 IF LEFT$(str$,1)=""
    THEN str$="0"+str$
30140 =FNPad(str$)
30150 :
30160 DEF FN10comp(str$)
30170 LOCAL C%,D%,I%,P%,Asc948%
30180 Asc948%=Asc9%+48
30190 $strbuff=STRING$(maxlen-
    LEN(str$),"0")+str$
30200 FOR I%=0 TO (maxlen-1)
30210 C%=strbuff?I%
30220 IF C%<>AscDOT%
    THEN strbuff?I%=Asc948%-C%
30230 NEXT
30240 I%=maxlen-1
30250 IF strbuff?I%=AscDOT%
    THEN I%=I%-1
30260 REPEAT

```

```

30270 IF strbuff?(I%-1)=AscDOT%
    THEN P%=2 ELSE P%=1
30280 strbuff?I%=(strbuff?I%)+1
30290 D%=strbuff?I%<Asc10%
30300 IF NOT D%
    THEN strbuff?I%=Asc0%
30310 I%=I%-P%
30320 UNTIL D%
30330 =$strbuff
30340 :
30350 DEF FNLongAdd(str1$,str2$)
30360 LOCAL C1%,C2%,CY%,D1,D2,
    I1,I2,SUM%
30370 I1=FNIntLen(str1$):
    I2=FNIntLen(str2$)
30380 D1=FNDecLen(str1$):
    D2=FNDecLen(str2$)
30390 IF (FNMax(I1,I2)+FNMax(D1,D2)+1)>
    (maxlen-1) THEN PROCLimit
30400 PROCAlignDP
30410 str1$=FNPad(str1$):
    str2$=FNPad(str2$)
30420 CY%=0
30430 FOR I%=maxlen-1 TO 0 STEP -1
30440 C1%=ASC(MID$(str1$,I%+1,1)):
    C2%=ASC(MID$(str2$,I%+1,1))
30450 IF C1%<>AscDOT%
    THEN SUM%=C1%+C2%+CY%-48
    ELSE SUM%=AscDOT%
30460 IF SUM%>Asc9%
    THEN SUM%=SUM%-10:CY%=1
    ELSE CY%=0
30470 strbuff?I%=SUM%
30480 NEXT
30490 =$strbuff
30500 :
30510 DEF FNMax(A,B)
30520 IF A>B THEN =A ELSE =B
30530 :
30540 DEF PROCLimit
30550 VDU7
30560 IF D1>D2
    THEN str1$=LEFT$(str1$,
        maxlen-(D1-D2))
30570 IF D2>D1
    THEN str2$=LEFT$(str2$,
        maxlen-(D2-D1))
30580 ENDPROC
30590 :
30600 DEF PROCAlignDP
30610 LOCAL D1,D2,L1,L2
30620 L1=LEN(str1$):L2=LEN(str2$):
    D1=FNDecLen(str1$):
    D2=FNDecLen(str2$)
30630 IF D1<D2
    THEN str1$=RIGHT$(str1$,
        L1+D1-D2)+STRING$(D2-D1,"0")
30640 IF D2<D1
    THEN str2$=RIGHT$(str2$,
        L2+D2-D1)+STRING$(D1-D2,"0")

```



```

30650 str1$=FNPad(str1$):
      str2$=FNPad(str2$)
30660 ENDPROC
30670 :
30680 DEF FNLongSub(str1$,str2$)
30690 =FNLongAdd(str1$,
              FN10comp(str2$))
30700 :
30710 DEF FNPad(string$)
30720 LOCAL pad$
30730 IF LEFT$(string$,1)="9"
      THEN pad$="9" ELSE pad$="0"
30740 =STRING$(maxlen-LEN(string$),
              pad$)+string$
30750 :
30760 DEF FNIntLen(str$)
30770 LOCAL I%,sign
30780 $strbuff=str$
30790 sign=?strbuff
30800 I%=0
30810 REPEAT
30820   I%=I%+1
30830   UNTIL strbuff?I%<>sign
30840 =INSTR(str$,".")-I%-1
30850 :
30860 DEF FNDecLen(str$)
30870 =LEN(str$)-INSTR(str$,".")
30880 :
30890 DEF FNLongPrint(str$)
30900 LOCAL I%,OK%,sign$,temp$
30910 IF LEFT$(str$,1)="9"
      THEN str$=FN10comp(
              FNPad(str$)):sign$="-"
      ELSE sign$=""
30920 IF RIGHT$(str$,1)="."
      THEN str$=LEFT$(str$,
              LEN(str$)-1)
30930 I%=1
30940 REPEAT
30950   OK%=MID$(str$,I%,1)<>"0"
30960   IF NOT OK% THEN I%=I%+1
30970   UNTIL OK%
30980 temp$=sign$+RIGHT$(str$,
              LEN(str$)-I%+1)
30990 IF temp$="" THEN temp$="0"
31000 =temp$

```

THE ROUTINES IN USE

To see the routines at work, add the short program in the second listing to the procedures. This will also help to show how the various procedures and functions should be used. When you run the complete program, you may enter long numbers, see how they are stored internally and look at the results.

Running the program also makes apparent the main limitation of this approach - it is painfully slow compared with normal computer arithmetic. Although the programs here are far from optimised, there is no way out of the fact that the technique is NOT quick. Even so, it's still much faster than doing it by hand.

Main Program

```

1000 MODE0
1010 INPUT "How many digits? "N%
1020 PROCInit(N%)
1030
1040 REPEAT
1050   INPUT "'A"? "A1$
1060   INPUT "B"? "B1$
1070   A$=FNLongEq(A1$)
1080   B$=FNLongEq(B1$)
1090   PRINT "'A stored as: "'A$
1100   PRINT "B stored as: "'B$
1110   AB$=FNLongAdd(A$,B$)
1120   PRINT "'A+B: "'FNLongPrint(AB$)
1130   PRINT "Stored as: "'AB$
1140   AB$=FNLongSub(A$,B$)
1150   PRINT "'A-B: "'FNLongPrint(AB$)
1160   PRINT "Stored as: "'AB$
1170   UNTIL 0
1180 END

```

Next month we will take a look at long precision multiplication and division.

B

GHOST HOST (continued from page 11)

```

790 GOTO 250
800 REPORT:PRINT" at
";ERL:END
810 :
1000 DEFPROCSETUP
1010 *RINGS 1
1020 *STANDARD 5
1030 *V23A
1040 *DISCONNECT
1050 *ECHON
1060 CLOSE#0
1070 OSWORD=&FFFF1
1080 pass%=3:passwo
rd$="PASSWORD"
1090 DIM bf 100
1100 ENDPROC
1110 :
1120 DEFFNNonline
1130 A%=&7B:X%=&82:Y%=0:??&82=1
1140 CALL OSWORD=?&82 AND 1
1150 :
1160 DEFFNin
1170 A%=&7B
1180 X%=bf MOD 256:Y%=bf DIV
256
1190 ?bf=6:?(bf+1)=80:
?(bf+2)=0
1200 ?(bf+3)=100
1210 CALL OSWORD
1220 IF ?bf=255 ?bf=0
1230 ?(bf+1+?bf)=13:PRINT
$(bf+1)
1240 *SAY"|M|J"
1250 =$ (bf+1)

```

B

**ELECTRON &
BBC MICRO
USER SHOW**

**Royal Horticultural Hall
Westminster, London SW1**



SEE the first software, hardware and books for the Archimedes from companies like Acornsoft, Clares, Minerva, Computer Concepts, and many others.

BUT THAT'S NOT ALL!

- ★ Hardware and software galore for BBC Micro and Electron owners
- ★ Lots of bargains for the BBC Micro and Electron at rock-bottom prices
- ★ Technical advice from the experts over the whole range of Acorn machines

10am-6pm, Friday November 13
10am-6pm, Saturday November 14
10am-4pm, Sunday November 15

Here's your personal invitation to try out the revolutionary Archimedes for yourself

Be one of the first to play Zarch, the spectacular new game that is the first to make use of the Archimedes' incredible speed to magnificent effect – the four-directional scrolling really has to be seen to be believed!

Come along and meet the author, David Braben, creator of Elite. Try your hand at the game itself... and prepare to be amazed!

All this – and much, much more – at the 17th record-breaking Electron & BBC Micro User Show.

SAVE £1 A HEAD WITH THIS ADVANCE TICKET ORDER



Please supply:

- Adult tickets at £2 (save £1)..... £
- Under-16s tickets at £1 (save £1) £
- Cheque enclosed made payable to Database Publications Ltd. Total £ _____
- Please debit my credit card account

No.

Signed.....

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

Advanced ticket orders must be received by Wednesday November 4, 1987.

Post to: **Show Tickets, Europa House, 68 Chester Road, Hazel Grove, Stockport SK7 5NY.**

Name.....
Address.....
.....
.....

PHONE ORDERS: Ring Show Hotline: 061-480 0171
PRESTEL ORDERS: KEY *89, THEN 614568383
MICROLINK ORDERS: MAILBOX 72:MAG001

Please quote credit card number and full address

A147

Personal Ads

TEAC drive 40T 100k, Computer Concept ROMs: Wordwise Plus, Printmaster and Graphics extension. Also Spellcheck II and Beebcalc, £150. Spy2 £18. Acorn DNFS £17. QUEST paint and Mouse £55. Tel. Upminster (04022)23811 eves.

Master 128 with manual. Microvitec high resolution colour monitor. Cumana double sided 80T drive with power supply. Twin joysticks. Taxan-Kaga 80 column printer. Sharp cassette player. Many BBC games on cassette. Back issues of BEEBUG. All excellent condition. £850 ono. Tel. Swindon (0793) 610276 eves.

Unwanted present: **Watford Electronics Quest Mouse** and Quest Paint plus AMX Art ROM with utilities disc and AMX Desk disc for sale at £85 ono. Tel. Luton (0582)581757.

WANTED following issues of **BEEBUG magazine**: Vol.1 Nos.1,2,3,5&8. Tel. (0249)653893.

Master 128, 512 co-processor complete with Gem software and mouse, Microvitec med res colour monitor, Twin 40/80T DS drive (800k), 40T SS drive (100k), Data Recorder (tape), Twin Joysticks, ROM cartridge, View Store (ROM), Spellcheck (ROM). All with a wide range of software, manuals and a selection of blank floppies. £900 ono. Tel. Trevor (0252)871697.

BBC Model B, 1.2 O.S., BasicII + Wordwise. Computer recently fully serviced, excellent condition. £200 ono. Tel. Perth (0738)32729.

Back issues BEEBUG. The complete set from Vol.1 No.1 to Vol.5 No.9. £25, post paid. Tel. (0736)64573.

Seikosha GP-100A MkII printer with tractor feed and cut sheet attachment, as new in original box, £75. Wordwise Plus ROM with manuals. Typing tutor and program cassette, £30. Tel. Brian (0892)21021.

BBC B 1.2 OS, Acorn DFS, 40T disc drive, games, lockable disc box, 50 DDDS discs, mags, all leads and manuals, £320. Tel. (0382)68601.

Viglen console unit for BBC, £25. ATPL ROM/RAM board for BBC, £20. Tel. Salisbury (0722)27057.

AMX Mouse plus SuperArt package for the BBC Master 128 with manual. All originals, £39. Tel. 01-500 5267, eves.

Master 128. Unwanted gift. Boxed as new, full warranty. Never used. Offers around £410. Tel. (0992)34244 eves. and weekends.

Modem Nightingale, little used, £65 ono. Ultra Calc II, £25. J.Crabtree. 10 Pathfields, Dartmouth, Devon TQ6 9PH.

Z80 Second Processor complete with original documentation, discs and operating system, £150. Tel. (0634)67027.

BBC B, Single Teac and double Cumana disc drives. Solidisc DFS (cost over £200). OEL Telemod 2 for Prestel. Radio Shack TRS 80 printer DMP 100, all with interconnecting cables. Must be worth £450/500, but offers will be considered. Tel. (0206)384505.

BBC B with Watford DDFS, Wordwise Plus, AMX Mouse package. All with manuals and in original package. Plus lots of software. £250 ono. Tel. High Wycombe 882709 eves. and weekends.

WANTED **BEEBUG Vol.1** complete, good condition. Tel. Locks Heath (04895)3582.

BBC B 1.2 OS, series 7. Watford DD Disc Interface and ROM fitted. Complete with dust cover. Immaculate condition, £230. Phone Grays (0375)380369.

WANTED Teletex adaptor for model B. Tel. Glasgow 041-887 7200, after 6pm.

Graduate IBM PC compatible add-on to the BBC computer with software, £350. Z80 second processor with software, £200. 6502 Processor, £100. Viglen 40T disc drive 100k, £85. Tel. 01-391 0476.

ROM/RAM Expansion for BBC B. Ramamp RA32 board gives a total of ten ROM slots. Includes utilities for printer buffer, fast *Backup, DFS at &E00 etc, £25. Tel. 021-454 4307.

View and ViewSheet ROMs, complete with manuals, reference cards, key strips & fitting instructions, boxed, £25 each. Tel. (09323) 44999.

6502 Second processor and BITSTIK 1, £175. RH lightpen with 3 discs and manuals, £15. Tel. (0223) 880419.

BBC Master 128 + View family, Mouse, software, games, utilities, languages, 32k RAM card, £380. Philips monitor 80, £50. Twin disc drive 40/80 DS/DD, £150. Together £560. Canon PW 1080A printer and fontaid, £220. The lot £770. Tel. (0865)224834.

Acorn Z80 second processor plus all cp/m discs and books, also Wordstar, £180 ono. Tel. Martin (0703)266321 ext.255, office hours.

Sciways for Scientists

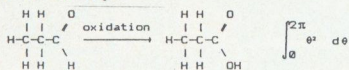
- * Over 350 defined characters accessible with simple 2-key codes
- * All characters printable on both screen and printer
- * User defined characters can be stored on disc
- * All characters and facilities can be used with word processing programs or with BASIC
- * Tested with BASIC I & II, Wordwise, Wordwise Plus, View 2.1, View 3.0, on the Master 128, B+ and Model B, and with Epson/compatible printers

MAIN FEATURES:

Greek Alphabet: The full Greek alphabet is supported in upper and lower case, and in upright and italic

Α Β Γ Δ Ε Ζ Η Θ Ι Κ Λ Μ Ν Ξ Ο Π Ρ Σ Τ Υ Φ Χ Ψ Ω
α β γ δ ε ζ η θ ι κ λ μ ν ξ ο π ρ σ τ υ φ χ ψ ω

Scientific Characters: A large number of scientific characters is available, mainly mathematical but also those used in chemistry



$$\int_0^{2\pi} \theta^x d\theta$$

$$\left\{ z: \arg \left[\frac{z-1}{z+3} \right] = \pi \right\} \cup \left\{ z: \arg \left[\frac{z-1}{z+3} \right] = -\pi \right\}$$

Orders accepted from schools, colleges, establishments, etc. Private orders — cheque with order, please.

PRICE: £38.52 inc VAT, p&p
16K ROM, 40/80 disc
and manual

MAYHEW TELONICS,
376a RINGWOOD ROAD,
POOLE, DORSET BH12 3LT
Tel. (0202) 747695

(not compatible with
Inter-Word, external shadow
RAM boards)

TubeLink

Advanced BASIC for BBC B/B+/Master

TubeLink is proud to offer the most powerful version of BASIC ever written for a 6502-based BBC micro. Advanced BASIC (formerly Archie BASIC) is a new version of BASIC, compatible with Acorn's new Archimedes BASIC for the BBC micro models B/B+/Master with 6502 second processor or Turbo.

Just look at some of the features it offers:

- Over 35 new keywords
- New program loop structures (WHILE, etc)
- Multi-line 'IF' statements
- New graphics commands (CIRCLE etc)
- New error handling and debugging modes
- Instant on-screen help on keywords and status
- New statements and functions
- Mouse support (new MOUSE keyword)
- New string handling features
- INSTALL and LIBRARY commands for procs
- Extended assembler
- Includes Acorn's very latest 1987 Hi-BASIC

Advanced BASIC can even be used to run Archimedes programs on your model B/B+/Master, or alternatively for writing programs to be later run on an Archimedes micro.

Advanced BASIC is supplied on disc for loading into a sideways RAM system, or on 16k EPROM, together with comprehensive user guide and 1987 Hi-BASIC on disc under license from Acorn. (To use extended graphics commands on B/B+ requires Acorn's GXR ROM. Mouse commands require suitable driver ROM, eg. Chauffeur or AMX Super ROM). State make and model of second processor in use. Full leaflet on request.

£29.95 on 40/80 track disc

£34.95 on 16k EPROM

Unique offer: Advanced BASIC and PMS B2P 65C02 second processor for just £115 - saving £15 on the normal RRP. Includes Advanced BASIC, second processor, instruction guides, and Tube Toolkit ROM. Add £5 if Advanced BASIC ROM version required. (B2P is only suitable for model B use).

UK Postage FREE on all items (Overseas please add £3). FREE catalogue on request. Dealer enquiries welcome. Access/Visa accepted. Phone 01-205 9393 (24hrs) for more information

TubeLink, PO Box 641 LONDON NW9 8TF

Advertising in Beebug

For advertising details,
please contact

Yolanda Turuelo
on

(0727) 40303

or write to
Dolphin Place
Holywell Hill
St. Albans
AL1 1EX

BEEBUG

Archimedes Competition Results

We were overwhelmed with the number of entries received in our competition to win a brand new Acorn Archimedes computer system. I shudder to think of the total number of hours spent in the task of designing and writing all those programs. Next month we will be publishing more details on the winning program. The program is too long to justify inclusion in the magazine, but will be included on the magazine disc and cassette.

JUDGING THE ENTRIES

All the program timings referred to below were obtained on a standard BBC model B with OS1.2 and Basic II. The shortlisted programs were then also tested on an Electron, B+, Model B and B+ with 2nd processor, Compact, Master 128, and finally an Archimedes A310. All the shortlisted 10 programs ran perfectly on all machines, except one program which did not run on the Archimedes. This entry was still accepted as the rules had not specified that it needed to run on an Archimedes.

We were amazed at the range of run times submitted, the longest being 1,003,960 seconds (11.5 days), the shortest claimed was 1.06 seconds, while many entries stated that there were NO solutions to:

HORSE+OFTHE+YEAR=HARVEY.

However, most of the entries did find the only solution to this alphametic of:

19037+96817+4720=120574.

There were over 40 entries having total run times of less than 50 seconds. Finally the 20 entries under 30 seconds were called in for further testing, including two entries which claimed total run times of less than 2 seconds! During testing we discovered that these entries were specifically written for the "HORSE" alphametic, and one would not even let other alphametics be entered - these were rejected.



Mr. P. Barrett receiving his prize from Graham Stanley, BEEBUG Showroom Manager

Other shortlisted programs crashed, or gave incorrect answers when tested on a range of alphametics. However, we finally finished up with 10 fully tested and working programs, the fastest of which tested all solutions to "HORSE" in 8.17s (1.16s on Archimedes, some seven times faster), and averaged only 8.98s for all our test Alphametics. The most tricky alphametic in our testing series was:

RIDER+DROPS+REINS+HORSE=LOSES

with two solutions. The winning program solved this in 27s and the slowest of the final 10

programs took 428s although:

MINCE+PIES=EATEN

with 16 solutions allowing zeros, and 3 solutions without zeros, also caused a wide range of timings from 14s to 74s.

Meaningful Alphametics are very difficult to originate, so much so that I must pay credit where due to the authors. BEEBUG member Douglas St P. Barnard, author of the "Braintwister" column in the Daily Telegraph for over 25 years, has published Alphametics from time to time in his column. Douglas has supplied me with the authors for those he is aware of:

CROSS+ROADS=DANGER

is by J.A.H.Hunter,

HORSE+OFTHE+YEAR=HARVEY

is by Peter H.Mabey,

RIDER+DROPS+REINS+HORSE=LOSES

is by Hugh ApSimon. I have not yet succeeded in finding an author for:

MINCE+PIES=EATEN.

After all the testing, the winner is Mr P.Q. Barrett of Hemel Hempstead, Hertfordshire, who is now the proud owner of an Acorn Archimedes computer. Our congratulations go to him.

RUNNERS UP ARE:-

2nd Alan Dickinson of Oxford

3rd Marion Fielding of Reading

4th Adrian Wrigley of Westerham, Kent

5th Tom Kibble of Richmond, Surrey

6th Ross Baldwin of Deptford, London

7th O.J. Hodder of Windlesham, Surrey

8th Natalie Wood of NSW, Australia

9th E.V. Pegge of Machynlleth, Powys

We are very pleased by the generosity of the following companies who have provided additional prizes for the runners up as follows:-

COMPUTER CONCEPTS - Three Inter-Series or Spellmaster products.

WATFORD ELECTRONICS - 64k ROM/RAM Board or 32k Solderless board with 16k RAM and battery.

VOLTMACE - Two 3B single joysticks.

TAXAN - KX 117 Green Screen Monitor.

CUMANA - CSX100 5" disc drive or CS354 3.5" disc drive.

AMS - AMX Design

We would like to thank everyone who entered our Archimedes competition for making it such a success. We hope you enjoyed the challenge which the competition presented.

Fast Flexible Professional Software

Individual programs designed to integrate. Use separately or link together to form a powerful system. Systems delta, Gamma and Sigma designed by highly acclaimed professional programmers with proven ability on the BBC, now providing definitive software for the Archimedes.

ARCHIMEDES

- DeltaBase** £29.95 Extremely powerful and simple to use general Database (160 records, import data from other databases)
- System DeltaPlus** £69.95 Extensive relational Database Management System. WWP card saves 2 billion records
- GammaPlot** £34.95 Versatile Graphics program. Multiple charts on screen at any one time
- System GammaPlus** £69.95 Powerful Graphics Management System. Available October 1987
- System Delta Personal Accountant** £69.95 each module The only flexible accounting suite. User-modifiable and integrate
- Order Processing/Invoicing** Sale Ledger Stock Management Purchase Ledger Nominal Ledger
- Specialist Applications** £89.95 Flexible specialist software available for school Administrator, Video Rental, Bookshop, Hotelier, Company
- Minotaur Game** £18.95 Full feature 3 dimensional animated maze type game. Use brains or force to kill the Minotaur

BBC MICRO

- System Delta with Card Index** £64.95 Join the most comprehensive and flexible data management system on the BBC
- Multihost Reporter and InterView Line** £25.95 each Individual record & 2 PCs. Whole text search, name & year for any internal & add
- System Delta Personal Accountant** £45.95 (Nominal £25.95) Full comprehensive accounting module which may be integrated if
- Order Processing/Invoicing** Sales Ledger Stock Management Purchase Ledger Nominal Ledger
- School Administrator** £69.95 Pull in a staff record, add extra fields if desired. Entry reports for examination boards. Form
- System Gamma** £60.95 Comprehensive management program for travel. Video Rental
- Specialist Applications from £54.95** The first Graphics Management System. Anyone can produce stunning graphics
- Font Generator** £19.95 Create, edit or print fonts on the fly. Any one can produce stunning graphics

COMPUTER SOFTWARE AND SYSTEMS SPECIALISTS 69 Sidwell Street Exeter Devon EX4 6PH Telephone: (0392) 37756

Tandy CGP-115 colour graphic printer/plotter with BBC printer cable, dust cover, spare paper etc. Excellent condition in original packaging, only £50. Tel Ripon (0765)87413.

BBC B with Torch Z80 and perfect software. Twin 40/80 disc drives in monitor stand housing, PSU, Philips monitor, recorder, joystick, £660+. Tel. Oxford (0865)880362.

Acorn 40T disc drive, £50. Games on cassette: Missile Base, Snapper, Castle of Riddles, Philosophers Quest, Snooker, Planetoid, Sphinx Adventure, 75p each. Novex colour Monitor (needs repair to power supply), £50. Tel. 061-881-2970, after 6p.m.

Wordwise Plus with Bruce Smith's Users Guide, £30. Ultracalc, with BBCsoft Utilities, £30. Acornsoft printer driver generator (unused), £4. View, ViewSheet & ViewStore manuals £3 each. Also number of books at half price or less. All items as new. Tel. Royston (0763)42593 after 7pm. & weekends.

Watford 128K ROM/RAM board with 16k battery backed RAM + read/write protect switches. Complete with manual and disc, £75 ono. Watford 32k Shadow RAM board and Interword, new unregistered with book "Understanding Interword", all for £75 ono. Solidisc DFDC inc. latest DFS and ADFS ROMs with manual, £25 ono. Acorn ADFS ROM with manual, £20 ono. ACP Advanced 1770 DFS ROM with manual for Master, £25 ono. The Music System (8271 version), £20. Tel. Dartford (0322)348281 after 6p.m.

WANTED BEEBUG Volume 1 complete. Tel. 01-631 3797 ext.3792, 9am-6pm weekdays, 01-672 7745 eves, weekends.

BBC B, DFS/DDOS, ATPL board, View, ViewSheet, Toolkit, Acorn 6502 second processor, £325 ono. Tel. Bedford (0234)750748.

BBC B, series 7, Watford DFS, Torch dual 40/80T drives, Sleuth Basic Debugger and Graphics ROMs, Datacorder, joystics. User guide, Advanced User Guide, Assembly Language manual, plus various software and publications, £420. Might split. Tel. (0775)60005.

Watford 32k RAM/RAM Board, expandable to 128k, fitted 16k CMOS RAM - see BEEBUG Vol. 5 No.6, £25. Also RAM AMP 6 ROM extension board, £10. All manuals supplied. Tel (0932)66076.

Microtext Plus ROM version, complete with discs and all manuals. As new, still boxed, £125. Tel. (0304)812943, eves. and weekends.

WANTED: Software to enable Oberon International's OMNI-Reader to learn new type fonts. Also, details of any other software available. Tel (0242)527798.

BBCsoft Monitor ROM + manual, £12. Acornsoft tapes: Elite £3, Castle of Riddles £1. Tel. Nottm (0602)231395.

Acorn Z80 second processor, with all documentation and software, £190. ACORN 65602 second processor £100. Music 500 with 5000 software and manuals together with APTL symphony keyboard plus real time software, £125. Tel: Hornchurch (04024)74633 eves.

Compact full colour system plus Epson LX86 printer, both under guarantee. Genuine offers. Tel. (0604)870465.

BBC B series 7 with APTL ROM board, Floppywise ROM, £245. Wordwise+ ROM and handbook £25. Watford ROMspell £18. Watford 32k Shadow RAM £40. Two Hitachi DSDD 40/80 disc drives in Technomatic plinth £145. Epson MX80 printer + acrylic stand £80. All mint condition. Tel. (0234)750050.

ViewStore database ROM £15, BEEBAID utility ROM £15. MAX ROM £10. ELITE on disc £8. All original and in good condition. Tel. Ian, Reigate (0737)241033 after 5 p.m.

Colour monitor suitable for full 80 column display (0.38mm dot pitch) with BBC B or IBM PC. Little used Taxan Kaga RGB III with manuals and leads, £250 ono. Tel: 01-624 3698 eves/weekends.

ATPL ROM board with 16k RAM £28/ Disc Doctor + Toolkit ROMs £10 each. BBC B keyboard £12. Interword complete £40. Tel. Cambridge 833203.

BBC B issue 4 with DFS, Wordwise+, 32k Sideways RAM, Canon Twin 40/80T disc drives with power supply, KAGA green monitor, Star DP510 dot matrix printer. Many discs and reference books. Price £500. Tel. Ferndown, Dorset (0202)871122.

BEEBUG Spellcheck III complete and boxed, ROM, 80T disc, manual, £18. Tel. (0492)531584.

Watford NLQ ROM still in original packaging, £22. Tel. (0865)66426 after 7pm.

6 View user guides, new (1st and 2nd edition), £5 each. View 3.0 ROM pack with user guide, printer driver generator etc, £35. 3 Wordwise 1.3 ROM packs (new), £13 each. Acornsoft games (disc): Freefall, Arcadians, Sphinx Adventure, Meteors, Super Invaders, Micro Power Magic, Fistful of Fun and lots more, as new, £2 each or £24 the lot. Tel: New Milton (0425)617338 eves.

BBC Model B with Watford 12 ROM board + 16k RAM and 32k RAM card. Philips mono (green) monitor, Quendata printer, Pace SS 40T disc drive, plinth, Interword and Spellmaster, Intersheet, Wordwise+, Penfriend and much more software on disc, all with original documentation and packaging. Many books (Advanced user guides etc.). £650 ono. Tel. 01-699 6773 eves.

NEW

Any Improvement On Britain's No. 1 Word Processing ROM Must Be A Plus Too

1985 saw the launch of what has become the most popular word processor for the BBC micro, **WORDWISE PLUS**. Produced with the co-operation of Computer Concepts, **WORDWISE PLUS II** offers you all the facilities of **WORDWISE PLUS**, and . . .

Preview in 80 columns

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

Easy file selection

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

Multiple documents

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

Improved search and replace

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

Check saved file

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

Easy to use

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

Extra CTRL keys

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

Supplied on one chip

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

Compatibility

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

Additional embedded command

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

Free utilities

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

Printer buffer

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

REVIEWS

"WORDWISE is good, WORDWISE PLUS is better and WORDWISE PLUS II is better still."
... MICRONET

"WORDWISE PLUS II is an excellent addition to the range of text production tools available to the BBC micro owner. The extensions offered by WW+2 are real icing on the cake".
... NETWORK USER, July 1987

GUARANTEE

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

40%
Off normal

price to
existing

**WORDWISE
PLUS owners
£39.95**



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

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

(07555) 7286

Please allow £1 for postage.

WORDWISE PLUS II £65.95

Upgrade from

WORDWISE PLUS £39.95

Spell Master £51.50

(£50.00 when purchased with
WORDWISE PLUS II).

16K RAM Module £14.95

(No soldering to install)



Business Ads

MIGIT Desktop Image Analyser: Measures distances, perimeter, area, roundness or shapes: Touchpad (6x6cm area) plus disc software for BBC micro, £30. **DIGIT Image Analysis Software** for Grafpad and Bit Pad 2 also available, £70-110. *Institute of Ophthalmology, Judd Street, London, WC1H 9QS. Tel. 01-387 9621x224.*

BEEB-Planner, version 8, CPA program, Time Analyse 250 activities, Calculates project cost, Three calendars, Various reports, Uses Sideways RAM, £39.95. *E.Sheffield, 8 Langdon Close, Camberley, Surrey, GU15 1AQ. S.A.E. for details.*

Data Encryption Programs for sale. Will encode/decode any BBC file. £1 for manual (will credit to purchase) or £12 for programs. Ensures privacy of data on shared Winchester, in offices, in post etc. *T.K.Boyd, Seaford College, Petworth, W. Sussex GU28 0NB.*

"Sharemagic" - Free automatic price updating from Teletext. Probably the best share software pack for a small investor, £25. Free brochure, send S.A.E. to: *Citymagic, Dept BB1, 40 Manor Road, Goldington, Bedford, MK42 9LQ. Tel. (0234)856050/67067.*

EVENTS

Electron and BBC Micro User Show
Old Horticultural Hall,
Westminster
13-15th November 1987

*We will have our usual stand at the
Micro User Show
and we will be pleased to meet any
BEEBUG members there.*

The Fourth High Technology in
Education Exhibition
Barbican Centre, London
20-23rd January 1988

PCW 88 Show
11th Personal Computer World Show
Olympia, London
21-25th September 1988

*We will also have a stand at the Micro User Show
for RISC User,
our new magazine for Archimedes users.
If you are interested in Archimedes please visit our stand,
and see RISC User for yourself.*

UK BULLETIN BOARDS

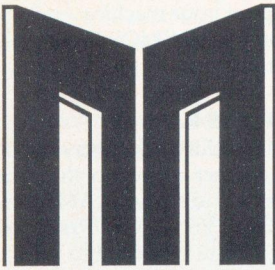
The complete list of Bulletin Boards will be continued in the next issue of BEEBUG.

Please note: the Bloxham NOBB Bulletin Board is no longer in operation.

NEW USER GROUPS (U.K.)

OXON

BBC Micro Wantage Users Group, meets at 8p.m. in the Wantage Civic Hall on the 2nd and 4th Mondays of each month. Subscriptions: adults 50p, under 18s 25p, first-timers free. *Contact Steve Cooper on Wantage (02357)2501*

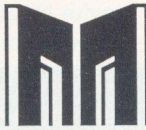


THE MASTER PAGES

Devoted to the
Master Series
Computers

This month, for a change, most of our Master pages are devoted to a single major utility for Master and Compact Users. The coding is longer than we usually publish but these two programs by Philip McKeever, together form a most useful tool, allowing you to examine and edit any RAM area in your system including private RAM and the battery backed CMOS RAM.

In addition, we have a further selection of hints and tips for the Master and Compact.



**MASTER
SERIES
Dynamic
Memory
Editor**

Examine and edit any area of RAM in the Master or Compact, whether 'private' or not, with Philip McKeever's excellent dynamic memory editor.

This program will display in hex and ASCII format all of the available RAM in the Master or Compact; including the shadow RAM (from &3000 to &7FFF), private RAM (from &8000 to &8FFF), filing system RAM (from &C000 to &DFFF), and CMOS RAM.

Any byte of RAM may in addition be edited with ease.

The simple command:

`*MEDIT`

is used to engage the editor from within sideways RAM. An optional parameter may also be supplied, to specify the start address of the RAM to be examined. Thus:

`*MEDIT 3000`

will cause the editor to display RAM at &3000, and so on. If no address parameter is given, the address defaults to the current value of PAGE (normally &E00). You may also abbreviate the name down to *MED (no full stop).

ENTERING THE LISTINGS

Before describing the use of the editor in detail, we will concentrate on getting the software up and running. The program comes in two parts. The first is a ROM header, and the second is the editor proper. The first step is to type in the listings. After they have been separately typed in and saved, the ROM header (listing 1) should be run. If all is well you should then CHAIN listing 2 (the main routine). Once the latter has been assembled, it will automatically locate itself in sideways RAM above the header, and will save the complete sideways ROM image to a file called MediObj.

Now press Ctrl-Break (or use *INIT if you have a Master ROM) to initialise the ROM image. If you now type:

`*ROMS`

or

`*HELP MEDIT`

you will see evidence of the image. To enter the editor simply type:

`*MEDIT`

and you should see a screen similar to that in the illustration. Pressing Escape will take you back to Basic.

To use the editor once the image has been cleared (e.g. by switching off the machine), you should use:

`*SRLOAD MediObj 8000 n`

where n is the destination RAM bank number (normally 4-7), then initialise as before with

with *MEDIT. Remember that altering CMOS RAM may de-configure your machine, and that the Compact's CMOS RAM has a limited life.

```

ADDR      Hex Byte      ASCII AM
ODB0      00 00 00 00 00 00 00 00 00 00 00 00
ODB8      00 00 00 5D 9D 09 4C 97 09 09 09 09
ODC0      5C 98 09 4A 99 09 A1 9D 09 09 09 09
ODC8      09 BD 95 09 8E 9D 09 00 00 00 00 00
ODD0      00 00 00 00 00 00 00 00 00 00 00 00
ODD8      00 00 00 00 00 00 00 00 00 00 00 00
ODE0      00 00 00 00 00 00 00 00 00 00 00 00
ODE8      00 00 00 00 00 00 00 00 00 00 00 00
ODF0      00 00 00 00 00 00 00 00 00 00 00 00
ODF8      00 DA 00 00 00 D9 00 00 00 00 00 00
OE00      ←0D 00 01 0E EE 20 85 20 00 00 00 00
OE08      E5 20 8D 54 43 40 0D 00 00 00 00 00
OE10      02 0B 20 2A 4D 45 44 49 00 00 00 00
OE18      54 0D 00 00 03 0C 20 2A 42 00 00 00
OE20      50 52 49 4E 54 0D FF 03 00 00 00 00
OE28      30 2C 31 30 3A E5 8D 54 00 00 00 00
OE30      43 40 0D FF 03 75 74 68 00 00 00 00
OE38      6F 72 20 20 50 68 69 6C 00 00 00 00
OE40      69 70 20 4D 63 4B 65 65 00 00 00 00
OE48      76 65 72 0D 00 28 1A F4 00 00 00 00
OE50      20 42 65 65 62 75 67 20 00 00 00 00
OE58      20 4F 63 74 6F 62 65 72 00 00 00 00

```

SIDEWAYS RAM

Since the utility is sideways RAM-based, direct access to the other ROMs or sideways RAM is not possible. To get around this problem, you will need to have an image of your target ROM on disc, and *LOAD this into user RAM at say &2000. You can then view the image using *MEDIT 2000. Alternatively, you would use:

```
*SRREAD 2000 +4000 8000 n
(n=4 to 7) to look at sideways
RAM images. Unfortunately
*SRREAD will not work for ROMs
(i.e. for values of n<4 or n>7).
```

Ctrl-Break or *INIT. Alternatively, the file on disc may also be blown directly onto EPROM.

EDITING RAM

The editor has two input modes, hex and ASCII. These may be toggled using the Tab key, and you will see an input mode indicator at the top right of the screen. In hex mode, you should position the cursor at the byte to be edited, and press a key corresponding to the top nibble of the required value. The screen will then freeze waiting for the low nibble to be entered. In ASCII mode, simply press the key corresponding to the desired ASCII code.

PRIVATE RAM

When first engaged, the editor is in normal RAM mode. The Copy key may be used to toggle to private RAM and back again. But note that when you toggle to private RAM, you will automatically switch the display to shadow RAM.

CMOS RAM

To edit CMOS RAM, press Ctrl-C. Pressing Ctrl-C a second time will write back any altered values. If you wish to exit CMOS RAM mode without updating your CMOS RAM, you should press Escape, and re-enter the package

NOTES

The utility uses RAM bank 4 for initial assembly. To change this, adjust the values set in line 130 of listing 1, and line 1010 of listing 2. If you wish the program to respond to *M as well as *MED, then delete line 410 in listing 1.

THE KEYS

Cursor - Cursor keys move the cursor around the display. Shift speeds this up.

Tab - Toggles input mode. The state is indicated by a letter on the top line of the display, A for ASCII, H for hex.

Copy - Toggles all private RAM in or out of the current memory arrangement (1/4K of 1 MHz bus memory, filing system RAM, private RAM, shadow RAM).

Ctrl-C - Engages CMOS RAM editor. A second press of Ctrl-C writes back new values. To avoid this, use Escape.

Ctrl-P - Change target area to PAGE.

The program uses zero page locations &70-&7A. The CMOS editing routine uses &900-&97F as a buffer.

The main routine is located at &9000 in sideways RAM. Using memory below &8FFF would be disastrous because when private RAM is switched in, this part of sideways RAM is also overlaid.

Note that memory between &FE00 and &FEFF is also toggled by the Copy key.

Listing 1

```

10 REM Program Editor Header
20 REM Version B 0.9
30 REM Author Philip McKeever
40 REM Beebug October 1987
50 REM Program subject to copyright
60 :
100 osnewl=&FFE7:oswrch=&FFEE
110 romtype=&82
120 thememoryeditor=&9000
130 rambank=4
140 :
150 FOR pass = 0 TO 7 STEP 3
160 P%=&8000
170 O%=&4000
180 [ OPT pass
190 EQUW 0
200 EQUW 0
210 JMP serviceentry
220 EQUW romtype
230 EQUW copyrightstring MOD 256
240 EQUW 2
250 .title
260 EQUW "Master Memory Editor"
270 EQUW 0
280 .copyrightstring
290 EQUW 0
300 EQUW "(C) BEEBUG 1987"
310 EQUW 0
320 .serviceentry
330 PHA: CMP#4:BEQ editorcode
340 CMP#9:BEQ helpcode:PLA: RTS
350 :
360 .editorcode
370 PHY:PHX:LDX#255:DEY
380 .loop:INX:INY:LDA (&F2),Y
390 AND #&DF:CMP medilabel,X
400 BEQ loop:CMP #13:BNE notmine
410 CPX #3:BMI notmine
420 .ok
430 JMP thememoryeditor
440 .notmine
450 CMP #0:BEQ ok:PLX:PLY
460 PLA:RTS
470 :
480 .helpcode
490 PHY:PHX:JSR printhelp
500 PLX:PLY:PLA:RTS
510 :
520 .printhelp

```

```

530 JSR osnewl:LDX#&FF
540 .helploop:INX:LDA title,X
550 JSR oswrch:BNE helploop
560 JSR osnewl:JSR osnewl:LDX#255
570 .loop
580 INX:LDA command,X:JSR oswrch
590 BNE loop:JSR osnewl:RTS
600 .command
610 EQUW " MEDIT <address>"
620 EQUW 0
630 .medilabel
640 EQUW "MEDIT"
650 EQUW &FE
660 ]
670 NEXTpass
680 :
690 REM Put header into sideways RAM
700 OSCLI "SRWRITE 4000+1000 8000 "+STR
R$(rambank)
710 END

```

* * * * *

Listing 2

```

10 REM Program Memory Editor
20 REM Version B 0.6
30 REM Author Philip McKeever
40 REM Beebug October 1987
50 REM Program subject to copyright
60 :
100 PROCsetvariables
110 PROCmemoryeditor
120 OSCLI"SRWRITE 4000+1000 9000 "+STR
$(rambank)
130 OSCLI"SRSAVE MediObj 8000+4000 "+S
TR$(rambank)
140 END
150 :
1000 DEF PROCsetvariables
1010 rambank=4
1020 osnewl=&FFE7:oswrch=&FFEE
1030 osbyte=&FFF4:osrdch=&FFE0
1040 display=&70 :ptr=&72
1050 screen=&74 :key=&76
1060 flagx=&77 :offset=&78
1070 ammount=&79 :cursorx=&7A
1080 page =&18 :acccon=&FE34
1090 version=INKEY(-256)
1100 IF version = 251 THEN cbytes =0:machine$="no cmos"
1110 IF version = 253 THEN cbytes =50:machine$="&932"
1120 IF version = 245 THEN cbytes =128:machine$="&980"
1130 ENDPROC
1140 :
1150 DEF PROCmemoryeditor
1160 FOR pass= 0 TO 7 STEP 3
1170 P%=&9000:O%=&4000
1180

```



```

1190 [ OPT pass
1200 .thememoryeditor
1210 JSR findstartaddress
1220 JSR init
1230 JSR heading
1240 .repeat
1250 JSR printscreen:JSR printcursor
1260 JSR scankeyboard:CPY #&1B
1270 BNE repeat
1280 :
1290 .exiteditor:LDA #4:LDX #0
1300 JSR osbyte:LDA #31:JSR oswrch
1310 LDA #0:JSR oswrch:LDA #23
1320 JSR oswrch:LDA#23:JSR oswrch
1330 LDA #1:JSR oswrch:JSR oswrch
1340 LDX #7:LDA #0:.vdus:JSR oswrch
1350 DEX:BNE vdus:LDA #8:STA acccon
1360 PLX:PLY:PLA:LDA#0:RTS
1370 :
1380 .findstartaddress:STZ display
1390 STZ display+1:STZ flagx
1400 .spaceloop:LDA (&F2),Y:INY:CMP #32
1410 BEQ spaceloop:CMP #13:BEQ noaddr
1420 .startaddress:JSR ascihex:PHA
1430 LDA flagx:AND #1:BNE badnumber:PLA
1440 CLC:ASL display+1:ROL display
1450 ASL display+1:ROL display
1460 ASL display+1:ROL display
1470 ASL display+1:ROL display
1480 TSB display+1:LDA (&F2),Y:INY
1490 BEQ exitfindaddress:CMP #13
1500 BNE startaddress:.exitfindaddress
1510 LDA display:LDY display+1
1520 STA display+1:STY display:RTS
1530 .noaddr:LDA page:STA display+1
1540 STZ display:RTS
1550 .badnumber:STZ &100:LDX #255
1560 .errorloop:INX:LDA bnumber,X
1570 STA &101,X:BNE errorloop:JMP &100
1580 .bnumber:EQUB 34
1590 EQU$ "BAD NUMBER!":EQUB 0
1600 :
1610 .init:LDA #12:LDX #3:JSR osbyte
1620 LDA #4:LDX #1:JSR osbyte:LDA #22
1630 JSR oswrch:LDA #7:JSR oswrch
1640 LDA #23:JSR oswrch:LDA #1
1650 JSR oswrch:LDA #0:JSR oswrch
1660 LDA #ASC"A":STA &7C26:LDA #ASC "M"
1670 STA &7C27:STZ key:LDA #2:STA flagx
1680 LDA #8:STA acccon:JMP fixpage
1690 :
1700 .heading:LDX #255:.printloop:INX
1710 LDA title,X:STA &7C00,X
1720 STZ &7C28,X:BNE printloop
1730 STZ &7C4E:STZ &7C4F:RTS
1740 .title
1750 EQU$ CHR$(134)+"ADDR "+CHR$(145)+"
///// "+CHR$(134)+"Hex Byte"+CHR$(145)+"
///// "+CHR$(134)+"ASCII":EQUB 0

```

```

1760 :
1770 .printscreen:LDA #&B0:STA offset
1780 LDA #&7C:STA screen+1
1790 LDA #&50:STA screen
1800 LDA #30:JSR oswrch:LDA display
1810 PHA:LDA display+1:PHA
1820 .screenloop:LDA display+1
1830 JSR hexdig:LDA display
1840 JSR hexdig:LDA #ASC" "
1850 JSR illwrch:JSR illwrch:LDY #0
1860 .hexloop:LDA (display),Y
1870 JSR hexdig:LDA #ASC" ":JSR illwrch
1880 INY:CPY #8:BNE hexloop:JSR illwrch
1890 JSR illwrch:LDY #0:.ascloop
1900 LDA (display),Y:CMP #127:BCC tstpc
1910 BRA dodot:.tstpc:CMP #32
1920 BCS writechr:.dodot: CMP #160
1930 BCS writechr:LDA #ASC".":.writechr
1940 JSR illwrch:INY:CPY #8:BNE ascloop
1950 SEC:LDA offset:SBC #8:BEQ endascii
1960 STA offset:LDA display:CLC
1970 ADC #8:STA display:BCS incdisp
1980 JMP screenloop:.incdisp
1990 INC display+1:JMP screenloop
2000 .endascii:PLA:STA display+1
2010 PLA:STA display:RTS
2020 .hexdig:PHA:LSR A:LSR A
2030 LSR A:LSR A:JSR digit:PLA
2040 .digit:AND #15:CMP #10:BCC jst30
2050 ADC #6:.jst30:ADC #&30:JSR illwrch
2060 RTS
2070 :
2080 .printcursor:LDX cursorx
2090 LDA #ASC"]":STA &7DE0,X
2100 INX:INX:INX:LDA #ASC "["
2110 STA &7DE0,X:RTS
2120 :
2130 .scankeyboard:LDA #&81:LDX #0
2140 LDY #0:JSR osbyte:BCS nokeypressed
2150 STX key:PHX:LDA #21:LDX #0
2160 JSR osbyte:LDA #202:LDX #0
2170 LDY #255:JSR osbyte
2180 TXA:BIT #8:BNE shiftpressed
2190 PLX:CPX #138:BEQ downarrow
2200 CPX #139:BEQ uparrow
2210 .notmine:CPX #137:BEQ rightarrow
2220 CPX #136:BEQ leftarrow
2230 CPX #135:BEQ copykey
2240 CPX #3 :BEQ ctrlc
2250 CPX #9 :BEQ tabkey
2260 CPX #16 :BEQ ctrlp
2270 JMP inputdata:.nokeypressed
2280 RTS
2290 .shiftpressed:PLX
2300 CPX #138:BEQ shiftdown
2310 CPX #139:BEQ shiftup
2320 BRA notmine:.downarrow:LDA #8
2330 STA amount:JMP incremstart
2340 .uparrow:LDA #8:STA amount

```



```

2350 JMP decremstart:.rightarrow
2360 JSR incptr:JMP cursorcontrol
2370 .leftarrow:JSR decptr
2380 JMP cursorcontrol:.copykey
2390 JMP changemap:.ctrlc
2400 JMP editcmos:.tabkey
2410 JMP hexascii:.ctrlp
2420 LDA page:STA display+1
2430 STZ display:JMP fixpage
2440 .shiftdown:LDA #80
2450 STA ammount:JMP incremstart
2460 .shiftup:LDA #80:STA ammount
2470 JMP decremstart
2480 :
2490 .incremstart:CLC:LDA display
2500 ADC ammount:STA display
2510 BCS chadisplayi:.ptrup
2520 CLC:LDA ptr:ADC ammount
2530 STA ptr:BCS chaptri:RTS
2540 .chadisplayi:INC display+1
2550 BRA ptrup:chaptri
2560 INC ptr+1:RTS
2570 :
2580 .decremstart:SEC:LDA display
2590 SBC ammount:STA display
2600 BCC chadisplayd:.ptrdown
2610 SEC:LDA ptr:SBC ammount
2620 STA ptr:BCC chaptrd:RTS
2630 :
2640 .chadisplayd:DEC display+1
2650 BRA ptrdown:.chaptrd
2660 DEC ptr+1:RTS
2670 :
2680 .incptr:CLC:LDA ptr
2690 ADC #1:STA ptr:BCS incptrh
2700 .label:INC cursorx:INC cursorx
2710 INC cursorx:JMP cursorcontrol
2720 .incptrh:INC ptr+1:BRA label
2730 :
2740 .decptr:SEC:LDA ptr
2750 SBC #1:STA ptr:BCC decptrh
2760 .label2:DEC cursorx:DEC cursorx
2770 DEC cursorx:JMP cursorcontrol
2780 .decptrh:DEC ptr+1:BRA label2
2790 :
2800 .cursorcontrol:LDA cursorx:CMP
#29
2810 BEQ addcursor:CMP #2:BEQ
delcursor
2820 RTS:.addcursor
2830 CLC:LDA display:ADC #8
2840 STA display:BCS highdisplay
2850 .fiveinflagx:LDA #5:STA cursorx
2860 RTS
2870 .highdisplay:INC display+1
2880 BRA fiveinflagx
2890 :
2900 .delcursor:SEC:LDA display:SBC #8

```

```

2910 STA display:BCC highdis:.label4
2920 LDA #26:STA cursorx:RTS
2930 .highdis:DEC display+1:BRA label4
2940 :
2950 .changemap:LDA &7C26:PHA:LDA
&7C27
2960 PHA:LDA acccon:EOR #&3D:STA
acccon
2970 LDA &F4:EOR #&80:STA &F4:STA
&FE30
2980 PLA:EOR #&1E:STA &7C27:PLA
2990 STA &7C26:JMP heading
3000 :
3010 .editcmos:LDA flagx:EOR #2
3020 STA flagx:AND #2:BEQ readcmos
3030 LDX #255:.writeloop:INX:PHX
3040 LDA &900,X:STZ &900,X:TAY:LDA
#162
3050 JSR osbyte:PLX:CPX #cbytes
3060 BNE writelook:LDX #255
3070 .clearmessage:INX:STZ &7C28,X
3080 CPX #38:BNE clearmessage:RTS
3090 :
3100 .readcmos:LDA #8:STA display+1
3110 LDA #&B0:STA display:LDA #9
3120 STA ptr+1:STZ ptr
3130 JSR fiveinflagx:LDX #255
3140 .messagelook:INX:LDA
cmosheading,X
3150 STA &7C28,X:BNE messagelook
3160 LDX #255:.readloop:INX:PHX
3170 LDA #161:LDY #0:JSR osbyte
3180 TYA:PLX:STA &900,X
3190 CPX #cbytes:BNE readloop:RTS
3200 .cmosheading:EQU$ "CMOS RAM edit
b
etween &900 and "+machine$:EQU$ 0
3210 :
3220 .hexascii:LDA #7:JSR oswrch
3230 NOP:NOP:NOP:LDA &7C26:EOR #9
3240 STA &7C26:RTS
3250 :
3260 .fixpage:LDA display+1:STA ptr+1
3270 LDA display:AND #248:STA display
3280 STA ptr:SEC:LDA display
3290 SBC #80:STA display:BCC lesspage
3300 .label5:JMP fiveinflagx
3310 .lesspage:DEC display+1:BRA
label5
3320 :
3330 .asciihex:CMP #ASC"A":BCS checkf
3340 CMP #ASC"0":BCS check9:.false
3350 LDA #1:TSB flagx:RTS
3360 .checkf:CMP #ASC"F"+1:BCC changeF
3370 BRA false:.check9:CMP #ASC"9"+1
3380 BCC change9:BRA false:.changeF
3390 PHA:LDA #1:TRB flagx:PLA:SEC
3400 SBC #55:RTS

```



```

3410 .change9:PHA:LDA #1:TRB flagx:PLA
3420 SEC:SBC #48:RTS
3430 :
3440 .illwrch:PHA:CMP #&23:BEQ change5F
3450 CMP #&5F:BEQ change60
3460 CMP #&60:BEQ change23
3470 .label6:STA (screen):LDA screen
3480 CLC:ADC #1:STA screen:BCS scrnhi
3490 PLA:RTS:.scrnhi:INC screen+1
3500 PLA:RTS:.change5F
3510 LDA #&5F:BRA label6:.change60
3520 LDA #&60:BRA label6:.change23
3530 LDA #&23:BRA label6
3540 :
3550 .inputdata:LDA #11:LDX #200
3560 JSR osbyte:PHX:LDA &7C26
3570 CMP #ASC"A":BEQ ascii:LDA key

```

```

3580 JSR ascihex:STA key:LDA flagx
3590 AND #1:BNE nodatainput
3600 LDA key:ASL A:ASL A:ASL A:ASL A
3610 STA (ptr):.lowibble:JSR osrdch
3620 CMP #&1B:BEQ gotoescape
3630 JSR ascihex:STA key:LDA flagx
3640 AND #1:BNE lowibble
3650 LDA key:ORA (ptr):BRA lastnib
3660 .ascii:LDA key:.lastnib
3670 STA (ptr):JSR incptr
3680 .nodatainput:LDA #11:PLX
3690 JSR osbyte:RTS
3700 .gotoescape:JMP exiteditor
3710 ]
3720 NEXTPass
3730 ENDPROC

```

B

Master Hints Master Hints Master Hints

SCREEN SAVE TO SIDEWAYS RAM

Mike Taylor

The following short program shows how to save a mode 7 screen to sideways RAM, and load it back in again. The whole screen is saved in a fraction of a second, and the technique can be extended to all sorts of applications:

```

10 MODE 7:PRINT"Some Text":A=INKEY(200)
20 REM Claim bank 4
30 *SRDATA 4
40 REM Save screen
50 *SRWRITE 7C00 7FFF 0
60 CLS:PRINT"Press space":A=GET
70 REM Load screen back in
80 *SRREAD 7C00 7FFF 0
90 END

```

Normally of course, you would put something more interesting on the screen between lines 10 and 20 - maybe with a *LOAD. Watch out for the 0 at the end of lines 50 and 80. This is the pseudo address of the start of RAM bank 4. Pseudo addressing is required once a *SRDATA has been issued.

CMOS RAM ANTI-THEFT DEVICE

Philip Mc Keever

A useful way of marking your Master, Compact or Archimedes for security purposes is to put your post code in the unused portion of the machine's CMOS RAM. To write to the CMOS

RAM, use:

```
*FX 162,N,A
```

where N is the CMOS RAM number, and A the ASCII code of the byte written. To read back the value stored, you need to use OSBYTE 161. For more details on this, see BEEBUG Vol.5 No.10 page 43.

EDITOR COMMENT STRIPPER

David Graham

Here is a command string to be used in the Master Editor to remove the comments from assembler listings, whether these appear on lines by themselves, or at the end of an active line of code. First select f4 (Find String), then enter the following:

```
*space\*\*~$$/$return
```

where *space* means *press the space bar*, and *return* means *press Return*. Each time that a match is found you should press "C" to continue or "R" to replace, as prompted.

DEFINITION LISTER

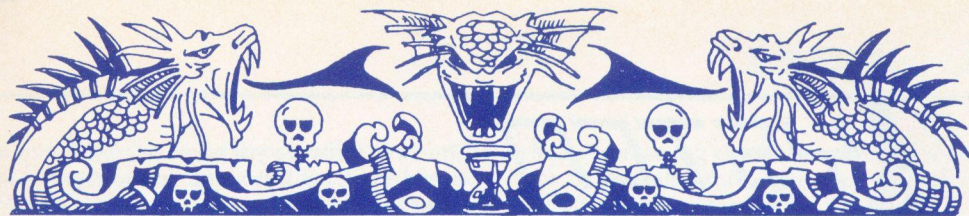
S. Knight

You can use:

```
LIST IF DEF
```

to produce a handy list of all procedures and functions in a program. The list also includes the line number of the start of each definition, and any parameters used. Listed in this way, the display is also uncluttered by lists of calls to each procedure or function.

B



ADVENTURE GAMES ADVENTURE GAMES *By Mitch*

Now where was I? Ah yes, sorry I have not been with you for a while, but as things have been a little quiet around the Dungeon, the Dragon and I went for a stroll over the far mountains. It was only the news that Robico had released a new adventure that brought us scurrying back. This Welsh software house has produced a long line of successful games including my personal favourite, "Enthar Seven".

Reports indicate that as some software houses are no longer writing new adventures for the BBC micro, Robico's products are becoming more widely known and gaining in popularity. In addition to their own adventures, Robico have now decided to market one of last year's releases from the Magus company.

"The Village of Lost Souls" (reviewed in Beebug Vol.5 No.1) was acclaimed by reviewers, but largely ignored by the paying public. Robico have made alterations to the game and added their more powerful command parser.

As mentioned in my previous review, the game centres around a deserted village which has been the site of black magic rites. There are workshops, windmills and a large church to search. Dead bodies and a vampire are to be found and run away from! The game contains a surprising number of puzzles, most of which are quite neat and some of which are pretty awkward.

However, the main problem is the drawing of the map. The game play area is surprisingly large, and the text does not always make it clear what the various exits are. In one notable case the text categorically states that there is only one possible exit when there are in fact two. This means you must test all possible avenues within the village, which can be a very large task. Bearing this in mind the game has many hours of puzzles contained within it. Fun but tricky!

Title	The Hunt - Search for Shauna
Supplier	Robico 3 Fairland Close, Llantrisant, Mid Glamorgan CF7 8QH. Tel. (0443) 227354
Price	Cassette £9.95 inc. VAT Disc £12.95 inc. VAT

Having spent last summer helping Captain Kirk search for Spock, it came as no surprise that the Dragon and I should be called in to search for Shauna. Space is a very large place, and so it would seem are the space stations which circle the planets. Miles of corridors containing generator rooms, robot repair shops and laboratories must be explored before the luckless Shauna can be found. Luckily there are Lugobots to help 'lug' things around. Not so luckily there are Agrobots to help top up your aggravation level. Escaped animals from the experimental level may also come in handy should you manage to persuade them to put their specialized talents at your service.

In addition to the computerized text font, there are approximately 100 graphic pictures to brighten up your screen. Disc versions of the game contain slightly enhanced graphics which are loaded during play. Unlike other Robico games, this adventure was written by an outsider to the company. It appears the game dropped unannounced through the letter box and its quality persuaded the company to market it.

The inclusion of the graphics has obviously meant that the text has been slimmed down, but the result is a pleasant mix. The game allows you to command the various robots and animals you encounter to assist you with some of the trickier problems, but as they have minds of their own you had better remember to keep them apart. The difficulty level of the game does not seem too high, so the whole colourful mixture should appeal to novices and old hands alike.

continued on page 52

SYSTEM DELTA APPLICATIONS

System Delta has established itself as a major tool for database designers, but you need to be a programmer to make the most of this system. Minerva has now released three complementary applications packages to make life easier for System Delta users. Geoff Bains reports.

Product	System Delta Applications: MAIL SHOT REPORTER INTER/VIEW LINK
Supplier	Minerva Systems 69 Sidwell Street, Exeter, Devon. Tel. (0392) 37756
Price	£19.95 each on disc inc.VAT

The System Delta database programming language (see Beebug Vol.5 No.4) from Minerva Systems provides the competent Basic programmer with almost limitless power for database management.

The System Delta ROM contains only the primitives accessible from Basic to create your own database management programs, exactly tailored to your own requirements. The package also includes a simple card index program as a demonstration of the language and as an application utility. That is all very well if you are keen on writing your own software, even if most of the really hard work has already been done for you. However, if what you really want is a series of ready-written applications or maybe the starting point for your own programming adaptations, then the latest releases from Minerva to accompany the System Delta package could be the answer.

The new software packages are primarily to extend the Card Index and to interface it (and other System Delta software) to further programs for the BBC Micro.

MAIL SHOT APPLICATION

The Mail Shot application, as the name suggests, enables System Delta data files to be used for mail shots. There are three sides to this package - a label printer, a wordprocessor link-up, and a mail-merge function.

The **label printer** is little more than a specialised database formatter. Selected fields of each record can be positioned on the label wherever you want. The height and width of the label and the number of labels across the paper can be altered to suit just about any form of continuous stationery labels. Formats are created separately from the data file itself, and several formats can be stored for different labels for the same data - for address labels, file cover labels, or even video cassette labels.

The software is all menu driven with some additional features controlled by (prompted) single key commands. All the System Delta applications software follows a similar format and so it soon becomes familiar.

```

Words:625   Characters free:23747 I
@#@#@#@
@#@#@#@
18th July 1987
Dear Mr. @0,
CONGRATULATIONS!
We at Soapy Suds are pleased to inform
you that you and all the @0 family have
won an all expenses paid holiday to @5.
You have come first in the 'Count The
Soap Flakes' competition which you
entered back in October of last year.
    
```

Standard letter using Mail Shot

The second part of the package produces database output ready for use in a **word processor**. Any word processor which can take ASCII input is compatible. View, Wordwise Plus, and Interword all fit into this category.

Again the fields to be transferred are selected and the format of output defined in the normal System Delta way. The extension to this idea is the true mail-merge facility which interfaces to Wordwise Plus, Interword and View. A standard letter is produced using the word processor with embedded codes inserted where fields from the database are to appear. In the case of View and Interword, the program then produces files from the database ready for use by the word processor's own mail-merge system. For Wordwise Plus users a segment program is provided to merge the data with the letter.

```

System Delta      Full Reporter
Display          F GEOFF

REP'S NAME      LOSS      PROFIT
Leon R. Coll   100.00
Gus B. Greenest Swift
B. Green
Hyde
Sue T. Rait
Lund
Wheeler
C. Weensley
ran. Fletcher Allen
Fletcher
Thatcher
Cindy Island   20.00
446.00

=====
79 25 23 24 77
=====
Press a key

```

Using Full Reporter

The Mail Shot application is an excellent addition to the System Delta language and the Card Index application in particular. It is somewhat expensive for what is after all no more than a Basic program, but it does the job well (largely due to System Delta itself). However, the manual accompanying the package is quite awful. It is well produced but provides no real tuition. It ends up confusing more than it enlightens - a blight on an otherwise good package.

REPORTER

The Reporter application attacks the very heart of database management software, interrogating the database itself. Simply being able to search for particular fields or records, or to browse through a card index type file, is not really using a computer to its best. The Reporter package will compile lists from the database of

all entries meeting conditions specified by the user. It will also total and average numeric fields from each selected entry if required.

There are two Reporter packages on the disc - one the 'simple' reporter which merely lists out the fields of the records selected and displays totals and averages as required. This is command driven, unlike the other System Delta software. The other, 'full' reporter performs much the same tasks but in a more organised way. It takes longer to set up but report formats can be saved on disc and called up again for instant use.

The report is produced in the form of a table. Each column can display any chosen field from the database and conditional or mathematical constraints can also be imposed. The report for a database of a sales rep's performance figures, for example, can be configured to display the reps' names, coverage areas and profit and loss figures for any period. The latter two columns derive from a single database field, displayed in one column if the figure is positive and from the other if negative. Subsets of the whole database (say, all the salesmen covering one area) can be selected and subtotals for each subset displayed automatically.

The manual for this package is much better. It provides a proper tutorial, taking you through an example database report. This is a versatile piece of software which combines with the Card Index program to make a really useful database system.

INTER/VIEW LINK

The final applications package is for users of Intersheet and ViewSheet, and Interchart and Viewplot. Selected data from a database can be transferred with this software from the System Delta format into a format suitable for the other programs. The package will also produce ASCII files for word processors although this is already covered by the Mail Shot package.

Spreadsheets of up to 16 x 255 can be created with each row holding one record and each column the contents of selected fields making up that record. Data can be sorted according to

continued on page 52

1st COURSE

What's the Function of a Function?

The use of procedures in BBC Basic is well documented in the User Guide, and has been covered in magazine articles, but BBC Basic's functions have had a much lower profile. Paul Pibworth redresses the balance, for the benefit of beginners and others alike, by describing some of his own examples.

advantages over chunks of raw code or even subroutines. The first one is quite simple. Procedures let you see your program in nice digestible little chunks. You can plan, write, modify and even test these little chunks very conveniently without having to worry about other parts of the program at the same time.

I began learning how to drive my Beeb the way that lots of other users do, that is, by typing in listings from magazines. As I made progress the question, "what does that bit do?" was gradually replaced with "can I improve it?". With help and advice, especially with regard to the use or otherwise of GOTO, I progressed to the realms of structured programming, involving FOR-NEXT, REPEAT-UNTIL, and the liberal use of procedures, and ultimately functions.

USING PROCEDURES

Procedures are the key to good structure in programming. Without them it is difficult to maintain clarity and readability in a program. The first lines of a well-structured program, after the usual REM statements, could well look like this:

```

100 PROCinitialise
110 REPEAT
120 PROCchoice
130 IF choice=1 PROCdothis
140 IF choice=2 PROCdothat
150 IF choice=3 PROCdosomethingelse
160 PROCagain
170 UNTIL end
180 END
    
```

Although the purpose of this article is primarily to discuss the use of functions, let me first talk a little about procedures. They have several

advantages over chunks of raw code or even subroutines. The first one is quite simple. Procedures let you see your program in nice digestible little chunks. You can plan, write, modify and even test these little chunks very conveniently without having to worry about other parts of the program at the same time.

As you make progress, you learn quite quickly that the same procedures may be called as many times as you like from different parts of the same program, especially if you use them with parameters. Gulp, parameters? Yes, quite easy, just look at the example below which can be used to print various headings in double height text (in mode 7), at different positions on the screen:

```

1000 DEF PROCheading(x,y,A$)
1010 PRINTTAB(x,y);CHR$141;A$
1020 PRINTTAB(x,y+1);CHR$141;A$
1030 ENDPROC
    
```

This could be called, for example, with:

```

150 PROCheading(10,5,"Rhubarb")
    
```

to print the word "Rhubarb" in double height letters at Tab position (10,5). By varying the parameters (the information in brackets), this procedure could be called at different points throughout a program whenever such a heading is required. You could try adding a fourth parameter (called col say) in the range 129 to 135 (this is mode 7 remember) to control the colour of the heading. The procedure would then be defined as:

```

1000 DEF PROCheading(x,y,col,A$)
1010 PRINTTAB(x,y);CHR$col;CHR$141;A$
1020 PRINTTAB(x,y+1);CHR$col;CHR$141;A$
1030 ENDPROC
    
```

A call such as:

```

160 PROCheading(10,2,130,"Greengage")
    
```

would print the word "Greengage" in green (that's the code 130), indented 10 characters on line 2.

Other routines that lend themselves well to being procedure-based are sort routines, file handling routines, print routines and checking routines. Indeed, any self contained task within a program can usually be written as a procedure (the programs published in BEEBUG use this approach extensively). Your procedures can be listed in an exercise book and referred to when needed. Better still, they can be spooled to disc or tape (using *SPOOL) to form a procedure library, and *EXECed into a program when needed. Alternatively, the Part Merge/Save Utility published in BEEBUG Vol.5 No.6 provides a very convenient way of saving and merging individual procedures.

INTRODUCING FUNCTIONS

The User Guide devotes six pages in all to procedures and functions, and includes the phrase "You may well feel confused". I was, especially concerning functions. However, once met and understood, they do tend to grow on you. Well, they did with me, and I want to share my enthusiasm with you.

"What do they do ?", you may ask. They do all that you ask them to do (like procedures), AND they give you an answer. It will probably help if I show you some of the ways that I have used them.

The first example is very simple, and doesn't really need a function to do its job, but it does illustrate the principle. It takes a string t\$ and adds spaces at the front so that the string will be right justified in a field of width w%:

```
1000 DEFFNjustify(t$,w%)=
RIGHT$(STRING$(w%-LENT$, " ") + t$, w%)
```

The 'answer' is specified by whatever follows the "=" sign which terminates the function definition. Note that the "=" sign terminates a function in much the same way that ENDPROC terminates a procedure. Function definitions may be spread over several lines as in the next example which returns TRUE or FALSE depending on whether a person's age is under 12 or not:

This could be called by a line such as:

```
1000 DEF FNage
1010 INPUT "How old are you? " age
1020 IF age<12 THEN =TRUE ELSE =FALSE
```

200 x=FNage

In the function definition, the two '=' signs in line 1020 both serve to exit from the function, AND return a value TRUE (-1) or FALSE (0). This is still quite simple but shows enough to lead on to further developments. Indeed, the function above could become the basis of an all purpose question and answer routine as shown below.

```
1000 DEF FNquestion(A$)
1010 PRINT A$;
1020 REPEAT:R$=GET$
1030 UNTIL R$="Y" OR R$="N"
1040 IF R$="Y" THEN =TRUE ELSE
=FALSE
```

This could be called with lines such as:

```
300 ans=FNquestion("Another go Y/N")
```

I use this type of routine quite frequently in my own programs.

Even more useful is the fact that a function call can be placed wherever the value returned by the function could occur. Thus the function FNquestion could be used in:

```
400 IF FNquestion("Confirm Y/N") THEN
PROCdelete
```

Here the value returned by the function determines whether the statement following THEN (the PROCdelete) is executed or not.

Here is another example based on my own experiences. When entering and storing data in string arrays, I find it is often very convenient to keep all the strings the same length, or to have long strings with different types of information in particular segments of the string, as shown in the example.

```
A$(1)= "JACK          FROST      M 250273"
A$(2)= "POLLY         GONE       F 011072"
A$(3)= "JUSTYN       THYME      M 090542"
```

The individual entries can be padded out to a given length before being assembled together, as follows:

```
1030 INPUT "Forename" f$:f$=FNpad(f$,10)
1040 INPUT "Surname" s$:s$=FNpad(s$,15)
```

using:


```
2000 DEF FNpad(a$,a%)
2010 =LEFT$(a$+STRING$(a%," "),a%)
```

where a% specifies the final length to which a\$ is to be padded (with spaces). The technique used here is to add a string of a% spaces to a\$:

```
a$+STRING$(a%," ")
```

and then remove the leftmost a% characters using LEFT\$. In this way whether the original string was too long, or too short, it would end up exactly the required length (this function will produce an error if a\$ is longer than 127 characters and a%>127 - can you see why?) Notice too, that in this example the function returns a string. A function can be used to return either a number or a string depending solely on what follows the final "=" sign.

Similarly we could write a function FNextract to extract part of a string from character a to b, AND strip off trailing spaces all at the same time. The LOCAL statement acts in an identical way to LOCAL in procedures; it prevents any conflict with the variable t\$ used elsewhere in the program.

The procedure works by stripping off the last space (line 1040) until the rightmost character is not a space (line 1050). Because the REPEAT loop performs the test at the end, we need to make sure that there is at least ONE space at the end of the string t\$ to start with. That is the reason for adding a space in line 1020.

```
1000DEFNFNextract(a$,a,b)
1010LOCAL t$
1020t$=MID$(a$,a,b-a+1)+" "
1030REPEAT
1040t$=LEFT$(t$,LENT$-1)
1050UNTIL RIGHT$(t$,1)<>" "
1060=t$
```

For a long time, as I said earlier, functions were beyond me. But there are so very many occasions when functions can be used that perhaps I now use them excessively. I use them because I think that functions are one of the most useful features in BBC Basic. I hope that by now you will agree with me. If you haven't used them before, then why not give them a try in the next program you write.

ⓑ

ADVENTURE GAMES (Continued from page 47)

Keep an eye out for 'Blazing Star' which is another new game shortly to be released by Robico. Rising out of the dark mesa comes a shining light which is the sheriff's badge on the shirt of Marshall Slade, the meanest gun-toting son-of-a-gun who ever slapped leather. No doubt we can look forward to gun fights,

rattlesnakes and a diet of beans. It's heartening to see that one adventure house is continuing to support the BBC. Why don't you polish up your own masterpiece and send it to them for assessment? Sorry Robico, but it serves you right for calling me back!

ⓑ

SYSTEM DELTA (Continued from page 49)

rules specified from the menu. It is transferred directly into Intersheet, but some extra manipulation is required to satisfy ViewSheet.

In a similar way the selected data can be fed to Interchart and Viewplot for creating graphs. Usefully, data can be automatically grouped together in bands of values for histograms and pie charts. Once the groups have been defined the package creates an 'others' category for the surplus.

The Inter/View Link package is a fairly specialised one. It does its job well enough but I can't help feeling it is a little pricey. The same goes for all three packages. Nearly £20 for each Basic utility is a bit steep. I would recommend spending the £20 on the (equally extortionate) System Delta reference guide, and writing the applications software yourself. However, if that is not possible, these three packages will make your System Delta software even more powerful.

ⓑ

Regression Analysis

In the second of two articles, C.R. Woodings shows how to calculate and display Linear Regression and Correlation for sets of data.

This month's program takes a pair of datafiles created by ViewSheet and the Decoder program (see Streamlining Data Entry on the Beeb, BEEBUG Vol.5 No.10) and allows you to plot one against the other to check for correlations. However, this method of creating data files, though convenient, is not essential for using this month's program (see last month - Time Series Analysis).

The program presented here will automatically choose suitable scales for an instantaneous plot of the data points, will work out the linear regression equations, and will draw the two regression lines. Given a screen dump routine, you can transfer the graph to paper as illustrated in the figure. It also allows you to refine the presentation by allowing your own choice of scales, labels, and on-screen notes.

REGRESSION AND CORRELATION EXPLAINED

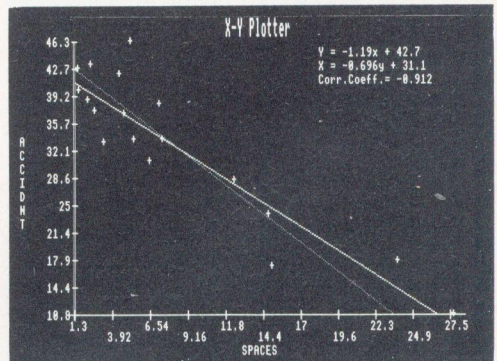
There are many situations in experimental work or in the analysis of data where there is a need to see if changes in one of the parameters measured are causing changes in another. The approach used is to plot one set of data (the X-axis variable) against another (the Y-axis variable) and to see if there is a linear relationship between the two. More often than not, the points plotted do not fall along a nice straight line, but form a rather indistinct cluster of points which leave the experimenter unsure as to whether or not changes in X are causing changes in Y or vice versa.

In such situations the statistician would calculate the correlation coefficient and the regression equations of the distribution. A

coefficient of 1 (or -1) indicates the maximum degree of correlation (a perfect fit), while a coefficient of 0 indicates no correlation at all. Armed with these measures it is possible to comment on the likelihood of a cause and effect relationship, and also to estimate values of X from unmeasured values of Y. The calculations are to say the least tiresome, but now with the aid of the BBC micro and the program listed here you can do it all in seconds and have a hard copy of your efforts into the bargain!

USING THE PROGRAM

The program should first be typed in and saved to disc or tape. We can then proceed as follows. Let us assume you have entered the data represented in the figure into two columns of a ViewSheet spreadsheet and have stored it on disc using April's Decoder program. You have used the Viewsheet "Column Heading" function key to head the columns ACCIDNT and SPACES respectively, and so your data is now in files D.ACCIDNT and D.SPACES. These files can also be created directly without using ViewSheet (as explained last month), but you lose the advantage of ViewSheet's editing facilities. You should put these files on the same disc as the program listed in this article and run the program. ADFS users should create a directory \$D and put the data files into this.



Example of linear regression

When you run the program the first screen which you will see asks you to choose datafiles from the list displayed (list not displayed with cassette files). You type in SPACES as the X-axis data filename and ACCIDNT as the Y-axis data

filename (if you make a mistake, an error message tells you and allows a re-try). You now see the menu screen, and if you select option 2, "Display X-Y plot", you will immediately get the graph of SPACES versus ACCIDENT, as shown in the illustration. The regression lines, the regression equations, and the correlation coefficient will all be displayed, and the graph title will be the default "X-Y Plotter". Similarly, the axes will be labelled with the defaults "SPACES" and "ACCIDENT" arising from the datafile names. Finally, the X and Y axis scales will start and finish with the lowest and highest values of each variable calculated. This may mean that the scale labels will involve awkward numbers with decimal points, and some of the points plotted may be on the axes themselves.

Nevertheless, you will have seen the data and statistical information in record time, and in many instances this will be enough to help you conclude that further attention to these variables is unnecessary. However, let's assume now that you like what you see, and want to tidy up the display. Press Escape to get back to the menu and select option 3, "Axis labels and print out."

The first screen shows you the current title, and prompts for a new one. Continuing with the example in the figure you would enter "Road Accidents to Children" to set a new title. The next two screens get the axis labels you prefer (i.e. "CHILD ACCIDENTS" for the Y-axis), and the final four screens get the start and finish values for the X and Y axis scales.

In the modified version the value 0 could be used for the X and Y axis origins, and the values 30 and 50 respectively for the X and Y axis maxima. These would give scale numbering without decimal points because they give integer values when split into 10 parts. After entering the last value the tidied graph is drawn automatically.

If you now want to add comments to any part of the graph, you can use the cursor keys to position the cursor and simply type away, using Delete to correct any mistakes. Remember though, that the letters you type will

overwrite any of the graphics on the screen if you are not careful. Most important - when you are ready to print it out you must press Tab not Escape in order to save the screen you've prepared for printing. There are, for obvious reasons, no on-screen instructions to help you at this point, so remember Tab to keep your editing, Escape to lose it.

Pressing Tab saves the current screen, and you are then offered the "print" option. If you have a suitable machine code dump such as those published in BEEBUG (see Ref.1 below) loaded at &A00, you can answer affirmatively and see the fruits of your labours transferred to paper. The edited graph will be re-displayed, and may be further edited and dumped until you press Escape. When you have finished, the file \$.SCREEN on disc will contain the last screen saved.

PROGRAM NOTES

Many REMs and long variable names have been used to help you follow the programming. The program will run as listed on the B+ 128, Master, Compact, and Model B with shadow RAM (although the screen dump to printer feature will not function if shadow RAM is enabled because of the address used in the *SAVE command in line 2840). Model B users without shadow RAM must omit all the REMs and the blank lines used to highlight the procedures. Furthermore, it will be necessary to reduce the size of the data arrays from 99 in line 130. Alternatively, load and run the program at a value of PAGE lower than &1900 (&1200 should be suitable for DFS users). You can also change mode=0 to mode=4 at line 110 to save space, but the resulting display will not be as neat.

PROCanal does the statistical analysis, calculating the regression equations ($Y=mX+c$), the correlation coefficient (r), and fits the best straight lines to the data using the method of least squares. A full and quite readable exposition of the method is given in reference 2 below.

REFERENCES

(1) The print routine expects to find a machine code screen dump resident at page &A00 in

memory (CALL &A00 in line 2220). A routine suitable for Epson printers was published in BEEBUG Vol.1 No.9, and this has been used in the development of this program. Commercially available printer dump ROMs also include suitable routines, and if you have one, line 2220 can be altered to call it.

(2) "Facts from Figures" (chapter 16) by M.J.Moroney, published by Penguin Books.

Please note that there was an error in line 100 in the Decoder program listed in Vol.5 No.10 page 25. This should read:

100 DIM os 40:ON ERROR GOTO 920
and not as listed.

```

10 REM Program LINREG
20 REM Version B1.8
30 REM Author C.R.Woodings
40 REM BEEBUG October 1987
50 REM Program subject to Copyright
60 :
100 ONERRORGOTO290
110 mode=0
120 *FX200,1
130 DIMxdata(99),ydata(99),fnn$(1),buf
f 80,os 40
140 c1$=CHR$134:flashon$=CHR$136:flash
off$=CHR$137
150 MODE7
160 REM Find current filing system
170 A%=0:Y%=0:fs=USR(&FFDA) AND &FF
180 DFS=(fs=4):ADFS=(fs=8):CFS=(fs=1)
190 IF DFS PROCoscli("DIR D")
200 IF ADFS PROCoscli("DIR $.D")
210 K%=0:xmin=0:ymin=0:fnn$(0)="NO DAT
A"
220 title$="X-Y Plotter"
230 PROctop:PROcdload:PROcanal
240 *FX200,0
250 MODE7:REPEAT:O%=FNmenu:UNTIL O%>0
AND O%<5
260 ONO%GOTO210,280,270,310
270 PROCprams
280 MODE mode:PROCplot:GOTO250
290 IF ERR>189:CLS:PRINT"Data File lo
ading problem"
300 IF ERR<>17:PRINTTAB(0,23):REPORT:P
ROccont:GOTO230 ELSE GOTO250
310 VDU26,12:@%=&090A:PRINT'"BYE":*FX
4,0
320 IF NOT CFS PROCoscli("DIR $")
330 END
340 :
1000 DEFPROCdhlht(row,wds$,clr):LOCAL i
,c

```

```

1010 c=INT((40-LEN(wds$))/2)-2
1020 FORi=row TO row+1:PRINTTAB(c,i);CH
R$141;CHR$clr;wds$
1030 NEXT
1040 ENDPROC
1050 :
1060 DEFFNyn(x,y,A$)
1070 LOCALans:PRINTTAB(x,y);A$;" (Y/N)
? ";
1080 REPEAT:ans=(GETAND&DF):UNTILans=&5
9 OR ans=&4E:PRINTCHR$ans
1090 =(CHR$ans="Y")
1100 :
1110 DEFPROCcont
1120 PRINT flashon$"Press any key";flas
hoff$
1130 REPEATUNTILGET
1140 ENDPROC
1150 :
1160 DEFFNinput(len,loASC,hiASC)
1170 LOCALK%,Z%:K%=0:In$=""
1180 PRINTSTRING$(len,".");STRING$(len,
CHR$8);*FX15,1
1190 REPEAT:Z%=GET
1200 IFZ%=127 AND K%>0 THEN K%=K%-1:In$
=LEFT$(In$,K%):VDUZ%,46,8
1210 IFZ%>loASC AND K%<len AND Z%<hiASC
THEN K%=K%+1:In$=In$+CHR$Z%:VDUZ%
1220 UNTILZ%=13
1230 =In$
1240 :
1250 DEFPROCanal
1260 REM calculate the data ranges
1270 xsum=0:ysum=0:xmax=-1E37:ymin=-1E3
7
1280 xmin=1E37:ymin=1E37:ysum=0:x2sum=
0:y2sum=0
1290 FORN%=1TOK%
1300 IFxdata(N%)>xmax xmax=xdata(N%)
1310 IFydata(N%)>ymax ymax=ydata(N%)
1320 IFxdata(N%)<xmin xmin=xdata(N%)
1330 IFydata(N%)<ymin ymin=ydata(N%)
1340 REM calculate sums, sums of produc
ts and squares
1350 xsum=xsum+xdata(N%)
1360 ysum=ysum+ydata(N%)
1370 xysum=xysum+xdata(N%)*ydata(N%)
1380 x2sum=x2sum+xdata(N%)^2
1390 y2sum=y2sum+ydata(N%)^2
1400 NEXT
1410 REM calculate means and standard d
eviations
1420 xmean=xsum/K%
1430 ymean=ysum/K%
1440 stdevx=SQR(x2sum/K%-xmean^2)
1450 stdevy=SQR(y2sum/K%-ymean^2)
1460 REM calculate correlation coeffici
ent (r) and constants m and c

```



```

1470 r=(xysum/K%-xmean*ymean)/(stdevx*s
tdevy)
1480 m=r*(stdevy/stdevx)
1490 c=ymean-m*xmean
1500 ml=r*(stdevx/stdevy)
1510 cl=xmean-ml*ymean
1520 ENDPROC
1530 :
1540 DEFPROCplot
1550 REM set up screen display
1560 VDU26,19,1,0;0;19,0,7;0;:CLS:@%=&3
07
1570 Ysc=800/(ymax-ymin):Xsc=1000/(xmax
-xmin):REM screen scale factors
1580 MOVE4,4:DRAW4,1019:DRAW1275,1019:D
RAW1275,4:DRAW4,4
1590 REM draw the axes and plot the poi
nts
1600 VDU29,160;140;5
1610 MOVE-4,-16:DRAW-4,800:MOVE0,800:DR
AW0,-16:MOVE-20,0:DRAW1040,0
1620 FORN%=1TOK%
1630 MOVE(xdata(N%)-xmin)*Xsc,(ydata(N%
)-ymin)*Ysc+16
1640 PRINT"+"
1650 NEXT
1660 PROClabel
1670 REM draw the regression lines
1680 VDU24,0;0;1000;800;:REM window to
keep lines in graph area
1690 MOVE0,((xmin*m+c)-ymin)*Ysc
1700 DRAW(xmax-xmin)*Xsc,((xmax*m+c)-ym
in)*Ysc
1710 MOVE((ymin*ml+cl)-xmin)*Xsc,0
1720 PLOT21,((ymax*ml+cl)-xmin)*Xsc,(ym
ax-ymin)*Ysc:VDU26
1730 PROCnotes:REM allow on-screen note
s to be added after TAB is pressed
1740 VDU22,7:PROCTop
1750 IF FNyn(1,7,"Copy the graph to a P
rinter") PROCprint
1760 PROCreload:GOTO1730:REM Esc needed
to exit loop
1770 ENDPROC
1780 :
1790 DEFPROClabel:LOCAL b,N%
1800 IF mode=4 b=224 ELSE b=124
1810 X%=0:pos=16
1820 REM mark and scale the X and Y axe
s
1830 FORN%=0TO10
1840 X%=N%*100:Y%=N%*80+28
1850 MOVEX%-8,12:PRINT"| "
1860 MOVEX%-16,-36+pos:PRINT;xmin+N%*(x
max-xmin)/10:pos=-pos
1870 MOVE-8,Y%:PRINT" " :MOVE-b,Y%-14:PR
INTymin+N%*(ymax-ymin)/10
1880 NEXT
1890 REM add the axis labels

```

```

1900 D=LEN(xlabel$):E=INT(61-D)/2
1910 MOVEE*16,-90:PRINTxlabel$
1920 D=LEN(ylabel$):E=INT(21-D)/2
1930 FORN%=1TOD
1940 MOVE-148,(840-E*40)-N%*40
1950 PRINTMID$(ylabel$,N%,1)
1960 NEXT
1970 REM print the regression
equations in the least used top corner
1980 IF mode=0 THEN 2010
1990 IF r<0 b=500 ELSE b=50
2000 GOTO 2020
2010 IF r<0 b=650 ELSE b=50
2020 MOVE b,780:PRINT"Y = ";m;"x + ";
2030 IFc<0 VDU8,8:PRINT;c ELSE PRINT;c
2040 MOVE b,740:PRINT"X = ";ml;"y + ";
2050 IFcl<0 VDU8,8:PRINT;c1 ELSE
PRINT;c1
2060 MOVE b,700:PRINT"Corr.Coeff.= ";r
2070 REM add the graph title in double
height letters
2080 D=LEN(title$):E=INT(81-D)/2
2090 MOVEE*16-160,868:PROCbig(title$)
2100 ENDPROC
2110 :
2120 DEFPROCTop
2130 VDU26:CLS:CLOSE#0
2140 PROCdblht(1,"X-Y Plotter:
"+fnm$(0),131)
2150 PRINTTAB(0,3)CHR$146STRING$
(39,"£")
2160 PRINTTAB(0,22)CHR$146STRING$
(39,"£")
2170 VDU28,0,23,39,4
2180 ENDPROC
2190 :
2200 DEFPROCprint
2210 PROCreload
2220 VDU26,2:CALL&A00:VDU3:REM expects
a screen dump in page &A
2230 ENDPROC
2240 :
2250 DEFPROCdload
2260 REM expects to find VIEWSHEET
data files converted by DECODER program
2270 CLS
2280 IF NOT CFS PRINT"Choose datafiles
from list: -"
2290 IF DFS PROCoscli("INFO *")
2300 IF ADFS PROCoscli("$.D")
2310 FOR X=0 TO 1
2320 REPEAT
2330 PRINT"Enter "CHR$(88+X)"-Axis
Filename: "
2340 fnm$(X)=FNinput(12,31,127)
2350 UNTIL FNfileopen(X)
2360 IF X=0 INPUT#D,kX:FOR N%=1 TO
kX:I
NPUT#D,xdata(N%):NEXT

```



```

2350 UNTIL FNfileopen(X)
2360 IF X=0 INPUT#D,kX:FOR N%=1 TO kX:I
NPUT#D,xdata(N%):NEXT
2370 IF X=1 INPUT#D,kY:FOR N%=1 TO kY:I
NPUT#D,ydata(N%):NEXT
2380 CLOSE#D
2390 NEXT
2400 xlabel$=fnn$(0):ylabel$=fnn$(1)
2410 REM check that files chosen have t
he same number of datapoints
2420 IF kX<>kY THEN PRINT"" You have
";kX" X points and ";kY" Y points":IF NO
T FNyn(5,VPOS+1,"Do you want to proceed"
) THEN *fx125
2430 IF kX>kY THEN K%=kY ELSE K%=kX
2440 CLS
2450 ENDPROC
2460 :
2470 DEFfNmenu
2480 PROCtop
2490 VDU28,10,20,38,9:PRINT"1"cl$"Load
New Datafiles""2"cl$"Display X-Y Plot"
"3"cl$"Axis Labels/Printout""4"cl$"Quit
":VDU26
2500 PRINTTAB(7,20)"Enter Number of Pro
cedure ";:VDU8,8,8
2510 temp$=GET$:PRINTtemp$
2520 VDU28,0,24,39,4
2530 =VALtemp$
2540 :
2550 DEFPROCprams
2560 REM allows you to choose scales an
d labels
2570 ret$="<RETURN> accepts, or type in
a new one:"
2580 CLS:PRINTTAB(11,8)"Graph Title is:
-""title$'ret$
2590 temp$=FNinput(77,31,127):IF temp$<
>"" title$=temp$
2600 CLS:PRINTTAB(11,8)"X-axis label is
now:""xlabel$'ret$
2610 temp$=FNinput(60,31,127):IF temp$<
>"" xlabel$=temp$
2620 CLS:PRINTTAB(11,8)"Y-axis label is
now:""ylabel$'ret$
2630 temp$=FNinput(20,31,127):IF temp$<
>"" ylabel$=temp$
2640 CLS:PRINTTAB(11,8)"X-axis origin i
s at ";xmin'ret$
2650 temp$=FNinput(10,39,58):IF temp$<>
"" xmin=VALtemp$
2660 CLS:PRINTTAB(11,8)"Y-axis origin i
s at ";ymin'ret$
2670 temp$=FNinput(10,39,58):IF temp$<>
"" ymin=VALtemp$

```

```

2680 CLS:PRINTTAB(11,8)"X-axis maximum
is at ";xmax'ret$
2690 temp$=FNinput(10,39,58):IF temp$<>
"" xmax=VALtemp$
2700 CLS:PRINTTAB(11,8)"Y-axis maximum
is at ";ymax'ret$
2710 temp$=FNinput(10,39,58):IF temp$<>
"" ymax=VALtemp$
2720 ENDPROC
2730 :
2740 DEFPROCnotes
2750 REM allows on-screen notes to be w
ritten prior to printing
2760 *FX4,1
2770 VDU4:REPEAT:A%=GET
2780 IF (A%=138 AND VPOS<31) VDU10
2790 IF (A%=139 AND VPOS>0) VDU11
2800 IF (A%=136 AND POS>0) VDU8
2810 IF (A%=137 AND POS<79) VDU9
2820 IF A%>31 AND A%<128 PRINTCHR$(A%);
2830 UNTIL A%=9:REM press TAB to exit
2840 *SAVE $.SCREEN 3000 7FFF
2850 *FX4,0
2860 ENDPROC
2870 :
2880 DEFPROCreload
2890 VDU22,0,19,1,0;0;19,0,7;0;
2900 *LOAD $.SCREEN
2910 ENDPROC
2920 :
2930 DEFPROCbig(chars$)
2940 FOR count%=1 TO LENchars$
2950 ?buff=ASC MID$(chars$,count%,1)
2960 X%=buff:Y%=buff DIV 256:A%=&A
2970 CALL &FFF1
2980 VDU 23,224,buff?1,buff!1;buff!2;bu
ff!3;buff?4
2990 VDU 23,225,buff?5,buff!5;buff!6;bu
ff!7;buff?8
3000 VDU 224,8,10,225,11
3010 NEXT
3020 ENDPROC
3030 :
3040 DEFfNfileopen(c):LOCAL t$:t$=LEFT$(
fnn$(c),2)
3050 IF t$="D." OR t$="d." fnn$(c)=MID$(
fnn$(c),3)
3060 D=OPENUP(fnn$(c))/
3070 IF D=0 PRINT TAB(23)"File not
found"
3080 =(D>0)
3090 :
3100 DEFPROCcoscli($os)
3110 X%=os:Y%=os DIV 256:CALL&FFF7
3120 ENDPROC

```

B

Pearl Data Logger

David Peckett, author of our Oscilloscope and Chart Recorder programs, has been testing some real hardware, the Pearl 8000 Data logger.

Product Pearl 8000 Data Logger
Supplier Electronic Innovations Ltd,
 (formerly Masodax Ltd)
 84 Kings Road,
 Berkhamsted,
 Herts HP4 3BP.
 Tel. (04427) 3146

Price £172.50 inc. VAT

Pearl 8000 data logger from Electronic Innovations Ltd. is a product which may interest anyone who uses their BEEB as a laboratory instrument. It is a neatly-made device which could help schools, engineers or anyone else who wants to use the analogue port to measure changing values. Its design does much to overcome the limited accuracy of the BBC's ADC analogue port.

For the readers unfamiliar with this subject, a data logger is a device which can be used to monitor one or more signals. It samples each one at defined rates for a defined time and stores the results for later analysis.

Strictly, the Pearl (Parallel Entry Auto Ranging Logger) 8000 is not a data logger. However, when plugged into the Beeb, the whole system does become one, so the distinction is subtle.

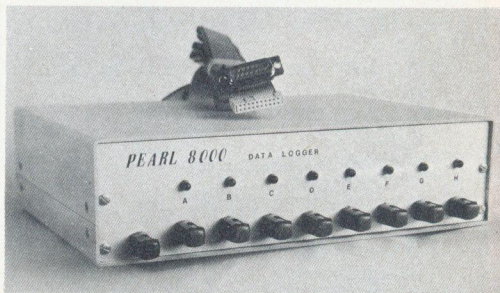
THE PEARL 8000 DESCRIBED

The Pearl 8000 is built into a metal box, which measures approximately 10"x7"x2.5", and finished in BBC cream. It is connected to the computer's analogue and user ports by 2 ribbon cables. The front panel has 8 spring-loaded connectors, one for each channel which can be monitored, and a 9th connector as an earth. There are also 8 LEDs to show which channel is

being read at any time. A 25-pin D socket at the back of the box provides a way of extending the channels and the device's control signals to a distant point. The standard of construction is good and shows careful attention to detail in its component and track layout.

The Pearl 8000 is controlled via the user port, by software supplied with the device, and it multiplexes up to 8 separate channels to ADC channel 1. A useful feature is that it is not limited by the usual 0-1.8V input range of the ADC. A key feature of the device is that it incorporates a software controlled attenuator which will reduce signals of up to 250V to suitable levels for the ADC.

The system can therefore be used to measure voltages (DC only) with absolute values from 0 to 250V. Since it also detects a reversed polarity, its full range is actually -250V to +250V. Although high voltages are reduced to measurable levels, small values are not amplified, so that although it will read down to a millivolt or so, the precision and accuracy of the low readings is poor (a precision amplifier is planned for the future).



The Pearl 8000 also overcomes the limited accuracy of the Beeb's standard ADC. The 7002 chip used in the computer relies on having a reference voltage supplied to it against which it measures all incoming voltages. The standard system has a nominal 1.8V reference, but it is fairly crude and therefore inaccurate. The Pearl 8000, however, provides a dedicated reference, using a precision 1.225V diode.

This approach greatly increases the system's accuracy, although it does nothing for the ADC's basic 10 bit (at best) precision. If you're

not sure of the difference between accuracy and precision, suppose you measure a 0.5V signal and get a reading of 1.3675234V. You have measured with great precision, but the accuracy is lousy. The point is that the Pearl 8000, for all its vast cost, in fact contains no analogue to digital converter. It uses the Beeb's own, and just supplies it with a stable reference source.

THE PEARL 8000 IN USE

A specialist device like the Pearl 8000 is almost certainly going to be used by people who understand what it can do for them and what they want to do with it. The instructions reflect this approach but, even so, the draft manual I reviewed left much to be desired. The manufacturers tell me that the manual will be improved but that they assume that users will know what they are doing. I fear that this might be a little optimistic!

For example, there was a serious lack of general-purpose software. The supplied programs were tricky to follow and would not be easy to adapt for purposes other than demonstration. Some of the programming techniques were individualistic to say the least, and all the software was optimised for the demos. It is unlikely, for instance, that any "real" programs would want to poke the readings from the 8 ports directly into the 8 integer variables A%-H%. There were not even sufficient guidelines on what should be considered when writing new software for the device.

That said, how did the system behave? Using the demonstration programs, very well. It was simple to use and acted as advertised. The system was accurate and, especially when taking several readings from each channel to reduce the effects of noise, the differences between the channels were insignificant. For example, applying a precision 10.0V to each channel, and averaging 10 readings from each, gave values between 9.991V and 10.002V. Not surprisingly, the variation was a little wider when taking only one sample from each channel: 9.988V to 10.022V. Still pretty good, though.

While the system is thus impressively accurate, I did notice that the supplied software incorporated individual calibration figures for the Pearl 8000. Given the system's design, individual calibration would be hard to avoid, but could concern anybody using more than one. The suppliers claim, however, that individual units should not differ by more than 0.5%.

Who might use the Pearl 8000? Virtually anyone who might want to use a BBC to monitor and record DC voltages. The voltages themselves could represent almost any physical item - pressure, temperature, speed, pH, the list is almost endless. It allows up to 8 voltages (11 if you want to fiddle a bit) to be measured, compared with the 4 available from the standard ADC port and, very importantly, the accuracy is much improved

Probably, the system would be of most use in schools or colleges, but I can imagine some hobbyists putting it to good use too. It could even be of value in small laboratories or in simple test equipment. Couple it with my recent chart recorder program (BEEBUG Vol.5 No.10) and you would have an even more useful tool.

CONCLUSION

The Pearl 8000 is a nicely-made and effective piece of hardware which adds significantly to the Beeb's capabilities. It gives you 8 input channels when the Beeb only provides 4, and it increases the accuracy of measurement considerably, as well as providing a voltage attenuator. But for what it does, it is certainly a little pricey.

Like too many add-ons, it is let down by its software and documentation. It would not be difficult for Electronic Innovations to improve them but, unless the company does so, it will remain unnecessarily difficult to exploit the system properly.

NOTE: The manual and software available to our reviewer are early versions. Production versions will in both cases be much improved according to the manufacturer. **B**



Bewitched, bothered and bewildered is how you are likely to feel after playing Mat Eastmond's lively game of Spooks 'n' Spirits. You really need to keep your wits about you to dodge the demonic thunderbolts as you struggle ever onwards in your quest.

Spooks 'n' Spirits is the name of this month's classic game. You play the part of the daring hero Des on his dash home from the graveyard for tea. But Des lives in no ordinary world and luckily has four lives.

Use the daggers to kill the frequent nasties, if you can't find any other way of avoiding them, but watch out for the thunderbolts that rain down on you in retribution. At the end of each level you will find the Keykeeper, the guardian to the next stage. Like you he has multiple lives, and you will need to kill him three times over to make sure he's really dead. Watch out too for the spooks and zombies and other unmentionable hazards.

The program comes in two parts which you should type in and save as SPOOK1 and SPOOK2 respectively. To get started simply CHAIN"SPOOK1", which sets up all the character definitions and automatically chains in SPOOK2. The main keys to use are Z and X for left and right, + and > for up and down, Shift to throw a dagger and Return to jump.

If you want an excuse to stay up all hours of the night then this is it. We take no responsibility for the consequences. You have been warned.

PROGRAM NOTES

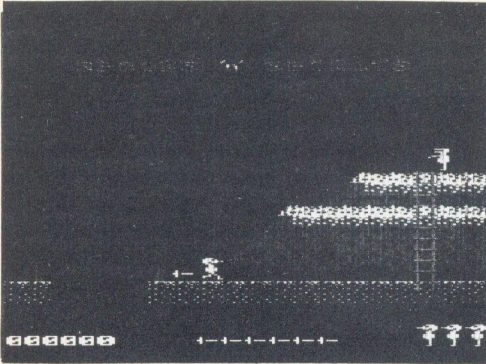
The VDU statements in the first program define the many user defined characters employed in the program, some for the screen designs, and some for the various characters in the plot.

The odd looking data statements, which appear towards the end of the second program, specify, in a highly condensed format, the screens used by the game. These are decoded internally by the program. Any typing errors in either of these two sections are likely to result in corrupted screen displays.

```

10 REM Program Spook1
20 REM Version B1.1
30 REM Author Mat Eastmond
40 REM BEEBUG October 1987
50 REM Program subject to copyright
60 :
100 ON ERROR GOTO1780
110 MODE2:VDU23,1,0;0;0;0;
120 VDU19,15,1;0;19,9,6;0;
130 VDU19,14,5;0;19,8,6;0;
140 VDU19,13,0;0;19,11,6;0;
150 VDU19,2,2;0;19,4,6;0;
160 VDU19,12,4;0;19,10,6;0;
170 ?&36D=&B:PROCVdus
180 ENVELOPE1,0,0,0,0,0,0,100,-100,-
100,-100,100,0
190 ENVELOPE2,1,-10,-10,-10,11,1,1,0,0
,0,-1,120,120
200 ENVELOPE3,1,0,0,0,0,0,126,-5,0,-
10,126,110
210 ENVELOPE4,133,8,4,8,3,1,1,126,0,0,
-10,126,0
220 ENVELOPE5,1,-1,0,0,200,0,0,100,0,0
,-10,100,0
230 ENVELOPE6,129,-1,-1,-1,30,30,30,10
0,-1,-1,-1,100,0
240 ENVELOPE7,2,1,-1,1,2,4,2,1,0,0,-1,
100,0
250 ENVELOPE8,1,1,1,1,5,5,5,-5,-1,-1,-
1,120,120
260 ENVELOPE9,2,+1,-1,+1,2,4,2,10,-1,0
,-1,106,70
270 ENVELOPE10,5,0,-1,0,7,255,0,126,-4
,0,-6,106,40
280 ?&34F=24: ?&30A=25:PROCLogo(9999)
290 ?&34F=24: ?&30A=25:PROCsnns
300 IF PAGE>&1100 THEN PAGE=&1100
310 CHAIN"SPOOK2"
320 :

```

Spooks 'n' Spirits - The Graveyard

```

1000 DEF PROCsns
1010 RESTORE 1770:VDU 5
1020 FORT%=184 TO 1024STEP 56
1030 READ A:GCOL 0,5:MOVE T%,932:VDU A:
MOVE T%+4,936:IF T%<576 GCOL0,1:VDU A E
LSE GCOL0,2:VDU A
1040 NEXT:VDU 4
1050 ENDPROC
1060 :
1070 DEF PROCvdus
1080 VDU23,224,112,232,254,216,96,48,48
,120
1090 VDU23,225,120,120,48,16,16,16,16,2
4
1100 VDU23,226,120,120,48,16,40,228,132
,6
1110 VDU23,227,14,23,127,27,6,12,12,30
1120 VDU23,228,30,30,12,4,4,4,4,12
1130 VDU23,229,30,30,12,8,20,39,33,96
1140 VDU23,230,157,140,33,51,153,221,15
3,60
1150 VDU23,231,199,141,155,138,131,103,
241,227
1160 VDU23,232,255,61,25,1,127,251,223,
247
1170 VDU23,233,127,253,207,79,199,247,1
26,63
1180 VDU23,234,183,253,95,75,6,4,0,0
1190 VDU23,235,64,218,125,247,253,95,25
1,255
1200 VDU23,236,66,255,0,66,66,66,66,66
1210 VDU23,237,247,222,255,123,255,222,
255,119
1220 VDU23,238,16,144,146,174,81,146,14
6,146
1230 VDU23,239,128,192,96,224,160,192,1
12,208
1240 VDU23,240,240,80,248,232,184,236,1
26,235

```

```

1250 VDU23,241,24,60,60,60,153,153,126,
60
1260 VDU23,242,60,60,24,36,36,36,36,102
1270 VDU23,243,60,60,60,24,165,66,0,0
1280 VDU23,244,124,96,124,48,48,62,113,
120
1290 VDU23,245,62,6,62,12,12,126,142,30
1300 VDU23,246,238,238,238,0,187,187,18
7,0
1310 VDU23,247,40,40,16,16,8,170,169,19
7
1320 VDU23,248,137,73,73,50,28,24,24,24
1330 VDU23,249,24,24,56,56,60,60,60,60
1340 VDU23,250,1,3,3,7,11,14,11,31
1350 VDU23,251,31,29,55,63,95,125,127,2
19
1360 VDU23,252,60,60,66,90,74,126,126,1
26
1370 VDU23,253,24,16,24,255,255,24,24,1
6
1380 VDU23,254,24,16,24,16,24,24,40,219
1390 VDU23,255,16,56,68,16,0,16,0,16
1400 VDU23,192,3,5,3,1,28,127,207,231
1410 VDU23,193,192,160,192,128,56,254,2
43,231
1420 VDU23,194,131,129,7,12,6,12,60,60
1430 VDU23,195,193,129,224,48,32,48,60,
60
1440 VDU23,196,0,64,64,222,64,64,0,0
1450 VDU23,197,2,6,12,24,12,12,24,32
1460 VDU23,198,24,63,60,0,126,213,171,1
26
1470 VDU23,199,24,252,60,0,126,213,171,
126
1480 VDU23,200,34,119,243,255,255,255,2
55,255
1490 VDU23,201,56,126,98,48,12,70,110,5
6
1500 VDU23,202,120,124,36,36,126,96,96,
112
1510 VDU23,203,0,56,100,68,98,114,54,60
1520 VDU23,204,112,98,36,56,48,44,98,11
4
1530 VDU23,205,28,56,40,8,8,40,62,28
1540 VDU23,206,60,98,98,124,112,120,110
,102
1550 VDU23,207,6,60,120,24,24,24,24,28
1560 VDU23,208,0,65,130,0,56,36,36,36
1570 VDU23,209,0,0,56,68,87,204,240,252
1580 VDU23,210,252,252,124,62,14,12,24,
96
1590 VDU23,211,0,0,28,38,43,243,15,63
1600 VDU23,212,63,31,63,126,124,32,24,6
1610 ENDPROC
1620 :
1630 DEF PROClogo(d)
1640 SOUND1,9,53,25
1650 SOUND2,9,117,25

```



```

1660 SOUND3,9,97,25
1670 COLOUR7
1680 PRINTTAB(4,15);"SPOOKS'n'SPIRITS "
1690 PRINTTAB(9,17)"by"
1700 PRINTTAB(5,19)"MAT EASTMOND"
1710 ?&34F=32:??&30A=19
1720 VDU17,6,31,4,13,224,10,8,225,31,5,
14,252,17,5,31,8,13,192,193,10,8,8,194,1
95,17,2,31,11,13,245,10,8,17,3,228,17,6,
31,13,14,199,17,1,31,14,12,247,247,10,8,
8,248,248,10,8,8,249,249
1730 FORT=OTO d:NEXT
1740 VDU24,0,0;1279;990;:CLG:VDU26
1750 ENDPROC
1760 :
1770 DATA 201,202,203,203,204,201,32,20
8,32,201,202,205,206,205,207,201
1780 REM Error reporting routine
1790 MODE 7
1800 REPORT:PRINT " at line ";ERL
1810 END

```

* * * * *

```

10 REM Program SPOOK2
20 REM Program subject to copyright
100 ON ERROR GOTO 3120
110 DIM A$(7),A%(1),j%(8),s%(11),n%(8)
120 REPEAT
130 ?&34F=24:??&30A=25:COLOUR7
140 PRINTTAB(6,12)"PRESS SPACE"
150 REPEAT UNTIL INKEY=99
160 PROCsetup
170 PROCscore(0):PROCchime:PROCnew
180 REPEAT
190 PROCman:PROCcheck
200 IF INKEY=1 IF w%>0IF J%=0PROCgun
210 IF X%>1247 OR X%<0PROCnew
220 IF Y%=288 PROCdie
230 UNTIL f%
240 VDU4
250 VDU23,1,0;0;0;0;
260 FORa=0TO1999:NEXT
270 ?&34F=24:??&30A=25:COLOUR7
280 PRINTTAB(7,8);"GAME OVER "
290 PRINTTAB(8,10);STRING$(6-LEN(STR$
(s)), "0");s
300 SOUND1,9,20,20:SOUND2,9,53,20
310 FORa=0TO5999:NEXT
320 UNTIL FALSE
330 :
1000 DEF PROCman
1010 A%=X%:B%=Y%:C%=V%:H%=POINT(X%+28,Y
%-68)
1020 IF H%=0 OR H%=15 OR H%=3 IF J%<10R
J%>8 IF H%>14 IF F%<2 Y%=Y%-32:F%=1 EL
SE F%=0:IF V%=241 V%=224
1030 IF INKEY=73 IF H%=14 Y%=Y%+32:F%=2
:V%=241

```

```

1040 IF INKEY=105 IF POINT(X%+28,Y%-100
)=14 IF J%=0 Y%=Y%-32:F%=2:V%=241
1050 IF X%-s%<64 IF X%-s%>-64 IF Y%-s1%
<33IF Y%-s1%>-32 PROCdaget
1060 IF F%=2 SOUND0,1,6,1
1070 IF F% GOTO1130
1080 IF INKEY=74 IF J%=0IF H%>0IF H%<15
J%=9
1090 IF J% J%=J%-1:Y%=Y%+j%(J%):SOUND1,
1,n%(J%),1:IF H%=13 OR H%=2 OR H%=1 OR H
%=14 IF J%<7 J%=0:IF INT(Y%/32)<>Y%/32 Y
%=INT(Y%/32)*32
1100 IF INKEY=67 X%=X%+28:V%=224
1110 IF INKEY=98 X%=X%-28:V%=227
1120 IF A%=X%IF B%=Y%IF C%=V%ENDPROC
1130 MOVE A%,B%:VDU C%,10,8,C%+D%
1140 MOVE X%,Y%:VDU V%,10,8,V%+M%
1150 D%=M%:IF J%=0 M%=M%+1:IF M%=3M%=1
1160 ENDPROC
1170 :
1180 DEF PROCcheck
1190 IF R%=0 PROCzombie
1200 IF R%=2 PROCjumper
1210 IF R%=3 PROCfire
1220 ENDPROC
1230 :
1240 DEF PROCdaget
1250 IF w%=60R s%(scr)=0 ENDPROC
1260 IF scr>0 IF scr<>7 SOUND2,4,140,1:
GCOL3,3:MOVE s%,s1%:VDU196:s%(scr)=0:w%=
w%+1:PROCscore(75):GCOL3,6:ENDPROC
1270 SOUND2,4,140,1:w%=w%+1:PROCscore(0
):GCOL3,6
1280 ENDPROC
1290 :
1300 DEF PROCzombie
1310 G%=L%:K%=I%:I%=I%+N%:d%=L%
1320 IF I%<A%(0) OR I%>A%(1) N%=-N%:L%=
L%+1:IF L%=2 L%=0
1330 IF X%-I%<40IF X%-I%>-36IF Y%-O%<64
IF Y%-O%>-64 PROCdie
1340 MOVE K%,O%:PRINT A$(Z%+d%)
1350 MOVE I%,O%:PRINT A$(Z%+L%)
1360 ENDPROC
1370 :
1380 DEF PROCjumper
1390 G%=I%:K%=O%
1400 IF L%=7N%=-N%:L%=-1:SOUND3,5,20,2
1410 IF O%-Y%<65 IF O%-Y%>-1 IF X%-I%<1
28 IF X%-I%>-1 PROCdie
1420 L%=L%+1:I%=I%+N%:O%=O%-j%(L%)
1430 MOVE G%,K%
1440 VDU192,193,10,8,8,194,195
1450 MOVE I%,O%
1460 VDU192,193,10,8,8,194,195
1470 ENDPROC
1480 :
1490 DEF PROCfire
1500 FOR a=0TO29:NEXT

```



```

1510 K%=0%:G%=I%:O%=O%+L%
1520 IF O%>800 OR O%<256 PROCfset
1530 GCOL3,1:MOVE G%,K%:VDU A%(0)
1540 MOVE I%,O%:VDU A%(0):GCOL3,6
1550 IF I%-X%<56 IF I%-X%>-33 IF Y%-O%<
65 IF Y%-O%>-33 PROCdie
1560 ENDPROC
1570 :
1580 DEF PROCfset
1590 IF O%>800 O%=256 ELSE O%=800
1600 IF A%!=255SOUND0,3,4,1ELSE SOUND
3,2,0,1
1610 IF RND(3)=1 IF X%>50 IF X%<1216 I%
=X%:ENDPROC
1620 IF V%=224 I%=X%+128 ELSE I%=X%-96
1630 ENDPROC
1640 :
1650 DEF PROCgun
1660 w%=w%-1:PROCscore(0):gun=TRUE
1670 SOUND1,6,200,1
1680 T%=X%:IF V%=224 U%=64ELSE U%=-64
1690 GCOL3,3:MOVE T%,Y%-32:VDU196
1700 REPEAT
1710 T%=T%+U%
1720 IF R%=2 IF Y%-32=O%-32 IF T%-I%<12
9 IF T%-I%>-1 SOUND0,3,6,2:die=die-1:PRO
Cscore(1000):MOVE T%-U%,Y%-32:VDU196:T%=
1900
1730 MOVE T%-U%,Y%-32:VDU196:MOVE T%,Y%
-32:VDU196:GCOL3,6:PROCcheck
1740 IF R%=0 IF T%-I%<65 IF T%-I%>-65 I
F Y%=0% OR Y%=O%+32 MOVE I%,O%:PRINT A$(
Z%+L%):SOUND0,-15,6,2:PROCscore(750):R%=
3:I%=800:O%=800:L%=-48:A%(0)=197:GCOL3,1
:MOVE I%,O%:VDU197
1750 GCOL3,3
1760 UNTIL T%<-65 OR T%>1279
1770 GCOL3,6:gun=FALSE
1780 IF die=0 SOUND1,8,99,9:N%=60:L%=0:
FORa=0TO6:PROCjumper:NEXT:FORa=0TO4999:N
EXT:PROCscore(1500)
1790 IF die=0 IF scr=6 scr=7:PROCchime:
X%=1300:die=3:PROCnew
1800 IF die=0 IF scr=11 scr=0:Q%=Q%+1:P
ROCchime:X%=1400:die=3:PROCnew
1810 ENDPROC
1820 :
1830 DEF PROCdie
1840 IF gun=TRUE GCOL3,3:MOVE T%,Y%-32:
VDU196:T%=1280
1850 VDU24,0;224;1279;992;
1860 SOUND1,10,150,10:GCOL3,6
1870 MOVE X%,Y%:VDU V%,10,8,V%+D%
1880 MOVE X%,Y%:VDU C%,10,8,C%+D%
1890 V%=241:M%=1
1900 REPEAT:Y%=Y%-32:MOVE X%,Y%+32:VDU
C%,10,8,C%+D%:MOVE X%,Y%:VDU V%,10,8,V%+
M%:C%=V%:D%=M%:UNTIL Y%<200

```

```

1910 SOUND0,3,6,1:FORa=0TO999:NEXT:l%=1
%-1:PROCscore(0):X%=rx:Y%=ry:MOVE X%,Y%:
VDU V%,10,8,V%+M%
1920 ENDPROC
1930 :
1940 DEF PROCearth
1950 VDU17,15,17,128,31,X,Y
1960 FOR U%=Y TO Y+V
1970 FORT%=X TO X+L:VDU229+RND(4):NEXT
1980 VDU31,X,U%:NEXT
1990 FOR T%=X TO X+L
2000 IF RND(3)=1 VDU231 ELSE VDU234
2010 NEXT
2020 VDU31,X,Y-1,17,15,17,131:FOR T%=X
TO X+L:VDU235:NEXT:VDU31,X,Y-2,17,141,17
,2:FOR T%=X TO X+L:VDU235:NEXT:COLOUR128
2030 ENDPROC
2040 :
2050 DEF PROCnew
2060 IF X%=1300 X%=280:Y%=352:GOTO2100
2070 IF X%=1400 X%=532:Y%=352:GOTO2100
2080 IF X%>1247 scr=scr+1:X%=0
2090 IF X%<0 scr=scr-1:X%=1216
2100 VDU17,128,4,28,0,24,19,4,12,26
2110 PROCscreen:VDU5:GCOL3,6:D%=M%
2120 MOVE X%,Y%:VDU V%,10,8,V%+M%
2130 IF R%=0 MOVE I%,O%:PRINTA$(Z%+L%)
2140 IF R%=2 MOVE I%,O%:VDU192,193,10,8
,8,194,195
2150 IF R%=3 GCOL3,1:MOVE I%,O%:VDU A%(
0)
2160 IF s%(scr)=1 GCOL3,3:MOVE s%,s1%:V
DU196
2170 GCOL3,6:N%=24:rx=X%:ry=Y%
2180 ENDPROC
2190 :
2200 DEF PROCtri
2210 x=-1:FOR U%=Y TO Y+L STEP2:VDU17,1
5,31,X,U%
2220 IF x<>-1 FORT%=X TO X+x:VDU229+RND
(4),10,8,229+RND(4),11:NEXT
2230 VDU239,10,8,240:x=x+1:NEXT:VDU31,X
,Y+L+2,17,15:FORT%=X TO X+x:VDU234:NEXT:
VDU17,2,17,141,31,X,Y,239
2240 ENDPROC
2250 :
2260 DEF PROCtril
2270 x=0:VDU17,128,17,15:FOR U%=Y TO Y+
L STEP2:VDU31,X-x,U%,250,10,8,251,11
2280 IF x>0 FORT%=X-x TO X-1:VDU229+RND
(4),10,8,229+RND(4),11:NEXT
2290 x=x+1:NEXT:VDU31,X-(x-1),U%-1:FORT
%=X TO X+x-1:VDU234:NEXT:VDU17,2,17,141,
31,X,Y,250
2300 ENDPROC
2310 :
2320 DEF PROCcladder
2330 VDU5:x=X*32:y=1024-(Y*32)

```



```

2340 GCOL0,14
2350 FOR a=y-(L*32) TO y STEP 32
2360 MOVE x,a:VDU236:NEXT:VDU4
2370 ENDPROC
2380 :
2390 DEF PROCscreen
2400 RESTORE 3000:IF scr>0 FOR o=0TO sc
r-1:READ T$,a,a,a,a,a,a,a,a:NEXT
2410 READ T$:FOR A%=1TO LEN(T$)STEP7
2420 K=ASC(MID$(T$,A%,1))-49:X=ASC(MID$(
T$,A%+1,1))-49:Y=ASC(MID$(T$,A%+2,1))-4
9:L=ASC(MID$(T$,A%+3,1))-49:C=ASC(MID$(T
$,A%+4,1))-49:BC=ASC(MID$(T$,A%+5,1))-49
:V=ASC(MID$(T$,A%+6,1))-49
2430 VDU17,C,17,BC+128
2440 IF K=1 FORL%=X TO X+L:VDU31,L%,Y,V
+192:NEXT
2450 IF K=2 PROCearth
2460 IF K=3 VDU31,X,Y:PRINT A$(V)
2470 IF K=4 PROCtri
2480 IF K=5 PROCcladder
2490 IF K=6 PROCtr11
2500 IF K=7 FOR B%=Y TO Y+L:FOR L%=X TO
X+V:VDU31,L%,B%,246:NEXT,:VDU17,12,17,1
28:FORL%=X TO X+V:VDU31,L%,Y-1,238:NEXT
2510 NEXT
2520 READ R%,I%,O%,L%,A%(0),A%(1),Z%
2530 READ s%,s1%:IF R%>0 ENDPROC
2540 IF Q%=2 IF scr<7 Z%=4:O%=O%-32
2550 IF Q%=3 Z%=6
2560 ENDPROC
2570 :
2580 DEF PROCchime
2590 VDU4,28,0,24,19,4,12,26,17,7
2600 ?&34F=24:??&30A=25
2610 VDU23,1,0;0;0;0;
2620 IF scr=0 PRINTTAB(5,9);"THE GRAVEY
ARD " ELSE PRINTTAB(6,9);"THE CASTLE "
2630 SOUND1,9,53,20:SOUND2,9,117,20
2640 FORa=0TO4999:NEXT
2650 ?&34F=32:??&30A=19
2660 VDU28,0,21,19,5,12,26
2670 ENDPROC
2680 :
2690 DEF PROCscore(d)
2700 s=s+d:VDU4,31,0,28,17,7
2710 ?&34F=24:??&30A=25
2720 PRINT;STRING$( (6-LEN(STR$(s))) , "0"
);s
2730 ?&34F=32:??&30A=19
2740 PRINTTAB(8,28);SPC(7)
2750 IF w%>0 VDU31,8,28,17,3:FORT=1TO w
%:VDU196:NEXT
2760 IF o%=1% VDU5:ENDPROC
2770 o%=1%
2780 PRINTTAB(17,27)" "
2790 PRINTTAB(17,28)" "
2800 IF l%=-1 VDU5:ENDPROC
2810 IF l%=-2 f%=3:VDU5:ENDPROC

```

```

2820 FORT=19TO 19-1% STEP-1
2830 VDU31,T,27,17,6,227,10,8,228
2840 NEXT:VDU5
2850 ENDPROC
2860 :
2870 DEF PROCsetup
2880 RESTORE 2990
2890 scr=0:l%=2:o%=1:w%=0:die=3:s=0
2900 X%=532:Y%=352:V%=224:M%=1:D%=1
2910 J%=0:F%=0:Q%=1:f%=0:gun=FALSE
2920 FORa=0TO11:s%(a)=1:NEXT
2930 FORa=0TO7:READ n%(a):NEXT
2940 FORa=0TO7:READ j%(a):NEXT
2950 FORa=0TO7:A$(a)="" :NEXT
2960 FOR K=0TO7:READ a:FOR V=1TO a:READ
C:A$(K)=A$(K)+CHR$(C):NEXT,
2970 ENDPROC
2980 :
2990 DATA 8,16,32,48,48,32,16,8,-32,-16
,-16,0,0,16,16,32,4,244,10,8,225,4,245,1
0,8,228,7,247,10,8,248,10,8,249,5,253,10
,8,254,252,1,198,1,199,4,209,10,8,210,4,
211,10,8,212
3000 DATA 3@@511427H>>3<27I>>3<21H2>3<2
1I2>3^7?>41113=C81147<A611142E121347E121
36S>;111,0,1152,672,0,952,1247,0,448,320
3010 DATA 21HD>3<21ID>3^31@;1175<71114
6;3@1342;3@1329C53>>4BF15146=>;111,0,64,
352,0,16,1248,0,644,640
3020 DATA 75;711121HD>3<21ID>3^36=?1162
2B73>[2<B93>[6>B71114681@134@F15144BF151
4,0,32,544,1,16,476,0,868,320
3030 DATA 21H2>3<51:711121BD3>[26A?=1_
,3,600,700,48,255,0,0,756,576
3040 DATA 7D;811121BD3>[21A?=1_,3,128,7
00,48,255,0,0,756,576
3050 DATA 3@C51147?A611157;511131=61162
1HD>3<21ID>3^21B93>[60A81116;;>111,0,760
,352,0,756,952,0,756,576
3060 DATA 31C711558A511121H<>3<21I<>3^2
?H6>3<2?I6>3^4?E1@13,2,1000,384,0,0,0,0,
420,544
3070 DATA 21I;@1985H23>@85::3>7259@3>'8
A>:3>460>;111,0,896,320,0,700,952,4,264,
800
3080 DATA 23I>@198=H23>881><3>387>33>38
=>33>58B>:3>3219A3>g6Q:>111,0,896,672,0,
784,1016,0,784,352
3090 DATA 23I>@198A==3>481A93>381>33>68
6H23>78<:33>96;>91116G:=111,3,600,700,36
,255,0,0,336,672
3100 DATA 25I<@1981:@3>488;?3>48?<>3>6,
0,448,736,0,448,672,4,880,736
3110 DATA 25I<@981<>3>485<33>58A<>3>48
=<33>425H<@19,2,752,736,-1,0,0,0,504,736
3120 REM Error reporting routine
3130 MODE 7:REPORT
3140 PRINT" at line ";ERL
3150 END

```


MODEM MASTER

Modem Master from BBC Soft has been attracting a lot of interest from comms enthusiasts. Despite its low price, the software has much to commend it, as Peter Rofford describes.

Product Modem Master
Supplier BBC Software
 80 Wood Lane,
 London W12 OTT.
 Tel. 01-743-5588
Price £12.95 inc. VAT

Modem Master is a terminal software package designed for accessing viewdata and scrolling on-line text systems. It is supplied on a 40 track 5.25" DFS disc along with a very good 52 page manual. The disc can be converted to 80 track with a supplied utility, and may be transferred to ADFS. The package will work with the Model B, B+, B+128, the Master 128, and is second processor compatible.

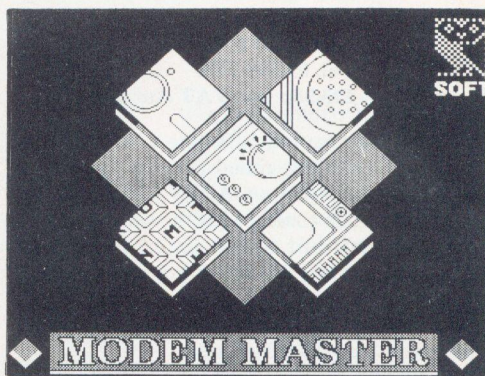
Modem Master is supposed not to be modem specific, and is claimed to work with most manual modems. There are several modem drivers supplied on the disc, including ones for manual, Hayes, DTI schools and the Pace Nightingale. You can write a driver for your own modem, and details to assist you in this are given in the manual, but you will need to be a machine code programmer. The program will store a list of up to 40 on-line databases along with their phone numbers, format and log-on details. This makes the package easy to use by those not familiar with computers or comms. By simply booting the disc and pressing a few keys you can log onto the required database.

VIEWDATA

The Viewdata section of Modem Master provides a good selection of facilities mostly initiated by the function keys. Loading and storing of frames, frame printing, frame tagging and telesoftware downloading are just a few of the many features. There is also a very competent mailbox editor which allows off-line editing of mailbox frames. It will monitor the number of Escape codes you employ, allowing

easy use of colour and graphics, without exceeding Prestel's limit of 32 Escape codes per frame. No pixel editor is provided for graphics, but characters can be chosen by asking the editor to cycle through them at the current cursor position.

A rather novel feature of Modem Master is the mailbox number notepad, in which a number can be entered at any time when on-line. This is stored and can be recalled later when sending a mailbox frame, really excellent for those like myself who can never find pencil and paper!



SCROLLING TEXT

In scrolling text mode, the software is equally comprehensive, providing many features like buffering, uploading and downloading of files in ASCII format, Xmodem file transfer, choice of screen mode and the ability to configure the terminal in a variety of ways to suit your needs.

SIDEWAYS RAM

The package is a trifle slow when accessing the disc, and someone used to ROM packages may find this irritating. Master and B+128 owners can be spared this, as the disc is supplied with a utility which allows it to run in sideways RAM.

CONCLUSION

Compared to packages such as BEEBUG's own Command ROM or Soft Mac's Commsoft, Modem Master does lack their power, sophistication and certain other facilities. Do not dismiss it however, as for a modest price you get a very competent terminal package which provides good easy-to-use facilities. B

ADU ROM from Pineapple

If you find the ADFS difficult to use, then the ADU ROM from Pineapple Software may be of interest. Peter Rochford reports.

Product ADU ROM
Supplier Pineapple Software,
39 Brownlea Gardens,
Seven Kings, Ilford,
Essex IG3 9NL.
Tel. 01-599 1476

Price £33.35p inc. VAT and p&p.

ADU from Pineapple Software is a 16K filing system utility ROM, aimed at users of both the BBC Model B with an ADFS fitted and the Master series. It comes with a well-written 28 page manual and is packaged in a video cassette-type case. The ROM has some 15 'star' commands as listed in Figure 1, several of which provide 'instant' versions of many of the utilities on the ADFS disc.

```

ADFS Utilities 1.02
ADU
BACKUP <source> <target>
CATALL (<dir>)
DFSADFS <DFS drive> <ADFS dir>
DIRALL (<dir>)
DIRCOPY <$.afsp> <$.dir>
DIRDESTROY <dir>
DISCEDIT
DRIVE (<drive>)
FORMAT <drive> (<size>)
KILLADU
MENU (<dir>)
PWRBRK
VERIFY <drive>
VFORMAT <drive> (<size>)
    
```

Figure 1

By using several control keys, you can move around the ADFS selecting directories or files. If a file is selected, ADU will try to identify what type it is and act accordingly. For example Basic files will be CHAINED and machine code programs *RUN. Files for View, ViewSheet, Inter-Word etc. are recognised, the ROM entered, and the specified file loaded.

Some of the more interesting commands are:

***BACKUP** - Uses all available main and second processor memory to backup a disc. Only the sectors with information on the source disc are copied onto the target disc. This feature makes the procedure VERY fast.

***DFSADFS** - Transfers all files from a DFS disc to an ADFS directory. Creates any new directories automatically.

***DIRCOPY** - Copies a directory structure from one place to another.

***DIRDESTROY** - Enables the deletion of any directory from a disc, and all items and directories below it.

***DRIVE** - This allows programs that were written for DFS and contain *DRIVE, to work with ADFS. The *DRIVE is trapped without error generation and a *MOUNT issued. A very nice feature.

***PWRBRK** - Simulates a power-up reset of the computer.

One command that needs special mention is *MENU. When invoked, this command provides a catalogue of the currently selected disc, along with a list of all the ROMs in the machine.

Whilst in menu mode, many other commands may be issued: to unplug ROMs for example, load RAM banks, wipe RAM banks and delete files. A feature I really like is the COMPACT command, which will compact the current disc until ALL free space has been moved. This saves the usual tedious process with the ADFS of having to repeatedly type *COMPACT to achieve this (although we published a utility last month in BEEBUG to do just this).

CONCLUSION

In general this is a useful ROM for ADFS users. The backup and compact commands are particularly good as they save so much time. The menu features are also very powerful and take some of the pain out of using the unwieldy ADFS. I have to say though, that comparisons must be made between this ROM and both Advanced Computer Products ADT ROM and BEEBUG's Master ROM. Both of these provide most of what ADU offers and much more for a similar price, so do check these out.



POSTBAG



POSTBAG

KEYSTRIP GENERATOR

The Keystrip Generator program published in BEEBUG Vol.6 No.4 is, as suggested, one of the best utilities yet for this purpose. However, the program does have the disadvantage that the name of the software to which the strip relates does not get printed. With a lot of strips, the one you want can be difficult to identify quickly.

I have therefore made some very simple changes to the program to allow it to print the relevant name in the space between the key numbers and the first line of labels:

```
1745 PROCprint(4,17,"Key
strip title   "):
REM allow 23 spaces
after the words
1746*FX15,0:
    REM Flush buffers
1747 INPUTTAB(7,19),NAME$
and then change line 1830 to:
1830PRINT" ";NAME$:
REM print title
```

Tony Briselden

The Keystrip generator certainly proved a popular utility, and Tony Briselden's amendment provides a very worthwhile little extra.

ADDRESS BOOK

I would like to suggest two little improvements to the excellent Address Book program (BEEBUG Vol.6 No 3). The first (line 1005) sets the printer (mine is a Kaga Taxan) for a left margin of 12 (code 27,108,12) and NLQ mode

(code 27,40). Any other code could of course be incorporated.

The second alteration (lines 1530-1540) is intended to prevent the deletion of a wanted file. It incorporates a ("Are you sure?") check when overwriting an address file already existing on disc.

```
1005 VDU2,1,27,1,64,1,27
,1,108,1,12,1,27,1,40,3
```

```
1530 F%=OPENIN(T$+"-BOOK
"):IF F% THENCLOSE#F:I$=F
Nchoice("Are you sure","Y","N
",11):IF I$="N" THEN ENDPROC
1540 IF F% THEN OSCLI("D
ELETE "+T$+"-BOOK")
```

```
1545 PRINTTAB(3,2)F$:SPC
10"Saving";S$;"book";R$;T$;
SPC10TAB(17,7);
```

A.R.Mattocks

This was another very popular program about which we have received considerable correspondence. We are always pleased to hear from BEEBUG members who have improved or amended the programs we publish, though we cannot guarantee to publish every letter which we receive.

SIDEWAYS RAM BUFFER

I have recently completed a modification to the sideways RAM buffer utility of mine which you published in BEEBUG Vol.5 No.8. This overcomes a problem in using the printer buffer with Wordwise.

The problem is that Wordwise uses Escape to move in and out of the menu. Escape also

clears the buffer via the count/purge vector. The modification checks the Escape flag at &FF and if set causes the normal 63 byte buffer to be purged. The modifications are as follows

```
2520 PLP: .esc:PLP:CLI:
    JMP (oldcnpv)
2535 LDA #&80:AND &FF:
    BNE esc
```

C.C.Radcliffe

We hope readers will find this modification useful. We would also mention that some difficulty has been encountered in using the program with the BEEBUG Sideways RAM Module (see BEEBUG Vol.5 No.7). The solution for Basic II users is to assemble the code in main memory ready to run from address &8000, and then load the assembled machine code into sideways RAM.

NO COMMENT

Although I am sure you are aware of the fact, I consider Ashley Kitson's Mode 7 Screen Editor (Vol.6 No.3) to be absolutely brilliant. Not only that, it must be the first program I have ever typed in that worked faultlessly first time. It was most heartening having at least the total code length check.

It never ceases to amaze me how, after five years, you can still keep publishing gems of this nature. It will no doubt occupy me for many hours.

D.P.Dyer



HINTS HINTS HINTS HINTS HINTS

and tips and tips and tips and tips and tips

INTERWORD MULTI-FILE

John Clemence

While trying to combine separate files into a multi-file document using Interword, I kept getting a "File Too Long" error, even though they'd all been saved using Interword.

The answer is to edit each file in turn by marking the first half page or so, and save this marked section to a separate file. This marked section is then deleted, and the rest of the file saved as usual. The short part of the file will then be accepted by Interword as part of a "multi-file" document, and then you can write the rest of the file to the cursor from option 4 of the main menu.

GIANT SCROLLER

David Shepherdson

The Giant Scroller (BEEBUG Vol.6 No.3) is excellent, however it is not necessary to wait a while when pressing Escape. You can also alter the start of the scroll. In addition, by selecting Big rather than Giant text, giving "line" a value of 5 will allow a more central display.

To speed up Escape add the following lines:

```
1615 BIT&FF:BPL notesc:BRK
1616 EQU17:EQUS"Escape"
1617 BRK:.notesc
```

To alter the start of scroll, add:
1035 screen=&7C00+40*line
where "line" is the screen line

to start on. Amend lines 1280, 1430,1440 and 1630:

```
1280 LDA #screen MOD 256:
STA &70:LDA #screen
DIV 256:STA &71
1430 .decoderow:STX &73:
LDA #(screen&27) MOD 256
1440 STA &70:LDA
#(screen&27) DIV 256:STA
&71:LDX #600
1630 LDA #&00:STA &70:
LDA #&7C:STA &71
```

MAKING YOUR REMS STAND OUT

Paul Willey

Prior to entering your program type *FX228,150. Now each time you put a REM statement in your program use:

```
100 REM"<Shift-Ctrl-f7>
<Shift-f4>Example rem
Note the quote following the REM - this is essential. In mode 7 this will make your remarks appear highlighted in blue on a white background.
```

QUICK ADFS & DFS ENABLE

Graham Stanley

For machines with both DFS and ADFS in situ, simply pressing Ctrl-A-Break or Ctrl-D-Break will enable the ADFS and DFS respectively. Similarly Shift-A-Break and Shift-D-Break will perform an auto-boot in the appropriate disc filing system.

MAGSCAN DATA FILES

David M. Pratt

Occasionally commas have appeared in Magscan files, this unfortunately allows some entries to be missed. use

the wildcard facility (see Magscan manual page 4) where, for example, <ROM> only finds ROM or Rom, whereas <.ROM.> will find Eprom, Promenade, ROMs ROMS etc. The alternative is to use Wordwise (or View) to replace all occurrences of a comma with a space.

ONE LINE METRONOME

A.S. Clarke

The following can be used in a program or as a key definition to provide a metronome.

```
INPUT"Beats per min",N%:
n%=(6000/N%):REPEAT
TIME=0:t%=TIME:REPEAT UNTIL
TIME=t%+n%:SOUND &11,-15,
56,1:UNTIL FALSE
```

INTERWORD AND LONG COMMANDS

Chris Wragg

An oversight occurred and the last paragraph was missed off Chris Wragg's hint on page 61 of BEEBUG Vol.6 No.3. Simply append this paragraph to the first hint on page 61.

When complete it is wise to check accuracy by printing out before the next step. When satisfied, place the cursor in the first command position, and use Ctrl-A to *suck* the subsequent commands on to the first. All except the last command are now invisible in edit mode, but are still there. Printing out now gives the graphics without spaces as required.

ⓑ

BEEBUG Technical BEEBUG Technical

The Beebug Technical Support Team replies to hundreds of technical enquiries every month. We believe that many of the answers will be of interest to other readers, and we will be featuring the more useful ones regularly under the heading of BEEBUG Technical.

USING RAM MODULES IN ROM BOARDS

Why can I not address additional RAM modules when I plug them into my BBC Micro?

This question is quite often posed by members who have an expansion ROM board fitted to their machine. The large majority of ROM boards also have the facility to install 16K bytes of sideways RAM. If this is the case, all attempts at writing to Sideways RAM will be intercepted and redirected to this default RAM. For this reason, if you wish to install additional RAM modules on your ROM board, you must first 'write protect' the default RAM so that data is not automatically written to it. You are then able to write to any RAM module simply by addressing the socket directly. In order to 'write protect' the RAM a switch must be fitted to the ROM board. These switches are available from Beebug.

Note that if you have a ROM board fitted to a BBC B+ it is not possible to use any

additional RAM modules of any description. Fortunately the BBC B+ 128 already has 64K bytes of sideways RAM. If you have a BBC B+ 64, Beebug is able to supply and install an additional 64K bytes of Sideways RAM at a cost of £37.05 to members.

READING DFS DISCS ON THE MASTER COMPACT

Is it possible to implement the DFS on my Master Compact?

Although it is possible to connect a second 5.25 inch disc drive to the Compact you are still faced with the problem that many discs will be formatted to run using the Acorn DFS. Unlike the Master 128, the Master Compact does not contain the DFS in firmware. However, there is a copy of the DFS supplied on the latest Welcome disc. This file is called \$.DFS and may be loaded into one of the Compact's Sideways RAM banks (for example bank 6) with the command,

```
*SRLOAD DFS 8000 6
```

If you have an early copy of the Welcome disc it is likely that this file will not be present. However, a copy of the latest Welcome disc may be obtained from Beebug. Please phone for details.

SIDEWAYS AND SHADOW RAM

I need more memory for use within a word processor. Do I need additional Sideways or Shadow RAM?

The terms Sideways and Shadow RAM regularly cause confusion, and you must be very careful that you purchase the correct type of memory expansion board for your needs. Shadow RAM eliminates the problem which arises through insufficient memory when using high resolution screen displays. The memory required for producing 80 column text or high resolution graphics can be as high as 20K bytes leaving only 12K bytes for normal use. The memory on a Shadow RAM board is used to store the screen, thereby allowing all the BBC micro's memory to be used for data.

Sideways RAM allows the user to store information in a bank of RAM located at &8000 in memory. This can be used for loading ROM images, developing ROM software, or for utilities such as a printer buffer. Note that most of these applications require additional software such as our Romit package. B

If you require our technical support services you are quite welcome to ring us, but it is often easier for us to document technical explanations in the form of a letter. If you wish to write to our Technical Support Department. Please remember to quote your membership number and to enclose an SAE.

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.

MEMBERSHIP SUBSCRIPTION RATES

£ 6.90 - 6 months (5 issues) UK only
 £12.90 - 1 year (10 issues) UK, BFPO, Ch.I
 £19.00 - Rest of Europe & Eire
 £23.50 - Middle East
 £26.00 - Americas & Africa
 £28.00 - Elsewhere

Prices held until Nov '87.
 Renew now and extend your
 subscription at last year's rates

BACK ISSUE PRICES until 30th Nov '87

Volume	Magazine	Cassette	5"Disc	3.5"Disc
1	£0 40	£0 40	-	-
2	£0 40	£0 40	£2	-
3	£0.50	£1	£3	-
4	£0.60	£2	£4 25	£5 75
5	£1	£3	£4 75	£5 75
6	£1.30	£3		

All overseas items are sent airmail. We will accept official UK orders for subscriptions and back issues, but please note that there will be a £1 handling charge for orders under £10 which require an invoice. Note that there is no VAT in magazines.

FURTHER DISCOUNTS

We will allow you a further discount:

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

POST AND PACKING

Please add the cost of p&p:

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

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

BEEBUG MAGAZINE is produced by BEEBUG Ltd.
 Editor: Mike Williams

Editorial Assistant: Kristina Lucas
 Production Assistant: Yolanda Turuelo
 Membership secretary: Sara Taylor
 Editorial Consultant: Lee Calcraft
 Additional thanks to Sheridan Williams and Adrian Calcraft.

All rights reserved. No part of this publication may be reproduced without prior written permission of the Publisher. The Publisher cannot accept any responsibility whatsoever for errors in articles, programs, or advertisements published. The opinions expressed on the pages of this journal are those of the authors and do not necessarily represent those of the Publisher. BEEBUG Limited

CONTRIBUTING TO BEEBUG PROGRAMS AND ARTICLES

We are always seeking good quality articles and programs for publication in BEEBUG. All contributions used are paid for at up to £40 per page, but please give us warning of anything substantial that you intend to write. A leaflet 'Notes to Contributors' is available on receipt of an A5 (or larger) SAE. Please submit your contributions on disc or cassette in machine readable form, using "View", "Wordwise" or other means, but please ensure an adequate written description is also included. If you use cassette, please include a backup copy at 300 baud. In all communication, please quote your membership number.

BEEBUG Ltd (c) 1987

Magazine Disc / Cassette

OCTOBER 1987 DISC/CASSETTE CONTENTS

FILER CATALOGUER - Keep track of the multitude of files stored on all your discs with this useful utility.

EXPLORING ASSEMBLER (PART 4) - Three complete example programs in this, the next part of our introduction to machine code programming:

MESSAGE PRINTER
FULL MODE 7 FILL
BLOCK MOVE

SPOOKS AND SPIRITS - Dodge the thunderbolts in this classic arcade game.

REGRESSION ANALYSIS - Calculate and display linear regression and correlation. Sample data files are included for you to experiment with.

BEEBUG WORKSHOP -

PRECISION ARITHMETIC. Routines are provided for addition and subtraction of numbers with up to 250 digits.

SUPER FUNCTION KEYS - Make every key on the keyboard a function key with this short utility.

THE GHOST HOST - Skeleton software to enable you to set up your own bulletin board system.

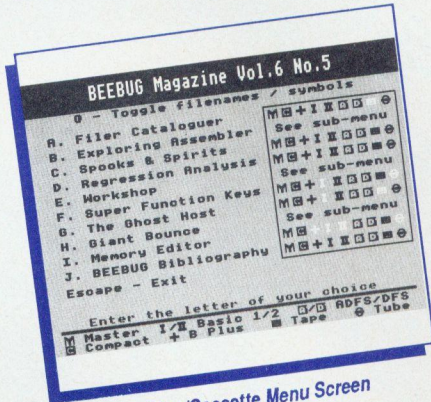
BARRY CHRISTIE VISUALS -

GIANT BOUNCE. Very fast animation effects using the 6845 video controller chip.

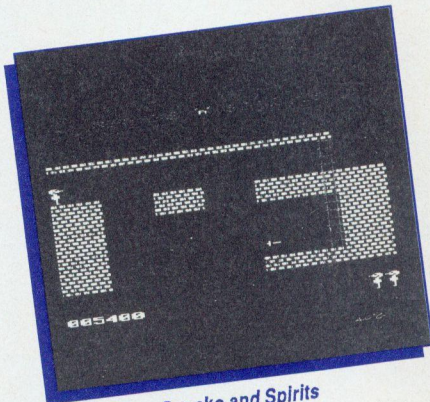
THE MASTER SERIES -

DYNAMIC MEMORY EDITOR. Display and edit the Master's memory including sideways RAM, private RAM and CMOS RAM.

MAGSCAN - Bibliography data for this issue of BEEBUG



Disc/Cassette Menu Screen



Spooks and Spirits

All this for **£3 (cassette)**, **£4.75 (5" disc)**, **£5.75 (3.5" disc) + 50p p&p**.
Back issues (5" disc since Vol 3 No 1, cassettes since Vol 1 No 10) available at the same prices

SUBSCRIPTION RATES	UK ONLY			OVERSEAS		
	5" Disc	3.5" Disc	Cassette	5" Disc	3.5" Disc	Cassette
6 months (5 issues)	£25.50	£29.50	£17.00	£30.00	£34.00	£20.00
12 months (10 issues)	£50.00	£58.00	£33.00	£56.00	£64.00	£39.00

Prices are inclusive of VAT and postage as applicable. Sterling only please.
Cassette subscriptions can be commuted to 5" disc subscription on receipt of £1.70 per issue of the subscription left to run (3.5" disc £2 per issue)
All subscriptions and individual orders to
BEEBUG, Dolphin Place, Holywell Hill, St. Albans, Herts. AL1 1EX



Beebug C is a full implementation of the language C to the Kernighan and Ritchie standard, including a number of powerful extensions. Beebug C produces fast, compact code and supports full floating point maths. It is supplied on ROM with a comprehensive set of library functions on disc.

Runs on a standard 32K BBC model B, B+ or Master 128
40/80 Track DFS and ADFS compatible
Support for Acorn operating system (Mode, Osbyte, Vdu etc.)
Powerful Command-Line interpreter with over 20 commands and qualifiers
Expandable run-time library on disc containing approximately 100 functions

Full macro-handling facilities
Twelve standard header files declaring nearly 200 functions and macros
Compiler warning messages for non-portable features
Debugging facilities and helpful error messages
Extensions including: void and unsigned data types, function prototypes, # pragma and # redef

To write C programs you will need a text editor or wordprocessor. Beebug C is supplied on two 16K ROMs with a library disc and user guide (please note that this guide explains how to use Beebug C but does not teach C).

Please add £1 carriage and state whether a 40 or 80 track disc is required.