Vol.6 · No.10 · April 1988

BEEBUG

FOR THE BBC MICRO & MASTER SERIES



- SHARE ANALYSIS PACKAGES REVIEWED
- OVERLAYING TECHNIQUES
- ADFS VIEW MENU

BEEBUG Vol.6 · No.10 · April 1988

FEATURES

ARM Wrestling 9
An ADFS Menu for View 12
Resistor Calculation 16
Now C Here - Strings 'n' Things 18
Overlaying Techniques 26
First Course -
Character Control (Part 2) 30
The Master Pages -
Enhanced Printer Buffer 41
Automatic Date Stamper 45
Master Hints 46
Video Cassette Cataloguer (Part 2) 48
Workshop -
Using Printers (Part 2) 50
The Comms Spot 52
Exploring Assembler (Part 9) 54
IBM to BBC Transfer Utility 59
Knight Quest 62

REVIEWS

Real Time Solids Modeller 6
Stock Market Analysis 22
PCB Autoroute 25
Solidisk Real Time Clock 47

REGULAR ITEMS

Editor's Jottings 4
News 4
Supplement 33-40
Master Hints 46
Points Arising 61
Hints and Tips 68
Postbag 69
Subscriptions & Back Issues 70
Magazine Disc/Tape 71

HINTS & TIPS

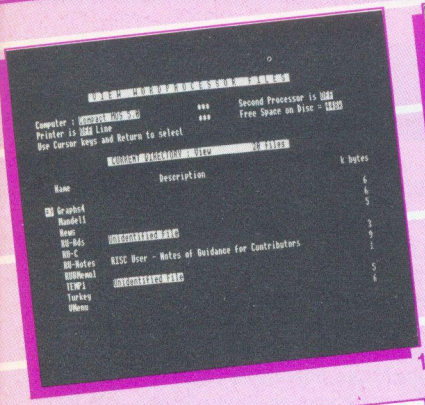
GENERAL
Invisible Calculations
Colour View
Basic Priority on the B+
Power-up Configuration
Double Stepping

MASTER
Automatic Backup
Simple Auto-Save

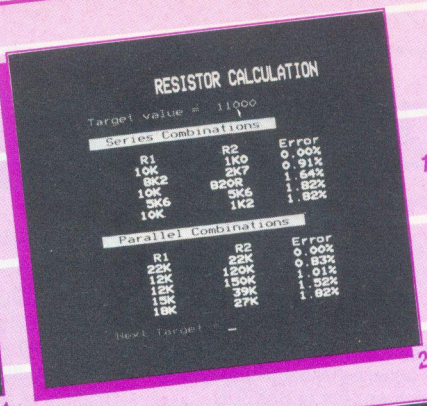
PROGRAM INFORMATION

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.

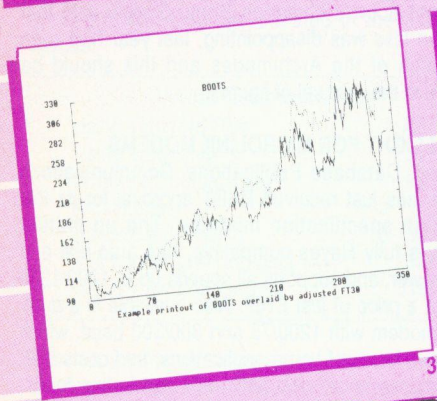


2.

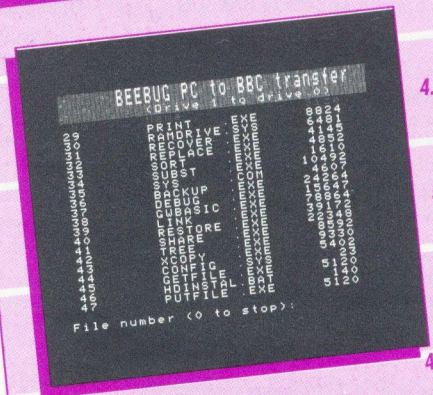
1. ADFS View Menu

2. Resistor Calculation

3. Share Analysis Packages



3.

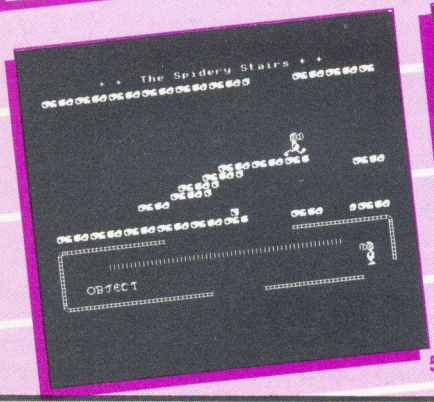


4.

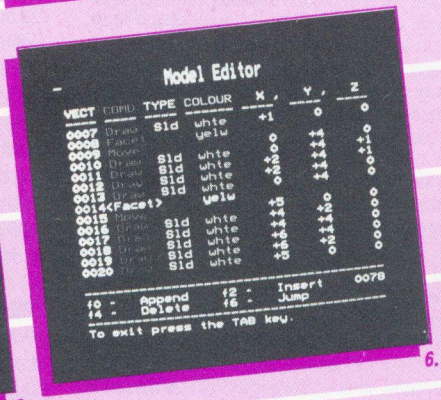
4. IBM to BBC File Transfer

5. Knight Quest

6. Real Time Solids Modeller



5.



6.

below. These show clearly the valid combinations of machine (version of Basic) and filing system for each item, and Tube compatibility. 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

Master (Basic IV)

Compact (Basic VI)

Model B (Basic II)

Model B (Basic I)

Electron

Filing System

ADFS

DFS

Cassette

Tube Compatibility

Tube

Editor's Jottings

LOOKING FORWARD WITH BEEBUG

With next month marking the start of our seventh year of publication, we have taken a number of steps to improve the service which we offer to BEEBUG members.

We are very pleased to welcome David Spencer to the position of Technical Editor, and we have also recruited a further member of staff to assist with production, advertising and administration. These, and other changes , will ultimately enable to meet our aim of publication at the start of each month - a move which we know members will particularly appreciate.

In Volume 7 we look forward to a further interesting year of publication when we shall be doing all that we can to support the use of Acorn's range of computers, and to provide BEEBUG members with a first rate magazine.

Please note that we will be distributing the complete index to volume 6 free with the next issue (Vol.7 No.1). This provides a most effective way of finding any article or program that you want to refer to. And if you haven't already done so why not get a binder to keep your complete set of volume 6 magazines in order - supplies will also be available at the Micro User Show.

THE MICRO USER SHOW

The first Micro User Show to be held in London this year takes place at its usual venue of the Royal Horticultural Hall, Westminster from the 13th-15th May. BEEBUG will have a large stand at this show, with all our magazines, cassettes and discs, plus our own range of software and hardware, and a wide range of other software and hardware from cables to complete systems. RISC User will have a separate stand for all things Archimedes, so don't forget to visit us there too.

Rumour has it that there will be no Acorn User Show this summer, so the Micro User Show will be even more important than before. And Acorn will also be attending the May show as well.

..News..News..

ACORN ANNOUNCES MAJOR LOSSES

Acorn Computers released their end of year figures on 31st March, showing a loss on the year of £3.2 million, against last year's profit of £1.35 million. This large loss is coupled with a reduction in turnover of almost £10 million to just over £36 million. Acorn claims that £2.4 million of the loss is due to the Custom Systems Division, which has now been closed. In a press statement Acorn's chairman Bruno Soggiu said that while the loss was disappointing, last year had seen the launch of the Archimedes and this should be reflected in the next set of figures.

GREEN LIGHT FOR MICROLINK MODEMS

Microlink, Database Publications' Communications division, has just received BABT approval for its low cost, high specification modems. The up market modem is fully Hayes compatible, has auto-dial and auto-answer, and supports all speeds up to 1200/1200 baud, at a price of just £169. Also available is a dual-speed modem with 1200/75 and 300/300 baud, which is suitable for most home applications, and costs £99. Both devices come complete with connecting leads, software and a month's free use of Telecom Gold.

The modems are currently available for BBC machines and the IBM PC (and compatibles), with versions being promised for the Atari ST, Amstrad PCW and the Apple Macintosh. For more details ring Mike Cowley on (0625) 878888 or write to: Microlink, Europa House, Adlington Park, Adlington, Macclesfield SK10 5NP.

EDUCATION SALES SOAR

While some people claim that the future for BBC computers in schools is not good, this is not reflected in the sales figures of at least one educational package, *Geordie Racer* from BBC Soft. This is an adventure type program linked to the current series of the BBC TV schools programme Look and Read, and has sold over 4000 copies since its launch three months ago. This makes it a very successful product for BBC Soft, and one which has taken the company by surprise. For details of this and other software from BBC Soft ring 01-576 0548 or write to: BBC Soft, Woodlands, 80 Wood Lane, London W12 0TT.

LEARN WITH ACORN

Acorn's own training centre is branching out and offering one and two day courses for hobbyists and other users of Acorn machines. There are courses on a wide range of subjects, including machine specific seminars on the Master series and Archimedes, as well as tutorials on the C programming language and applications such as ViewStore. Acorn will also consider requests for specific subjects if the demand is great enough. Courses planned in the next month include:

Acorn Courses in May

Using 1st Word Plus	4th May
Using ViewStore	5th May
Software Compatability	6th May
Archimedes	9-11th May
Hardware Servicing	16-18th May
The Master and Compact	23rd May
Using View Professional	31st May

The venue for all courses is Acorn's Cambridge training centre. For more details, and course fees contact Val Raworth, Acorn Training, Acorn Computers, 645 Newmarket Road, Cambridge CB5 8PB, or ring (0223) 214411.

LAUNCHPAD INTO BUSINESS

Launchpad Commercial Researchers, the Newbury based business research company, can now supply extracts of their large database of UK companies for immediate use with ViewStore. The database covers premier companies, government departments, public authorities and financial institutions, with information on each including company location, annual revenue, number of employees and most importantly contact addresses.

Launchpad claims that this service will be invaluable to both business and education alike, but at a minimum charge of £200 for each disc of data it will only appeal to those with a real need. For details of this and related services contact: Launchpad Commercial Researchers, 25 Priory Road, Newbury, Berkshire RG14 7QS, or ring (0635) 45849.

LEARNING THE SOFT WAY

Soft-Teach Educational, specialists in curriculum based educational programs for both junior and secondary schools, has just published its third catalogue, Software for Schools 1988. All the software in the 24 page catalogue pays special attention to the demands of the new GCSE courses for secondary pupils. Network versions of most programs are available, and Soft-Teach offer both an approval service and bulk discounts. The company also supplies specialist hardware for use in physics experiments.

For a copy of the catalogue contact: Soft-Teach Educational, Sturgess Farmhouse, Longbridge Deverill, Warminster, Wiltshire BA12 7EA, or ring (0985) 40329.

TELETALK LAUNCH

Telemap, who will be best known to BEEBUG members for its Micronet service, has announced a change of name (to Telemap Group Ltd.), and launched a new service called *TeleTalk*. This is a real-time multi-user conferencing service which allows several people to chat to each other simultaneously, via viewdata frames. More details from: Telemap Group Ltd., Durrant House, 8 Herbal Hill, London EC1 5EJ, or ring 01-278 3143.

HOTEL CALIFORNIA

Also from Telemap comes a completely new viewdata service in the form of *'Hotel California'*. The 'hotel' contains all the services you would expect in any real hotel, including such wonders as a Casino with money prizes, a night porter who keeps horoscopes, an agony aunt, a cafe for chatting to other guests and even a shopping mall run by a major mail order firm.

The Hotel California, which is only for over 18s, can be accessed with any computer using a V23 (Prestel) standard modem and viewdata software. Needless to say this service doesn't come free, and all calls to the 'Hotel' are charged at the rather high rate of 38p a minute peak time and 25p a minute at other times.

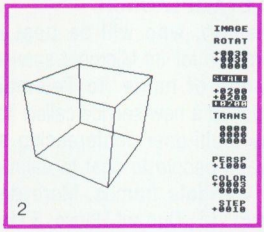
Hotel California is available on 0898 10 0890, or more information can be obtained from Telemap at the above address. B

REALTIME SOLIDS MODELLER

Geoff Bains looks at the new graphics package from Silicon Vision which takes us into the exciting world of solids modelling.

Product	**Realtime Solids Modeller**
Supplier	**Silicon Vision** **47 Dudley Gardens,** **Harrow HA2 0DQ.** **Tel. 01-422 2274**
Price	**£89.95** **£39.95 Realtime Graphics upgrade**

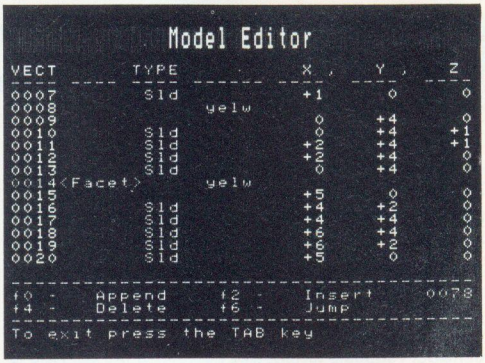
When BEEBUG reviewed the Realtime Graphics package, both in its original form from Glentop (Vol.4 No.5) and in its later reincarnation from Silicon Vision (Vol.5 No.10), it was difficult to see much in the way of possible improvements which could be made to the functions of the syst- em, although some of the ergonomics of it all were far from perfect. Then I compared the graphics produced by the system with Elite, for quality and speed - wireframe spacecraft flitting about the screen, and all from a Basic program anyone could write.



2

That was impressive then but now Elite has moved on to the Master version with solid spacecraft. Not to be outdone, Silicon Vision has released Realtime Solids which does all that the wireframe package did (and better) but with solid models, and it still only requires a model B (although it is quite happy with a B+ or Master).

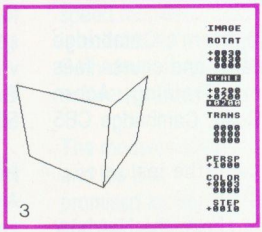
'Realtime Solids' takes all the hard work out of truly impressive graphics displays on your BBC micro. It provides all that is needed to create the mathematical models of complex solid shapes, draw them on the screen from any view



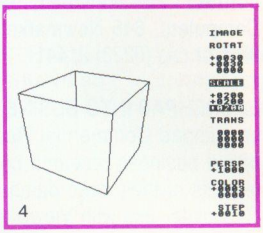
with full hidden line removal, output the view to a plotter and animate complex sequences.

The whole package is dependent on a 32K ROM containing all the routines for performing the co-ordinate transformations, and for the super-fast plotting required. The software also includes six discs containing the programs to create the models and demonstrations, and example models to play with.

Creating a model is a matter of mathematics. This is the hardest part as it requires you to visualise your object in terms of 3D co-ordinates. These



3



4

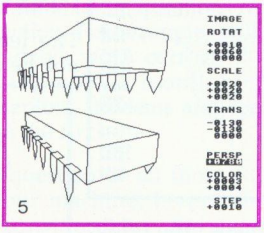
are entered into a kind of 'point processor' (see photo one) along with details of the the line type and colour, and whether the point is to be drawn or moved to.

That much is rather like the Realtime Graphics wireframe predecessor. However, the operation of the co-ordinate editor is much easier now with functions like point deletion and insertion available from the same screen. Also additional is the Facet command. This lets the system know that the following points make up the vertices of a flat side of the object. The colour for filling in areas can be separately defined and the face can be any shape.

The order of the vertices' co-ordinates determine from which side the face is visible. A face can only be seen from the side with the vertices appearing clockwise around its edge. This is the most difficult part of all to get to grips with and causes frequent (and sometimes unavoidable) problems with the final model.

The Realtime Solids package deals with three different views of models - wireframe, convex solid and concave solid. The wireframe model is familiar to all Beeb users. A wireframe cube is illustrated in photo two. This shows all the points and lines defined with no thought of real life possibilities. In other words you can see edges that are at the back of an object.

The convex solid view takes account of the opacity of the faces. Points and lines (or parts of lines) hidden behind a foreground face are not shown, nor are faces viewed from the 'wrong' side. A cube correctly defined, with vertices going clockwise around each face when viewed from the outside, is a convex solid - it has only an outside (as far as the model is concerned). However, remove the top face from this cube and the system has problems (photo three) - it cannot see the 'wrong' side (the inside of the back faces).



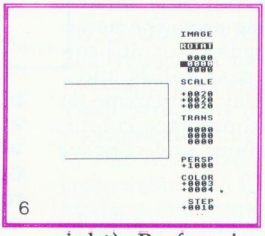
5

To cope with this, we display the open cube as a 'concave' solid. Now the system can 'see' both sides of each face and it interprets the open cube correctly (photo four). The problem with this concave view of models is that it takes longer to work out and to plot, something that matters when you come to animation. The same is true of any 3D solids system. All the spacecraft and planets in Elite are convex, closed solids for this very reason.

To get the views of complex concave models right a separate 'processing' facility is included in the Realtime Solids package. This orders the faces of a model in the systems own internal way to make sure it is drawn correctly and more quickly from then on. This enables spacially complex models like the chips in photo five to be drawn correctly (well, almost correctly). Even after processing some errors can occur - the front row of pins on the upper chip have no 'tops' as the system incorrectly thinks these are completely hidden).

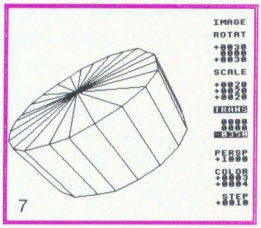
To make life simpler for creating complex models (and it certainly needs simplifying with all this to bear in mind), the system has some useful ways of building large complex models from small simple ones. A macro facility allows you to add one model to another having first rotated, translated or scaled it. In this way simple cube and pyramid building blocks become almost all you need.

Curved shapes and prisms can be created from even simpler primitives. The three sides-of-a-square in photo six can be rotated about the y-axis to form a disc (photo seven) with all the calculations and all the work done by the software. As well as the wireframe profiles of the old system, the Realtime Solids package can produce solids in this way.



6

Translate the primitive along the x-axis before rotation and you get an open centred torus (photo eight). Performing other transformations first gives you other final shapes.



7

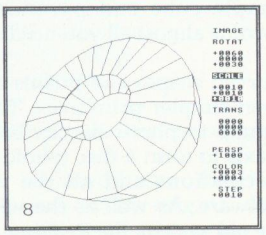
Another useful addition to the system is the 'extrusion' facility which creates a complex shape from a primitive by repeating it along an axis (photo nine). It's rather like squeezing icing through a primitive-shaped nozzle!

It doesn't take much imagination or effort to create extremely complex shapes from very simple beginnings. When you've mastered the ins and outs of your model, entered the primitives' co-ordinates and rotated and extruded them into the final object, you can save the model on disc and inspect all your handywork on the screen, viewed from any angle (it is this viewing mode which is shown in the photos). Different screen modes can be

used too, but the model B is limited to mode 4 and mode 5 (for colour). A Master or second processor is needed for modes 0-2.

These views can then be saved to disc as screen RAM dumps or as sequences of VDU codes. The VDU code files can be used to redraw the image on a machine without the Realtime Solids ROM or for feed- ing to a plotter.

This generalised 3D drawing and hidden line removing aspect of the Realtime Solids package is a feat in itself. However, the real strength (and the 'realtime' nature) of this software, like the Realtime Graphics System before it, lies in the Realtime Graphics Language (RGL) which forms the heart of the ROM.



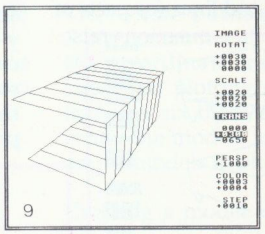
8

The ROM contains all the rou-tines for fast effective animation programming requiring only Basic programming skills from the user. The new RGL closely follows its predecessor. It has been extended to encompass solids and allows several types of display - wireframe (as before), convex solid without face colour, convex solid with face colour and concave solid.

Models are plotted on the screen by setting up the viewpoint with various Basic variables (xrot%, ytrn%, zscl% and so on) and calling the transformation and plotting routines in the RGL ROM. Speed is all important for animation of complex models such as these and the RGL is indeed fast. Although models such as the chalice take time to draw and so cannot be usefully animated, relatively simple objects such as Elite-type spacecraft can be realistically moved around the screen. A two-screen changeover process (all automated by the RGL) speeds the process considerably and removes all traces of flicker. More than one model moving at a time is quite possible as well, although too much overcrowding inevitably reduces the overall speed.

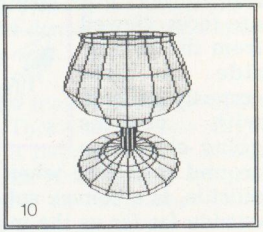
The new Realtime Solids system allows model data to be 'optimised' - sorted to remove repeated plots to any vertices - and a new 'turbo' command plots such optimised data even faster than the original software. Both new and old data formats, and new and old commands are in the new RGL for compatibility with old data and programs, but for new work the turbo option is well worth using.

Using the RGL is not easy. A whole host of variables have to be declared and zeroed, the right mode used for the size of model data table and for the machine, you must keep track of the memory usage the whole time and remember a whole host of commands. However, 23 example programs are given on disc and the easiest way to get to know the system is to mess around with these and adapt them to your own purposes. Indeed these programs will probably form the basis of all your future efforts.



9

The learning process is not helped much by the manuals. The original manual supplied with the Realtime Graphics system is reprinted with the new software (it's reduced in size to A5 size but is otherwise identical). The new aspects of the software and the new RGL commands added to extend the software to solids and hidden line removal are all covered in a separate manual. As the original manual was never too good in the first place, simply adding on like this is not a good idea, even though the extensions manual is much better written and produced. It is hard work to get to know the system.


10

However, getting to know the Realtime Solids package is certainly worth the effort. It is doubtful that any other software outside of games like Elite push the BBC micro to the same extremes of its capabilities. When you see a smoothly animated display of a solid object with full hidden line removal and hardly a trace of flicker, it is hard to believe that only a Beeb and a simple Basic program is behind it all. B

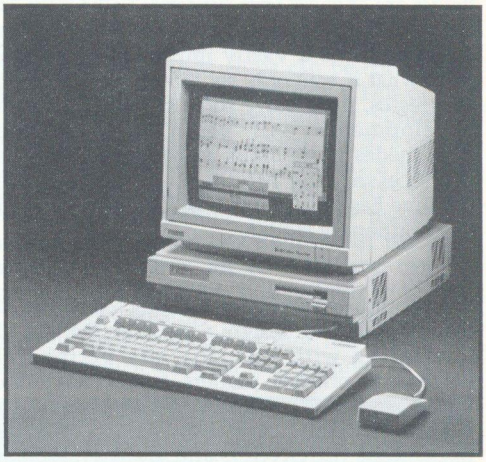
Arm Wrestling

Based on his own experiences, Chris Drage examines some of the issues to be faced in contemplating an upgrade to an Archimedes.

By now many BEEBUG readers will be weighing up whether or not to save their coppers, sell their faithful Beeb, and jump in at the deep end by buying an Archimedes. After all this hugely powerful machine is the obvious route to the future of micro computing, especially for existing Acorn users. Many readers, however, may well be a little perplexed in trying to answer some of the questions raised by this course of action. For example, will my trusty 5.25in disc drive work correctly? Can you use an existing colour monitor? Will existing teletext adaptors, modems and comms software work with the Arc?

Then there is the question of all those ROMs installed inside your machine. Investment in software can be considerable - can existing Basic and/or machine code programs be persuaded to run on the Archimedes? Questions like these are important to anyone considering the relative merits of whether to upgrade or not. Using my own experiences with an Archimedes A310M I will try to address some of these issues in this article and hopefully help readers to arrive at a decision. As to whether the Archimedes is a suitable upgrade path, and whether it will enjoy the longevity of the Model B, there are no obvious hard and fast answers - only time will tell.

The Archimedes is the first of a new generation of ARM computers. It represents a huge increase in power and performance compared with the 8-bit BBC/Master 128 or 16-bit IBM/MS-DOS computers. It will, however, be the software publishers as much as anyone who determine the success of the Archimedes. If Archimedes is equipped with extremely powerful, user-friendly software, then the machine's future is assured. Obviously such software takes time to develop. For example, Computer Concepts' word processing package by all accounts should provide a writing environment with desktop publishing facilities equalling and probably surpassing the 'professional' systems on the Mac and IBM compatibles. At this level it takes time to "get it right".



SOME OBSERVATIONS

Firstly, the A305 has limited memory and ought to be upgraded at some point to 1Mb of RAM. The restrictions imposed on many software packages currently available by the 0.5Mbytes of the A305 are quite significant (almost comparable with those imposed by the defunct Model A in its time). Thus if you are thinking of upgrading from your BBC B take note: think in terms of the A310 to avoid having to deconfigure every module etc from the new A305 machine in order to obtain enough free RAM for many applications.

Expanding the Archimedes to add any peripherals other than a printer or a modem does require the installation of an I/O podule and backplane. You will also need some form of interface if you wish to use a 5.25" disc drive with the Archimedes in order to transfer existing software.

The Desktop application is extremely fast, and if you are familiar with WIMP systems on other computers then you will appreciate the facilities offered by Arthur's window manager. However, the Desktop has many shortcomings. For example, the diary is only available in the current year, the clock has no alarm and the

calculator has no memory, square root function nor percent key! There are other disc-based desktops on offer (e.g. Brainsoft's), which are far superior.

To be fair, it must be said that some of these points are of short-term nuisance value only. In terms of software and hardware support, the Archimedes is a vastly improved computer from what it was only six month ago. As time progresses, third-party software and hardware developments will begin to push the Archimedes to its limits as happened with its predecessor, the Model B. At present the machine has only just reached the first rung of its evolutionary ladder.

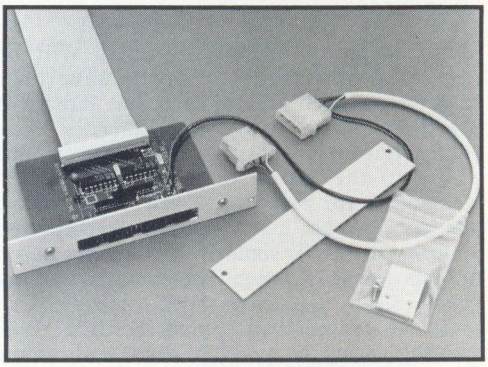
PERIPHERALS

The continued success of BBC micros is due to their open-ended design: a versatile method of interfacing with virtually any peripheral (keyboards, printers, plotters, teletext adaptors, modems etc). The Archimedes shares this design concept. For example, it can be networked (ECONET), and by a system of optional peripheral modules (podules), it can be made to interface with a similar variety of peripherals as the aged BBC B.

The problem I ran up against, having installed the I/O podule, was that neither my Acorn teletext adaptor nor my Viglen 28Mb hard disc can be persuaded to work. So don't expect all your add-ons to simply hook up to the Arc and run. Although attaching a modem represents few problems, at the time of writing we are still awaiting suitable comms software, and there are problems with the Archimedes' RS423 chip (Acorn has now produced a patch in the form of a relocatable module, which has been made available to RISC User members on their magazine disc for Vol.1 No.4).

Most printers, and the linear Graphics Plotmate (M series) plotter, present no more of a problem to use with an Archimedes than with the Model B. You will need to add a 25-way D-connector to your ribbon cable, though. The Plotmate also requires a different cable and a new ROM. The older Linear Graphics A4 plotter will not work with the Arc, however.

As for external disc drives, I successfully connected a Viglen, N.E.C. 5.25" disc drive to the Archimedes using BEEBUG's buffered 5.25" disc interface. Only fully-buffered interfaces such as this meet with Acorn's approval, and the 5.25" drive must also have its own power supply unit. The drive will then work quite happily. Remember to *CONFIGURE FLOPPIES 2 (or 3 if you attach a dual drive unit), to allow the Arc to recognise more than one physical floppy drive.



Beebug 5.25" Disc Interface

One of the advantages for adding a 5.25" drive is that IBM software can be run directly without the need to convert it to the 3.5" format. Many protected MS-DOS applications expect to run in drive A - the interface allows the external drive to be selected as this (a nice touch). In addition, 5.25" discs allow ADFS files to be ported directly between a Master 128 and the Arc. Remember though, that connecting a 5.25" floppy drive does not automatically mean that you can access BBC DFS discs. The Archimedes only supports the ADFS, though a conversion package from Resource does provide DFS emulation among other things.

As far as colour monitors are concerned, the Archimedes outputs an analogue RGB signal which is unacceptable to the TTL RGB monitors used with 8-bit Acorn micros. Some people have converted their existing Microvitec (and other) monitors but the message from every monitor manufacturer/supplier is clear: attempting to convert TTL RGB monitors internally is potentially dangerous, and the results will never compete with an analogue monitor. My advice is to forget it - be prepared to purchase the Archimedes' medium resolution monitor. Those readers owning a Philips 8833 (or older 8533), colour monitors are

better off as they already possess an analogue monitor comparable with that supplied with the Archimedes, and will only need to purchase a SKART monitor lead.

SOFTWARE

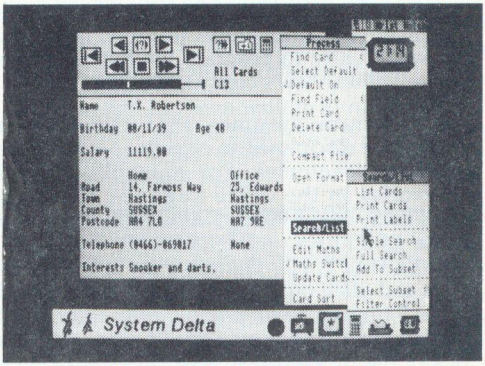
One of the first things you require after setting up the Archimedes is some means of getting your BBC/Master files onto the Arc's 3.5in ADFS discs. Several magazines have published file transfer articles but you must be prepared to spend an evening getting a cable made up and software typed in. For those who prefer the convenience of an off-the-shelf package, BEEBUG's Serial Transfer Kit is a worthwhile investment, as everything you require is included. Your BBC/Master becomes a terminal controlled by the Archimedes. DFS or ADFS 5.25" discs are catalogued, files marked and transfers automatically take place while you make a cup of coffee. Baud rates are also set from a window. Similar products are marketed by Brainsoft and Watford Electronics.

Many Basic programs developed on a Beeb will run directly on an Archimedes provided they make no direct references to memory (as in *LOAD for example). 6502 machine code programs will only run under the emulator, as the ARM processors obviously only recognises ARM machine code. Annoying problems do occur however, as I discovered in a program containing a number of *TAPE commands. These are not catered for by the emulator and as a result, crashed the program. Be prepared to do some editing.

ROM software on a Beeb often represents a large investment of cash, but in most instances such software will not work on the Archimedes, but then it may well not be applicable to the Archimedes anyway. Only correctly written ROM software (such as Acorn's View word processor) will run, and then only under the emulator (which emulates a 6502 second processor). For example, the Compact version of View (as a ROM image on disc) transfers correctly.

Most ROMs write to memory directly, and for this reason alone will not work on the Arc. The test is to try them on a BBC B with a second processor attached - if they run in the second processor, they should run in the 6502 emulator. Remember, too, that the Archimedes uses relocatable modules as an alternative to ROM-based software.

The addition of a ROM podule also represents an additional cost. However, the Computer Concepts' version utilises an extremely handy RAM Filing System (RFS), which when populated with RAM chips can be configured as a small silicon disc (up to 128K). This can be very useful for some applications. Similarly, when battery backing is added, the RFS can be booted on power-up to call into play a number of favourite routines stored in the RFS before returning the machine to ADFS - useful for setting up function key definitions, file path aliases etc. Thus an RFS library can be permanently installed. With regard to ROMs, Computer Concepts offer a £10 upgrade for each of their Inter series (not Interbase), for versions which work via their podule.



Minerva's System Delta Plus

The three packages I would choose to demonstrate the potential of the Archimedes are Minerva's System Delta Plus, Clare's Artisan and Resource's Desk Top Stories (for young children). One aspect they share is the glimpse they give us of future Archimedes' software. It took at least two years before software arrived which pushed the Model B to its limits - it will also take time for powerful Archimedes software to evolve. For evolve it must; already word processors like First Word Plus offer features guaranteed to astound the loyal Model B owner used to struggling with Wordwise.

Continued on page 49

AN ADFS MENU FOR VIEW

Effective and efficient management of View files is much easier to arrange on an ADFS format disc with this comprehensive utility by Alan Mothersole and Richard Beck.

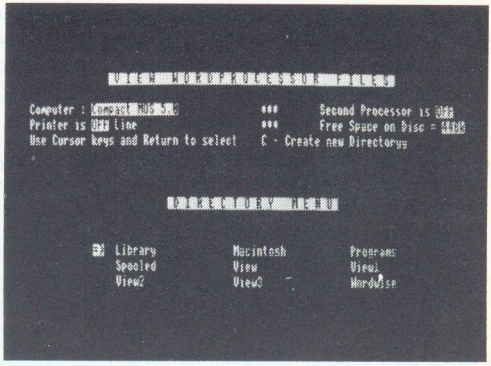
Managing large numbers of View files on an ADFS formatted disc can cause problems for the user. The purpose of the utility listed here is to provide such ADFS users with a practical means of organising their View text files on disc such that they can be readily accessed via a menu. This is made even easier as the utility allows a description of each file to be displayed in the menu rather than just a filename. The system is quite flexible and should cope with all user requirements. The utility also incorporates a number of further useful features, including the opportunity to program the function keys for use within View's command mode.

The program assumes that the user is sufficiently disciplined to organise his files into specific directories. For example, typical directories in the office might be **MEMOS, LETTERS, MINUTES, REPORTS**, etc, or at home **LETTERS, ADDRESSES, RECIPES, BOOKCHAPT**, etc. The program is written in Basic and after typing it in, it should be saved in the $ (root) directory on all your View word processing discs. You should also create a !BOOT file on each disc (use *BUILD, or on a Master use Edit), which chains the menu program.

HOW THE VIEW MENU WORKS

Booting the disc will load the program and display a list of directories on the disc, together with other information about your system (for more details on the coding for this refer to BEEBUG Vol.5 No.5). A cursor may be moved around the directories using the cursor keys, and any directory selected by pressing Return. The program then displays all the View files held in that directory together with a file description and the file length.

To provide the file description, the first line of each View file needs to begin with the command **CO** followed by a single line



description of the file. It is this statement that will be displayed by the menu program. Files other than View files should not normally reside in sub-directories. Any non-View files found will be flagged as unrecognised in the menu listing.

Use the up and down cursor keys to find the required View file, and then press Return to enter View with the selected text file loaded ready for editing. A maximum of ten files is displayed on the screen at any one time, but further files in a directory may be listed by using the up and down cursor keys to move beyond the bottom of the list, or return by moving above the top of the list. Pressing Shift and the cursor-up key together will return you to the directory display. You may then select an alternative directory if you wish.

In order to cater for varied user requirements, particularly in the form of more complex, hierarchical directory structures, pressing Shift-Return when a list of directories is displayed will then show any sub-directories. Again, you can move back up the hierarchy by pressing Shift/Cursor-Up. In addition, the Copy key may be used to toggle between a directory menu and a file menu at any level within an ADFS hierarchy (except within the root directory where only sub-directories are displayed). Most users will evolve a directory system to suit their needs and then stick with it, but the program is flexible enough to cope with most arrangements.

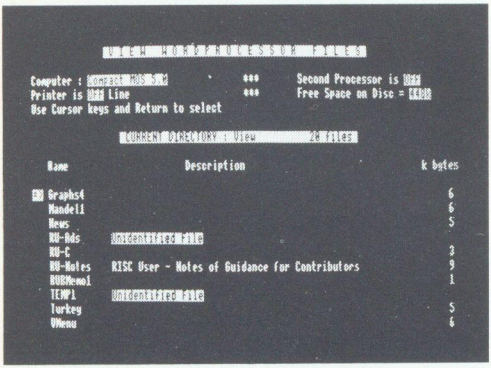
Pressing C (when a directory menu is displayed) allows the creation of a new directory. Once created the program then copies a file called **%Header** from the **$** directory to the new directory. This file must initially be created by the user and be resident

in the $ directory. It is simply a View file containing an often used View header format. When selecting the new directory from the menu this file will automatically be available in that directory.

All View printer and/or screen drivers should also be placed in the $ directory along with any other utilities. These will remain transparent to the user and will not clutter the appearance of the menu display.

The program allows the function keys to be pre-defined for use within View's command mode. Some typical definitions are included in PROCview (lines 2780 to 2880) but you may, of course, replace these with any definitions of your own (see BEEBUG Vol.4 Nos.9&10 for more information on this subject).

The number of files that can exist in a normal ADFS directory is 47. However, to allow the !BOOT, View-Menu, %Header and Printer Drivers to reside in the '$' directory the maximum number of directories is limited to 30. Of course, each directory can have up to 47 files in it. This should not present a problem in practice as the disc will probably become full before all directories and files are used.



SETTING UP THE VIEW MENU

Firstly, type in the program and save it as **$.View-Menu**. Secondly, create the !BOOT file (also in the $ directory) as follows:

```
*BUILD !BOOT
1 *BASIC
2 MODE3
3 *MOUNT0
4 CHAIN "View-Menu"
```

Press Escape and type *OPT4,3 <Return>. Thirdly, switch to View and create a suitable header file, such as the one shown below:

```
CO Header File
PL 66
TM 2
HM 2
DH/left/centre/right/
FM 2
BM 2
LM 2
DF/left/centre/right/
```

This should be saved as $.%Header.

Finally, copy any printer driver files onto this disc into the $ directory. Set up the printer driver filenames in the function key definitions in PROCview. If you use the Acornsoft screen driver as supplied with the View Printer Driver Generator, copy the files **TOSCRN** and **SCREEN** across into the $ directory. Then remove the REMs in lines 2850 and 2950.

The disc is now complete and should be booted up and used. View files on other discs or filing systems should be transferred across in the usual manner but remember they must reside in a named directory other than the $ directory and should have a first line beginning with the edit command **CO** and a single line description in order to be detected by the program.

The program has been written to run on Basic I. Basic II users may, if they wish, replace calls to PROCoscli with OSCLI and delete the definition of PROCoscli.

```
  10 REM Program ADFS View Menu
  20 REM Version B3.6
  30 REM Authors Alan Mothersole
  40 REM         and Richard Beck
  50 REM BEEBUG  April 1988
  60 REM Program subject to copyright
  70 :
 100 MODE 3:ON ERROR GOTO 3380
 110 PROCinit:PROCintro
 120 REPEAT
 130 PROCmp:PROCdm
 140 UNTIL FALSE
 150 END
 160 :
1000 DEFPROCinit
1010 VDU23,1,0;0;0;
1020 buffer%=&900
1030 DIM blk% 20,buf% 50,C% 18,S% 18
1040 DIM Dir$(31),name$(47),K% 100
1050 *FX18
1060 *FX12,0
1070 *FX4,1
1080 ENDPROC
1090 :
1100 DEFPROCintro
1110 char$=CHR$61+CHR$62
1120 CLS:PROCwind(1,TRUE)
1130 PROCh(" V I E W   W O R D P R O C
E S S O R   F I L E S ")
```

```
 1140 PRINT'"Computer : ";
 1150 RESTORE 3350
 1160 PROCcol(1):PRINT FNdisplay(INKEY(-
256));:PROCcol(2)
 1170 PRINTTAB(40)"***";SPC7;"Second Pro
cessor is ";:PROCcol(1)
 1180 IFFNtt tube$="ON" ELSE tube$="OFF"
 1190 PRINT tube$:PROCcol(2)
 1200 PRINT"Printer is ";:PROCcol(1)
 1210 IFFNpt PRINT"ON"; ELSE PRINT"OFF";
 1220 PROCcol(2):PRINT" Line";TAB(40)"**
*";SPC7;"Free Space on Disc = ";
 1230 PROCcol(1):PROCrsp:Sp$=RIGHT$((STR
$(Free%DIV1000)),3)
 1240 PRINTSp$;"k":PROCcol(2):PRINT"Use
Cursor keys and Return to select     C -
 Create new Directory":PROCwind(3,TRUE)
 1250 PRINT"Use Cursor keys and Return t
o select     C - Create new Directory"
 1260 PROCwind(3,TRUE):PROCwind(2,TRUE)
 1270 PRINT:PROCh(" Reading disc .......
 Please wait ")
 1280 ENDPROC
 1290 :
 1300 DEFPROCmp
 1310 FORX%=0TO550:buffer%?X%=0:NEXT
 1320 A%=&8:X%=blk%MOD256:Y%=blk% DIV256
 1330 ?blk%=0:blk%!1=buffer%:blk%!5=47:b
lk%!9=0
 1340 CALL &FFD1:E%=47
 1350 FORZ%=0TOE%-1:name$(Z%)=""
 1360 FORY%=1TO10
 1370 IFbuffer%?(Y%+11*Z%)>0 name$(Z%)=n
ame$(Z%)+CHR$(buffer%?(Y%+11*Z%))
 1380 NEXT:NEXT:P%=0:Q%=0:M=1
 1390 REPEAT
 1400 IFname$(P%)<>"" PROCfd
 1410 P%=P%+1
 1420 UNTIL name$(P%)="" OR P%=E%
 1430 E%=Q%:IF M>31 PROCerror(4)
 1440 ENDPROC
 1450 :
 1460 DEFPROCfd
 1470 ANS%=FNdir(name$(P%))
 1480 IF ANS%=2 Dir$(M)=name$(P%):M=M+1
ELSE name$(Q%)=name$(P%):Q%=Q%+1
 1490 IF M>31 PROCerror(4)
 1500 ENDPROC
 1510 :
 1520 DEFPROCdm:up=FALSE
 1530 PROCwind(1,FALSE):PRINTTAB(40,4)"C
 - Create new Directory"
 1540 PROCwind(2,TRUE):PRINT':PROCh(" D
I R E C T O R Y   M E N U "):PROCwind(3,
TRUE)
 1550 PROClist:x=11:y=1:oldx=11:oldy=1:P
ROCdrc(oldx,oldy):Select=FALSE
 1560 REPEAT
 1570 IF INKEY-106 THEN dflag=FALSE:UNTI
L TRUE:ENDPROC
 1580 G%=INKEY(0)
 1590 IFG%=136 x=oldx-20:PROCup1
 1600 IFG%=137 x=oldx+20:PROCup1
 1610 IFG%=138 y=oldy+1:PROCup1
 1620 IFG%=139 AND INKEY-1 THEN PROCoscl
i("DIR ^"):up=TRUE
 1630 IFG%=139 y=oldy-1:PROCup1
 1640 IFG%=13 Select=TRUE:dflag=INKEY-1:
PROCgo
 1650 UNTIL Select OR (G%AND&DF)=67 OR u
p
 1660 IF (G%AND&DF)=67 PROCcd:dflag=TRUE
 1670 IFSelect=TRUE PROCoscli("DIR "+Dir
$(Diroption))
 1680 IF up CLS:ENDPROC
 1690 PROCmp
 1700 REPEAT
 1710 IF dflag THEN PROCdm ELSE PROCdc
 1720 UNTIL up
 1730 ENDPROC
 1740 :
 1750 DEFPROClist
 1760 IF M>1 THEN PROClist2 ELSE PROCcol
(1):PRINTTAB(15,1)"No sub-directories":P
ROCcol(2):maxlist=1
 1770 ENDPROC
 1780 :
 1790 DEF PROClist2:X%=0
 1800 FOR Z%=1 TO M-1
 1810 IF Z% MOD3=1 THEN X%=X%+1
 1820 PRINTTAB(15+20*((Z%-1)MOD3),X%)Dir
$(Z%)
 1830 NEXT:maxlist=X%
 1840 ENDPROC
 1850 :
 1860 DEFPROCup1
 1870 IFx<11 x=11
 1880 IFx>51 x=51
 1890 IFy<1 y=1
 1900 IFy>maxlist y=maxlist
 1910 oldx=x:oldy=y:VDU8,8:PRINT"  ";:PR
OCdrc(oldx,oldy):ENDPROC
 1920 :
 1930 DEFPROCgo
 1940 IFoldx=11 Diroption=oldy*3-2
 1950 IFoldx=31 Diroption=oldy*3-1
 1960 IFoldx=51 Diroption=oldy*3
 1970 IF Diroption>M-1 Select=FALSE
 1980 ENDPROC
 1990 :
 2000 DEFPROCdc
 2010 @%=2:file%=0:E1%=E%-(E%=0)
 2020 PROCwind(1,FALSE):PRINTTAB(40,4)SP
C38
 2030 PROCwind(2,TRUE):PROCh(" CURRENT D
IRECTORY : "+Dir$(Diroption)+"     "+STR$
(E%)+" files "):PRINT:PRINTTAB(3)"Name";
SPC(22);"Description";SPC(33);"k bytes":
PROCwind(3,TRUE)
 2040 P%=10:PROCpr(P%):PROCdrc(0,0):oldy
=0:y=0
 2050 REPEAT:G%=INKEY(0)
 2060 IF INKEY-106 THEN dflag=TRUE
 2070 IFG%=138 y=oldy+1:PROCup2
 2080 IFG%=139 AND INKEY-1 THEN PROCoscl
i("DIR ^"):up=TRUE
 2090 IFG%=139 y=oldy-1:PROCup2
 2100 UNTILG%=13 OR dflag OR up
 2110 IF dflag OR up ENDPROC
 2120 file%=(P%-10+y):VDU26:CLS
 2130 PROCview
 2140 ENDPROC
 2150 :
 2160 DEFPROCpr(P%)
```

```
 2170 IF E%=0 PROCcol(1):PRINTTAB(3)"No
files":PROCcol(2):Z%=1:ENDPROC
 2180 Z%=P%-10:REPEAT
 2190 N%=OPENINname$(Z%):F%=BGET#N%
 2200 IFF%=128 PROCshow ELSE PROCshow1
 2210 CLOSE#N%:UNTIL Z%=P% OR Z%=E%
 2220 ENDPROC
 2230 :
 2240 DEFPROCshow
 2250 PRINTTAB(3)name$(Z%)TAB(15);
 2260 F1%=BGET#N%:F2%=BGET#N%:IF CHR$(F1
%)+CHR$(F2%)="CO" THEN PROCdisplay1
 2270 PRINTTAB(77)(EXT#N%+500)DIV1000:Z%
=Z%+1
 2280 ENDPROC
 2290 :
 2300 DEF PROCdisplay1
 2310 FOR C%=1 TO 63:F%=BGET#N%
 2320 IF F%=13 C%=64 ELSE PRINTCHR$(F%);
 2330 NEXT:ENDPROC
 2340 :
 2350 DEFPROCshow1
 2360 PRINTTAB(3)name$(Z%)TAB(15);:PROCc
ol(1):PRINT"Unidentified File";:PROCcol(
2)
 2370 Z%=Z%+1:ENDPROC
 2380 :
 2390 DEFPROCup2:LOCAL J%
 2400 IFy<1 AND P%=10 y=0:oldy=0
 2410 IFy>(E1%-P%+9) y=E1%-P%+9
 2420 FORJ%=10TO40 STEP 10
 2430 IFy<0 AND P%=(J%+10) y=9:CLS:PROCp
r(J%):P%=J%:PRINT
 2440 IFy>9 AND P%=J% y=0:CLS:PROCpr(J%+
10):P%=J%+10:PRINT
 2450 NEXT:oldy=y:VDU8,8:PRINT"  ";:PROC
drc(0,y)
 2460 ENDPROC
 2470 :
 2480 DEFPROCcd
 2490 IFM>29 PROCerror(5):ENDPROC
 2500 PROCgnn:IFnewname$="":ENDPROC
 2510 REPEAT:ok=TRUE
 2520 IFLEN(newname$)>10 PROCerror(1):ok
=FALSE
 2530 IF ok PROCas:PROCcaf ELSE PROCgnn
 2540 UNTIL ok
 2550 ENDPROC
 2560 :
 2570 DEFPROCgnn
 2580 PROCwind(2,TRUE):PROCh(" C R E A T
 E   N E W   D I R E C T O R Y "):PROCwi
nd(3,TRUE)
 2590 INPUTTAB(26)"Enter name of directo
ry: " newname$
 2600 ENDPROC
 2610 :
 2620 DEFPROCas
 2630 IFLEN(newname$)<10 REPEAT newname$
=newname$+" ":UNTILLEN(newname$)=10
 2640 ENDPROC
 2650 :
 2660 DEFPROCcaf
 2670 exist=FALSE:PROCmp
 2680 FORX=0TOE%-1
 2690 IFnewname$=name$(X) exist=TRUE
 2700 NEXT
 2710 IFexist=TRUE PROCerror(2):ENDPROC
 2720 PRINT':PROCh("Creating "+newname$+
" Directory"):PROCoscli("CDIR "+newname$
):PRINT':PROCh("Copying %HEADER from $ t
o "+newname$+" Directory"):PROCoscli("CO
PY "+"$.%HEADER"+" "+newname$)
 2730 ENDPROC
 2740 :
 2750 DEFPROCview
 2760 REM View function key definitions
to be set by user
 2770 FORy%=&B00TO&BFFSTEP4:!y%=&1010101
0:NEXT
 2780 PROCoscli("KEY9L "+name$(file%)+"|
MMODE3|MSETUP F|M*KEY9|M")
 2790 PROCoscli("KEY0 SCREEN|M")
 2800 PROCoscli("KEY1 SHEETS|M")
 2810 PROCoscli("KEY2 *CAT|M")
 2820 PROCoscli("KEY3 :|M")
 2830 PROCoscli("KEY4 :|M")
 2840 PROCoscli("KEY5 :|M")
 2850 REM PROCoscli("KEY6 PRINTER SCREEN
|M")
 2860 PROCoscli("KEY7 PRINTER $.EPSON|M"
)
 2870 PROCoscli("KEY8 :|M")
 2880 *KEY10OLD|MMODE3|M
 2890 REM Force required file to be load
ed :
 2900 *FX138,0,137
 2910 REM Force PRINTER driver on key 7
to be loaded
 2920 *FX138,0,135
 2930 *FX12,0
 2940 *FX202,48
 2950 REM *$.TOSCRN
 2960 *WORD
 2970 ENDPROC
 2980 :
 2990 DEFPROCh(H$):PRINTTAB((39-(LEN(H$)
)/2));:PROCcol(1):PRINTH$:PROCcol(2):END
PROC
 3000 :
 3010 DEFPROCdel(time):TIME=0:REPEAT UNT
ILTIME>=time:ENDPROC
 3020 :
 3030 DEFPROCdrc(x,y):PROCcol(1):PRINTTA
B(x,y)char$;:PROCcol(2):ENDPROC
 3040 :
 3050 DEFPROCcol(c)
 3060 IFc=1 COLOUR0:COLOUR129
 3070 IFc=2 COLOUR1:COLOUR128
 3080 ENDPROC
 3090 :
 3100 DEFPROCwind(w,clear)
 3110 IFw=1 VDU28,0,6,79,0
 3120 IFw=2 VDU28,0,10,79,6
 3130 IFw=3 VDU28,0,22,79,10
 3140 IFw=4 VDU28,0,23,79,22
 3150 IFclear CLS
 3160 ENDPROC
 3170 :
 3180 DEFPROCerror(e%)
 3190 PROCwind(4,TRUE)
 3200 IFe%=1 PROCh(" NAME TOO LONG! ")
 3210 IFe%=2 PROCh(" DIRECTORY EXISTS! "
)
```

Continued on page 21

RESISTOR CALCULATIONS

Brian Dean describes a short program designed to help anyone who indulges in electronics as a hobby.

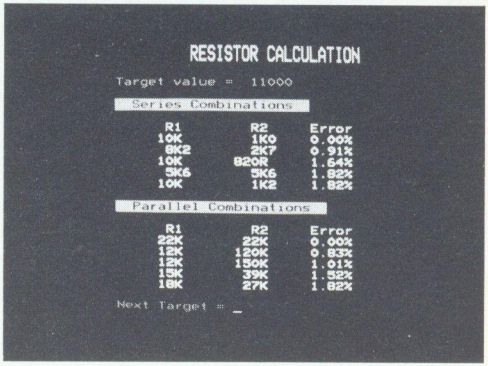
If you are interested in electronics then you will know that resistors are manufactured in a series of 'preferred values', each value being about 20 per cent greater than its predecessor in the series. A decade of resistance, such as the range from 100 ohms to 1000 ohms, is covered by twelve preferred values. These provide an adequate choice for most circuit designs, but sometimes more accurate values are required, especially when designing test equipment.

Not many people realise that almost any resistance value can be achieved quite accurately using a combination of only TWO preferred values. Often a parallel combination gives the best approximation, but it would be very tedious to calculate by hand the many dozens of possible combinations to discover the best one. This program does this for you, listing five series combinations and five parallel combinations of resistors, and indicating how close they are to the desired target value. By the way, if you are building a circuit and find that you are short of one or two resistors, this program can also help you to find pairs of values that may be used as alternatives.

The program is very simple to use. Just type it in and save it first. When run the program asks for a target value, and then lists the five best series and parallel combinations to achieve the desired result. Further target values may then be tried in turn. Pressing Return in response to the prompt to enter a new target value will exit from the program.

PROGRAM NOTES

As with many simple programs, a large part of the coding is devoted to error trapping and presentation of the results. The main routines are PROCseries and PROCparallel, which generate in turn a series of likely pairs of values (r1 and r2). PROCoffer compares these pairs with a table of previous pairs, retaining the best five. FNup(R) and FNdown(R) return the next preferred value above or below R, respectively.



The variable top, defined in line 1960, sets the highest value resistor that is to be used in generating the combinations. It is currently set to ten megohms. The lowest value used is ten ohms, and cannot be changed. The program checks the target value entered, and if outside this range seeks a revised target value. Series values are not calculated for target values less than 30, and parallel combinations are not calculated for targets exceeding one fifth of top.

```
   10 REM Program Resistor Calculator
   20 REM Version B1.3
   30 REM Author  Brian Dean
   40 REM BEEBUG  April 1988
   50 REM Program subject to copyright
   60 :
  100 MODE 7:ON ERROR GOTO 190
  110 PROCinit:PROCreset:PROCtitle
  120 REPEAT:CLS:PRINT'CHR$130;"Target v
alue = ";:IF target>0 PRINT target
  130 IF target<10 OR target>top REPEAT:
PRINTTAB(17,1)SPC10;:INPUTTAB(17,1) targ
et:UNTIL target >=10 AND target <=top
  140 PROCseries:PROCreset:PROCparallel:
PROCreset:PROCwarn
  150 PRINT'CHR$130;:INPUT"Next Target =
 "target
  160 UNTIL target=0
  170 MODE7:@%=&90A
  180 :
  190 ON ERROR OFF:MODE 7:REPORT
  200 PRINT" at line ";ERL:@%=&90A:END
  210 :
 1000 DEF PROCtitle
 1010 FOR I=0 TO 1:PRINTTAB(6,I)CHR$129;
CHR$157CHR$131;CHR$141;"RESISTOR CALCULA
TION   "CHR$156:NEXT
 1020 VDU28,0,24,39,2
 1030 ENDPROC
 1040 :
 1050 DEF PROCseries
 1060 series=TRUE
 1070 IF target<30 PRINT''CHR$131"Series
```

```
 combinations suppressed"'CHR$131"for va
lues under 30":ENDPROC
 1080 r1=FNup(target/2.1):REPEAT r=targe
t-r1
 1090 IF FNpref(r) THEN r2=r:PROCoffer
 1100 r2=FNup(r):PROCoffer:r2=FNdown(r):
PROCoffer
 1110 r1=FNup(r1):UNTIL r1>=target:r2=0:
PROCoffer
 1120 PROClist("Series")
 1130 ENDPROC
 1140 :
 1150 DEF PROCparallel
 1160 series=FALSE
 1170 IF target>top/5 PRINT''CHR$131"Par
allel combinations suppressed"'CHR$131"f
or values above ";top/5':ENDPROC
 1180 r1=FNdown(2.1*target):REPEAT r=(ta
rget*r1)/(r1-target)
 1190 IF FNpref(r) THEN r2=r:PROCoffer
 1200 r2=FNup(r):PROCoffer:r2=FNdown(r):
PROCoffer
 1210 r1=FNdown(r1):UNTILr1<=target
 1220 PROClist("Parallel")
 1230 ENDPROC
 1240 :
 1250 DEF PROCoffer
 1260 REM Try r1/r2 against previous bes
t
 1270 IF series THEN error=FNseries ELSE
 error=FNparallel
 1280 IF error>err(4) THEN ENDPROC
 1290 val1(4)=r1:val2(4)=r2:err(4)=error
 1300 FOR I%=4 TO 1 STEP-1:J%=I%-1
 1310 IF err(I%)<err(J%) THEN PROCswap
 1320 NEXT
 1330 ENDPROC
 1340 :
 1350 DEF PROCswap
 1360 K=val1(I%):val1(I%)=val1(J%):val1(
J%)=K
 1370 K=val2(I%):val2(I%)=val2(J%):val2(
J%)=K
 1380 K=err(I%):err(I%)=err(J%):err(J%)=
K
 1390 ENDPROC
 1400 :
 1410 DEF FNdecade(R):LOCAL D%,K
 1420 D%=-1:K=1
 1430 REPEAT K=K*10:D%=D%+1:UNTILK>R
 1440 =D% :REM Returns exponent of R
 1450 :
 1460 DEF FNup(R):LOCAL I%,M%
 1470 IF R<10 THEN=10 ELSE IF R>top THEN
=top
 1480 M%=10^(FNdecade(R)-1)
 1490 I%=0:REPEAT I%=I%+1:UNTIL pref(I%)
*M%>R
 1500 =pref(I%)*M%
 1510 :
 1520 DEF FNdown(R):LOCAL I%,M%
 1530 IF R<=10 THEN=0 ELSE IF R>top THEN
=top
 1540 M%=10^(FNdecade(R)-1):IF R=10*M% T
HEN=82*M%/10
 1550 I%=N%:REPEAT I%=I%-1:UNTIL pref(I%
)*M%<R
 1560 =pref(I%)*M%
 1570 :
 1580 DEF FNseries=ABS(target-r1-r2)
 1590 :
 1600 DEF FNparallel=ABS(target-(r1*r2)/
(r1+r2))
 1610 :
 1620 DEF FNpref(R):LOCAL I%,K%,M
 1630 K%=FALSE:M=10^(FNdecade(R)-1):IF R
<10 THEN=FALSE
 1640 I%=-1:REPEAT I%=I%+1
 1650 IF pref(I%)*M=R THEN K%=TRUE
 1660 UNTIL K% OR pref(I%)*M>R:=K%
 1670 :
 1680 DEF PROClist(set$)
 1690 PRINT'CHR$134;CHR$157;CHR$132;set$
;" Combinations  "CHR$156
 1700 PRINT'CHR$131;SPC6"R1"SPC8"R2"SPC5
"Error  "
 1710 FORI%=0TO4
 1720 PROCformat(val1(I%)):PROCformat(va
l2(I%))
 1730 error=err(I%)/target
 1740 @%=&20208:PRINT error*100;"%"
 1750 NEXT:@%=&A0A
 1760 ENDPROC
 1770 :
 1780 DEF PROCformat(R)
 1790 @%=&808
 1800 IF R<1000 PRINT R;"R ";:ENDPROC
 1810 IF R<10000 PRINT R DIV1000;"K";R M
OD1000/100;:ENDPROC
 1820 IF R<1000000 PRINT R DIV1000;"K ";
:ENDPROC
 1830 PRINT R DIV1000000;"M";R MOD100000
0/100000;
 1840 ENDPROC
 1850 :
 1860 DEF PROCwarn
 1870 IF FNpref(target) PRINT CHR$136"No
te : ";target;" is a Preferred Value!"
 1880 ENDPROC
 1890 :
 1900 DEF PROCreset
 1910 FOR I%=0 TO 4:err(I%)=top
 1920 NEXT
 1930 ENDPROC
 1940 :
 1950 DEF PROCinit
 1960 N%=12:top=10E6:target=0
 1970 DIM pref(N%),val1(4),val2(4),err(4
)
 198J FOR I%=0 TO N%:READ pref(I%):NEXT
 1990 ENDPROC
 2000 DATA 10,12,15,18,22,27,33,39,47,56
,68,82,100
```



STRINGS 'n' THINGS

Dec McSweeney continues his introduction to programming in C with an examination of character and string handling.

In part one, we looked at some aspects of C from a Basic programmer's point of view. Now we turn to an area where C is apparently deficient, namely string handling. There are no facilities in C for dealing with strings! This is not the tragedy it may appear. In fact, a string is treated as an array of characters, and may be manipulated with standard library routines - or you can write your own. The declaration of an array resembles the Basic DIM statement:

```
int narray[4];   /* 4 numbers (0-3)*/ char
s[40];      /* 40 characters */
```

This differs from Basic in two ways, namely the use of *square* brackets and the number of elements defined. In Basic, **DIM narray(4)** creates an array of 5 numbers, **narray(0)** to **narray(4)**. In C you get what you ask for - **int narray[4]** gives you four elements, **narray[0]** to **narray[3]**. Here, for example, is a function to accept a string of characters from the keyboard:

```
getstring(str)
char str[];
{
int n=0;
char c;
while((c=getchar()) != '\n'){
  str[n++] = c;
  }
  str[n] = '\0';
  return(n);
}
```

Several points arise here. The array, defined in the second line, has no size. This is because the size will be defined in the calling function. As an argument to the function, it is defined outside the braces.

Within the braces, a subscript n is defined and initialised to zero. The work is done within a **while** loop, which contains a call to **getchar()**. The sequence of events here is:

1. Evaluate the expression **c=getchar()**, i.e. fetch a character from the input stream and put it in **c**.

2. Compare it to '\n', the newline (Return) character (the **!=** means 'not equal').

3. If true (that is, if the character is not a newline), store the character in the nth element of **str** and then add 1 to **n** (postfix increment). As in Basic, element numbering starts at zero. In C, however, the last element of **str[n]** is the nth element, subscript n-1.

4. When the condition is false, that is when **c** is equal to '\n', store '\0' (a null byte, containing binary zero) after the last character and exit, returning the length of the accepted string.

The use of '\0' as a string terminator is standard in C, whereas in Basic a string is preceded by a byte containing its length. The function **printf**, for example, looks for a null byte as a string terminator. The **getstring** function will store the entered string in the named array, and the returned value is the length of the string.

The string is stored in an array defined by the call to the function. Hence, the main function might look like this:

```
/* First example */
/* Routine to accept a string from the
keyboard */
#include <h.stdio>
/* (required for getchar and printf) */
main(){
int length;
char message[80];
printf("You type it, I'll count it!\n");
length=getstring(message);
printf("Wow! %d characters!\n",length);
}
/* getstring follows here */
```

If you have access to a C compiler, try this out. You'll soon see that Deletes count in the computation, because we made no provision for them in **getstring**. You might also like to try entering more than 80 characters - this would cause an error in an equivalent Basic program, but the C compiler trusts you to handle your subscripts properly - don't let it down!

To cater for these problems, we must enlarge **getstring**. The Deletes are easily handled with an **if-else** construction in the **while** loop:

```
if(c == '\b')    /* \b = backspace  */
  if(n > 0)      /* test n - do not */
    n--;         /* decrement if zero */
  else
    n = 0; else
  str[n++] = c;
```

Note the use of "==" to test equality *(IS equal to)*, as opposed to the assignment operator "=" *(BECOMES equal to)*. Note also the nested **if** statements. Eat your hearts out, Basic programmers!

Keeping the subscript in range is more tricky. If **getstring** is to be of general use, the limit of 80 should not be imposed on each call. Adding another argument to the function, that of maximum length, is a better solution. The call might look like this:

```
;
int slimit = 80;
...
length = getstring(message,slimit);
```

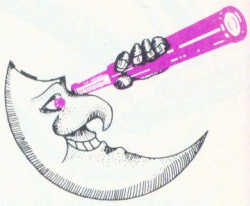
and the function declaration like this:

```
getstring(str,maxlen)
```

with another argument, **maxlen**, declared beneath that of **str**. Our **while** line changes to:

```
while((c=getchar()) != '\n' && maxlen-- > 0)
```

Here we decrement maxlen on each pass after testing it. The double ampersand (&&) is the logical AND, by the way. This version of **getstring** appears in full towards the end of this article.



We have now written a useful function al-, though a library routine **gets()** already exists to do much the same thing. Never mind.Last month, I mentioned the "call by value" nature of C functions. This means that they can't affect variables outside the function itself, which allows us to decrement **maxlen** with- out affecting **slimit**. So what happens to our input? The answer lies in the nature of the arguments. When we declare an array such as **message[80]**, we are reserving memory at an address known to the compiler.

Unlike Basic, the C compiler will happily give you this address. The effect of an array declaration is much more like the Basic **DIM X% 80** (where 81 bytes are reserved, the address of the first being stored in X%) rather than Basic's **DIM A$(80)**, where the Basic interpreter handles all the sordid details. In our example, **message[0]** is the first character of the array/string, **message[1]** the second and so on. The identifier **message** on its own denotes the address of the array. In fact, **message+1** is an expression which evaluates as the address of the second character, **message+2** the third, and so on. The function receives a copy of the array's address, and 'pokes' the characters one by one into **message.**

Variables which hold the address of another variable are called pointers. Array names are an example, but you can also explicitly define a pointer, and assign a variable's address to it:

```
int number;  /* declares an integer */
int *pnum;   /* declares a pointer */

pnum = &number;
          /* pnum points to number */
num2 = *pnum;
 /* contents at pnum stored in num2 */
```

Here we see the unary operators * and & at work. In the declaration **int *pnum;** the asterisk specifies that this variable is a pointer to integers. Note that a pointer may only be used for variables of its own type. In the assignment statements, '&' denotes 'the address



of...' and '*' denotes 'the contents of the address...'. Those of you who have tackled assembler programming will appreciate the difference between an address and the contents of an address! In fact, C is often referred to as a high-level assembler language, and the capacity to manipulate addresses is one reason.

So we can now modify variables from within a function, using pointers. The main strength of pointers, however, lies in handling arrays. Wherever a subscript may occur, a pointer can be used, and often does the job better, particularly in terms of producing efficient object code. Multi-dimensional arrays are often used in programming, and you can have them in C:

```
int coords [25] [2];
char month_names [12] [9];
```

The most obvious new feature here is that each subscript has its own pair of brackets. This is consistent with C's treatment of arrays. In the same way that **message** in our first example is the address of the character array, **month names[1]** is the address of the second element - in this case a nine-character object. Using both subscripts specifies a character, using only the first specifies a group of characters. **month names[n]** is a pointer. Let us illustrate an area where pointers help solve the Beeb owner's perennial problem - lack of memory.

Let's say that you want your computer to recite a limerick. Easy enough, just enter the five lines into an array using **getstring()**. This might do:

```
Instant poetry */
main(){
char line[5][50];
        /* 5 lines of 50 characters */
int n;
int slimit;
for(n=0; n < 5; n++)
   getstring(line[n],slimit);
for(n=0; n < 5; n++)
   printf("%s\n",line[n]);
}
```

Note here the use of **line[n]** as a pointer to the required line. If you like, a two-dimensional character array in C is similar to a one dimensional string array in Basic.

Each element will have to be the same length, that of the longest line. This is wasteful considering that line lengths can vary widely within a limerick. To save space we can have a one-dimensional array, and a pointer to the beginning of each line - an array of pointers!

```
/* I've never heard verse */
/* Entering & printing limericks */
#include <h.stdio>
main(){
char line[160];
char *pline[5];
int n;
int length;
for(n=0; n<5; n++){
  pline[n] = (n > 0) ? (pline[n-1] +
             length) : line;
  length = getstring(pline[n],50);
  }

/*   now we print the limerick  */

for(n=0; n<5; n++)
 printf("%s \n",pline[n]);
}

/* the final version of getstring */

getstring(str,max)
char str[];
int max;
{
int n=0;
char c;
while((c=getchar()) != '\n' && max-- > 0){
 if(c == '\b')
    if(n > 0)
      n--;
    else
      n = 0;
  else
      str[n++] = c;
  }
  str[n] = '\0';
  return(n); }
```

In **main**, we have a **for** loop to control the data entry. The first statement in the loop may look confusing, but demonstrates another useful conditional expression. It is the ternary

operator '?:'. The condition on the left of the ? (n > 0) is evaluated. If it is true, the value of the expression is that on the left of the colon, otherwise it is the value of the expression on the right. It is essentially an IF-ELSE statement:

```
10 IF A > B THEN RESULT = 2*B ELSE RESULT
= 5*A
```

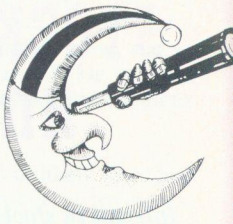
would become in C:

```
result = (a > b) ? 2 * b : 5 * a
```

A crucial feature of this operator is that the unused expression is not evaluated. This can be important in more complex situations.

In our example we use it to assign the correct value to our pointer. In the first pass when **n**=0, **pline[0]** must be set to the address of the array line. In subsequent passes, the pointer is set to the value of the previous element in the pointer array, plus the length of the last line entered. Obviously when **n** is zero, **pline[n-1]** is not the value we want. The value of **length** is also not correct at this point.

The only problem with our program is its lack of control over the array. What happens when our input adds up to more than 160 characters?



The compiler will cheerfully allow us to corrupt all our RAM by using out-of-range subscripts, so we should put in a test for 'end-of-array'. The most straightforward method may be to adjust our maximum line length in the call to **getstring**. The remaining space in our array is given by:

```
160 - (value of current pointer) - (value
of pointer[0])
```

So we rewrite the call to **getstring** as:

```
length = getstring(pline[n],
     (160-pline[n]-line));
```

A useful aspect of the array/pointer relationship is that this sort of computation gives a result in terms of the element size, rather than in bytes. This will be more important later on.

More about pointers next time, when with the help of a few words about file handling, I hope to present an elementary text processor, suitable for entering C programs.

B

AN ADFS MENU FOR VIEW- Continued from page 15

```
 3220 IFe%=3 PROCh(" DIRECTORY DOES NOT
EXIST! ")
 3230 IFe%=4 PROCh(" WARNING! This disc
has more than 30 directories ")
 3240 IFe%=5 Error$="No more than 30 dir
ectories allowed!!"
 3250 SOUND1,-15,70,10:PROCdel(150):CLS
 3260 ENDPROC
 3270 :
 3280 DEFPROCoscli(line$):$buf%=line$:X%
=buf%:Y%=buf%DIV256:CALL &FFF7:ENDPROC
 3290 DEFPROCrsp:A%=&71:X%=K%MOD256:Y%=K
%DIV256:CALL&FFF1:Free%=K%!0:ENDPROC
 3300 DEFFNdir(Z$):$S%=Z$:?C%=S%MOD256:C
%?1=S%DIV256:A%=5:X%=C%MOD256:Y%=C%DIV25
6:=(USR(&FFDD) AND &FF)
 3310 DEFFNdisplay(Z%):REPEAT:READ W%,W$
:UNTILW%=Z%ORW%=1000:=W$
 3320 DEFFNtt:A%=&EA:X%=0:Y%=&FF:=USR(&F
FF4)AND&FF00
 3330 DEFFNpt:VDU2,1,0,1,0,1,0,1,0,1,0,3
:=(ADVAL(-4)-59)
 3340 :
```

```
 3350 DATA -1,BBC Micro MOS1.0/1.2,1,Aco
rn Electron,245,Compact MOS 5.0
 3360 DATA 251,BBC B+ MOS 2.0,253,Master
 Series (UK),1000,Unrecognised System
 3370 :
 3380 IFERR<>17 PROCcrash:END
 3390 IFERR=17 PROCwind(4,TRUE)
 3400 PRINTTAB(24);:PROCcol(1):PRINT" Do
 you wish to Quit (Y/N) ";:PROCcol(2)
 3410 Z$=GET$:IF Z$="Y" OR Z$="y" PROCqu
it:END
 3420 PROCoscli("DIR"):CLEAR:GOTO 100
 3430 :
 3440 DEFPROCquit:VDU26,12:PROCoscli("DI
R"):PROCoscli("FX4 0"):PROCh(" View Menu
 Program Ended "):PROCh(" Current Direct
ory $  -  !BOOT to Re-run Program "):END
:ENDPROC
 3450 :
 3460 DEFPROCcrash:VDU26,12:REPORT:PRINT
" at line ";ERL:CLOSE#0:PROCoscli("DIR")
:PROCoscli("FX4 0"):ENDPROC
```

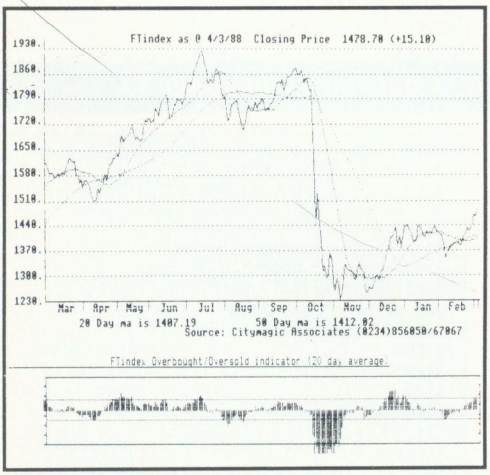
B

STOCK MARKET ANALYSIS

Interest in stocks and shares has increased dramatically in recent years. Mark Jolliffe, who has long taken a keen amateur interest in this subject, looks at two very different packages to assist the budding wheeler and dealer.

Product	Sharematic II
Supplier	Citymagic Associates 40 Manor Road, Bedford MK41 9LQ. Tel. (0234) 856 050
Price	£35.00 inc. VAT

Sharematic II is a package for the Master 128 equipped with a Morley or Acorn Teletext adaptor and twin double-sided, double-density disc drives. A printer is useful, but not essential.



The purpose of the package is to download London Stock Exchange share prices from the BBC Ceefax service, to store these and past prices, and provide various graphs and statistical reports. The package comes as two ADFS discs: one contains the various programs and utilities, and the other contains 450 days of the previously downloaded prices of the 168 shares plus the FT and FTSE indices (as up-to-date as direct mailing will allow). Old prices are removed as new prices are added, preserving the 500 day history. An instruction manual is also provided, both on the program disc, and as a booklet.

Price histories can be displayed or printed out in table form, or as graphs and charts covering any period of between 5 and 450 days during the last 500 days. One option provides a linear graph with 20 and 30 day moving average overlays, an overbought/oversold indicator, and a point and figure chart. A second option provides a similar linear graph, but overlaid with a 200 day moving average. The printout here is only available using a Printmaster ROM, giving a large graph. There is no facility to alter y axis scaling or produce logarithmic graphs.

A daily report gives various price movement indicators for all shares, with a buy, hold, or sell recommendation. The indicators are based upon the relationships between the 20, 30, and 200 day moving averages.

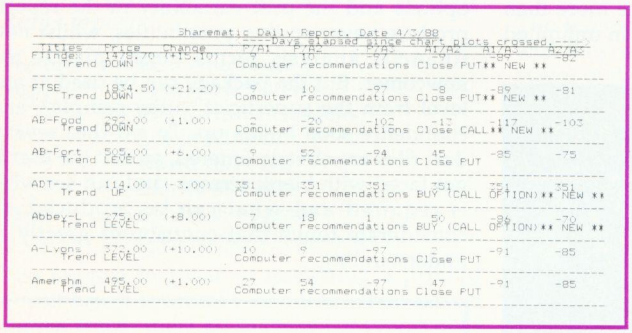
The programs are entirely menu-driven, and generally operate in mode 7. In theory, operation of the package is simple. The program disc is placed in drive 0, and the data disc in Drive 1. Booting-up the system results in the 170 current prices being downloaded from Ceefax, checked, and written to the data disc.

Good reception is clearly essential. The database must also be up-to-date with the previous day's prices (or Friday's prices for Monday's downloading), and the program's knowledge of company names must coincide with those in current use on Ceefax. All of this can only too easily lead to problems and frustration. However, you can always enter absent prices manually.

If reception is not perfect, then the data will not be downloaded. If all attempts to download fail, then the next day you will be back to square one: no up-to-date data, no automatic download.

After downloading, the program checks share names on Ceefax with those existing on file.

Mismatches can occur when a name changes, or when one company is substituted for another. An example of the former was when "GUS A" became "GUS 'A'", and of the latter when "Freeman" was replaced by "Eurotun". A utility is supplied to rename or replace the mismatches.



Sharemetic Daily Report. Date 4/3/88

Titles	Price	Change	Days Elapsed since chart plots crossed				
FTSE	1878.70	(+10.10)	6701	102	-6427	A1702A	A1702	A2702	-82
Trend DOWN			Computer recommendations Close PUT** NEW **				
FTSE	1874.50	(+21.20)	-5	10	-97	-8	-89	-81
Trend DOWN			Computer recommendations Close PUT** NEW **				
AB-Food	232.00	(+1.00)	-2	-20	-102	-17	-117	-103
Trend DOWN			Computer recommendations Close CALL** NEW **				
AB-Fort	505.00	(+6.00)	9	52	-94	45	-85	-75
Trend LEVEL			Computer recommendations Close PUT				
ADT----	114.00	(-3.00)	751	751	751	751	751	751
Trend UP			Computer recommendations BUY (CALL OPTION)** NEW **				
Abbey-L	275.00	(+8.00)	7	18	1	50	-86	-70
Trend LEVEL			Computer recommendations BUY (CALL OPTION)** NEW **				
A-Lyons	372.00	(+10.00)	10	9	-67	2	-91	-85
Trend LEVEL			Computer recommendations Close PUT				
Amersham	485.00	(+1.00)	-23	54	-97	17	-91	-85
Trend LEVEL			Computer recommendations Close PUT				

Another utility enables the package to download prices unattended, for those brave enough to go on holiday leaving their computer switched-on. Most of the required facilities seem to have been built in to the main program, which provides the useful side-effect that default responses are entered after long pauses. Thus the computer can be left unattended to complete one evening's data catching; the utility extends this indefinitely.

First attempts to use the package were not encouraging. Nothing would induce the program to download, due to missing data. The manual gave no clue, and only inspired guesswork saved the day. Later versions of the manual are more informative, and would have saved much time. Why are otherwise good programs so often accompanied by poor documentation?

The share renaming utility also failed on two occasions, leading to a confusing "Disc Failed" error message. A debugging session corrected the problem, but one wonders how many other people suffered the same failure at the same time.

Charts and graphs are slow to display on screen or dump to printer but are of a very acceptable quality, with the occasional exception of the point and figure chart, which sometimes became very large. The daily report does not allow shares to be selected, and printing out a full report of all prices is terribly slow, and of doubtful worth. Chartism is a fickle science, and you would follow these recommendations blindly at your peril.

This package is unlikely to appeal to a large number of investors. The single computer type, and the need for the Teletext adaptor and twin double-sided disc drives would make it expensive to set up. But it is a relatively cheap program in expensive company. Compared against other financial packages, it covers a very restricted part of the Stock Market, and gives limited analysis. But the automatic access to the 500 day histories of 170 prices is very painless and the resulting information most interesting. Alternative sources to Ceefax get expensive when used daily, and newspaper share prices are unavoidably out-of-date. Perhaps this package is more useful for those learning about the Stock Market than for those for whom it is only too real at times.

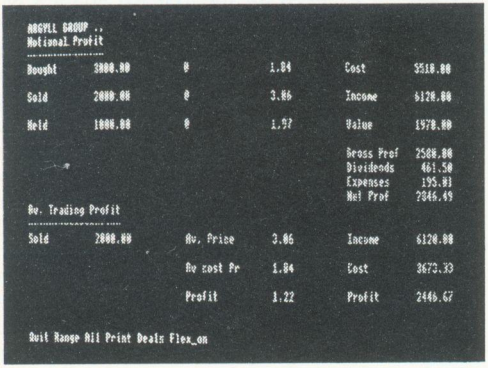


Product Sharemaster
Supplier Synergy Software
7 Hillside Road,
Harpenden, Herts AL5 4BS.
Tel. (05827) 2977
Price £99.95 inc. VAT

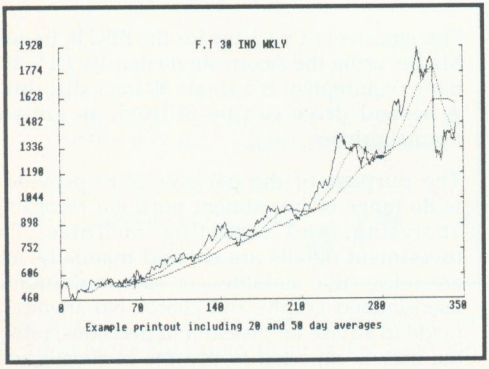
Sharemaster is a package for the BBC B, B+, and Master using the Acorn single density DFS. The basic requirement is a single 80 track disc drive. A second drive can be utilised, as can any normal printer.

The purpose of the package is to provide a wide range of investment portfolio recording, analysing, and reporting facilities. The investment details are entered manually, and are retrievable as tables, or as a multitude of user-defined graphs and charts. No attempt is made to advise on investment decisions; rather, the user is expected to become knowledgeable in the subject and make his or her own judgements.

The software is on a single 80 track DFS disc and accompanied by a smoothly written, comprehensive manual. This disc can maintain a portfolio of 17 shares, but a second and third disc can be added to increase the portfolio by 18 and 15 shares respectively to a total of 50. The disc is not protected, and permission is given for the user to make copies - hence more than one disc set can be created, giving an unlimited portfolio size.



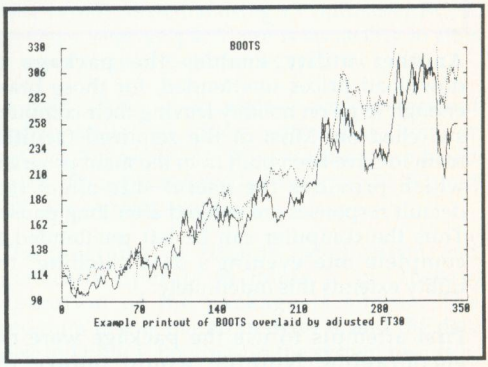
A configuration file allows various default settings to be preset, such as single or double drives, printer type, and text colour. Each share price record can be up to 500 entries, elected to be daily or weekly when created. Each company can have a text entry for recording comments or historic details, and all buying and selling details can be recorded for all transactions. The graphing opportunities defy simple description.



Example printout including 20 and 50 day averages

The program uses mode 0 throughout, and is menu driven either with menu screens or using prompts. On-screen instructions are minimal minimal, and explanations of errors are terse or non-existent. But the manual is splendid, and will resolve any operating problems.

At first glance the package seems impossibly complicated. The permutations of nested menu options and the user-variability in the reports produce never-ending possibilities, whilst new financial analysis terms might puzzle the newcomer. But the package can be used at any level from the superficial to the complex, and confidence soon builds up. To aid this, several data files are already entered, so that the user is not faced with blank screens, and can see what is required and what can be produced - a considerate touch that contributes greatly to understanding.



Example printout of BOOTS overlaid by adjusted FT30

The main menu provides access to the Share Price Editor, Deal Database, Graphical Analysis, Profits and Valuation, Price Analysis, and File/Printer Manager. Each of these leads either to further menus, or options via screen prompts. Each operation seems to have sensible default actions or values so that the built-in flexibility need not be overpowering. Prices are entered or altered through the Editor, whilst the Deal Database caters for detailed transaction records, such as buy and sell prices and dates, commission and VAT, actual and notional profit, and informative text.

Output can be in the form of simple price printouts or various numerical and graphical analyses. It is the latter which constitute perhaps the most complex and informative part of the package.

Continued on page 29

PCB AUTOROUTE

Upgrade your Pineapple PCB designer with the latest autorouting facility. Geoff Bains reports.

Product	**PCB Autoroute**
Supplier	**Pineapple Software**
	39 Brownlea Gardens,
	Seven Kings,
	Ilford IG3 9NL.
	Tel. 01-599 1476
Price	**£55 (PCB upgrade)**
	£185 (complete package)

The biggest bugbear of designing a printed circuit board is actually deciding where all the copper tracks must go to make the connections needed, without crossing over each other or attempting other such impossibilities.

Although a software package such as Pineapple's PCB (see BEEBUG Vol.5 No.8) is a great help to board designers, it still leaves the real decision making up to you. PCB is essentially only a specialised drawing package. Now (and somewhat miraculously) PCB autorouting has come to the BBC micro in the form of an add-on package for Pineapple's PCB.

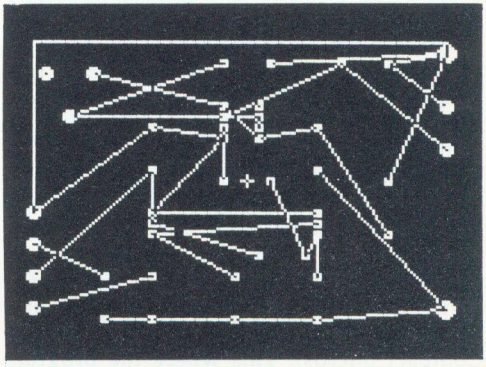
PCB Autoroute is, like PCB itself, in ROM. It interleaves with the existing PCB program perfectly. When you get to the tracking stage of designing a board, a function key is pressed to enable the new software addition.

Now you enter the connections to be made. This is all done visually. From each component lead (already marked on the board with a solder pad) the cursor is used to rubber band a line directly to where that pad is to be connected. This doesn't have to be another pad- it could be just an empty point on the board or, more likely, to join onto a 'bus' of connections.

At this stage some intervention can be made on the forthcoming automatic course of events. Each connection can be biased towards the topside or solderside of the board, or with equal preference for either. You can also bias the connection from the start point in a particular direction (entered as compass directions NE, NW, SE and SW) so as to start off the autorouting algorithms in the correct direction.

There is nothing to stop you entering some tracks completely by hand, either at the start or half way through the interconnection defining. You just switch back to the routing stage of the normal PCB program and take it from there.

The interconnections can be easily edited too and they are all stored on disc like all other stages of the design process, so you do not have to finish it all at one sitting - very useful for complex boards which get a little mind-boggling after staring at the screen for a while.



You do have to be pretty methodical when making the connections or you get in a terrible mess with nothing more useful than a spaghetti picture on the screen. However, if you are logical and ordered about the whole procedure it is simple enough. Once all the desired connections are made, another function key is pressed and the software gets on with it. The autorouting process takes anything from 30 seconds to a few minutes depending on the complexity of the board, but it draws tracks as it goes along so it is great fun to watch. When it is finished, it reports connections it cannot complete, which can be simply highlighted to be altered or drawn in by hand.

PCB Autoroute will not solve all your PCB design problems - it does require both careful thought beforehand and a fair amount of tidying up afterwards. Although it appears expensive, there is nothing else remotely similar, even on other micros, for less. Many would have said it couldn't be done but it just goes to show there is life in the old Beeb yet.

OVERLAYING TECHNIQUES

Bernard Hill explains how to expand your programs beyond the normal memory limits, using disc and sideways RAM.

A basic BBC micro (even a Master) has a very limited amount of RAM available for Basic programming. Even with shadow memory installed and PAGE down to &E00 this amounts to only 28.5K. Without shadow memory and with a DFS installed this could be as little as 4.75K on a model B. Other ROMs such as the ADFS will reduce this even further. Yet even a simple BBC micro can have sideways RAM fitted at a very low cost. Unfortunately, this additional RAM cannot be used directly from Basic without tailor-made software (or by using the SRREAD and SRWRITE commands on the Master and Compact micros). This where the program listed here comes in. It will allow you to expand your programs to use sideways RAM (and a simpler version uses disc) with a technique called overlaying.

OVERLAYING

If a Basic program is too large for the available memory, it can be split into a number of smaller programs which can be CHAINed in, one after the other, even returning to the main program if required (as used by Masterfile II). The process becomes awkward if the programs have to communicate with each other via the resident integer variables or other spare RAM areas. If we want to keep the same variable names and values between separate sections of code then we have to employ another method.

The solution lies in the well established technique of overlaying. The available memory space is divided into two sections: the root and the overlay areas. A master program sits in the root area, and controls the loading of program sections from disc to the overlay area. Then control jumps from the master program to the overlay, and back upon completion. In this way different sections of program can be loaded and executed in the overlay area. The number of overlays is governed only by the capacity of the disc drive, but it clearly slows down execution of the program during the loading phase.

But what if those sections of program could reside in sideways RAM, and be loaded into the overlay area when required? Execution speed is then hardly reduced, as memory swapping can be very fast indeed. What is needed is a simple piece of housekeeping software to make the whole process very streamlined.

In the rest of this article we shall first see how this can be done very simply from disc, and then go on to a sideways RAM implementation, which could also be used with tape systems as the overlays are only loaded once into sideways RAM, no matter how many times they are executed.

IMPLEMENTING OVERLAYS ON DISC

First we reserve an overlay area of memory. The simplest way of doing this is to reduce HIMEM. Let us suppose all our overlays will occupy &400 bytes (1K) or less. So if we reduce HIMEM by &400 this provides the space we need.

But how do we access any module resident in the overlay area at the new HIMEM? The answer is by a very neat trick. Whenever Basic sees a GOTO or GOSUB, it searches for the line in question starting from the current value of PAGE. So if we set PAGE to HIMEM (the beginning of our overlay area) and start with GOSUB 10, control can be transferred to line 10 of a program in the overlay area. If that program then ends with RETURN the jump is automatically made back to themaster program. Finally, reset PAGE and we can then continue as normal. Program 1 shows how this can be embedded inside a single procedure PROCexecute, whose sole parameter is the filename of the program to be loaded and executed in the overlay area. Note that variables are preserved across the modules.

The overlay, to be saved as OVER1:

```
10 PRINT i,i^2;SPC5;SQR(i)
20 RETURN
```

Main program, save as ROOT1:

```
  10 Osize=&400:HIMEM=HIMEM-Osize
  20 FOR i=1 TO 100
  30 PROCexecute("Over1")
  40 NEXT
  50 END
  60 :
1000 DEFPROCexecute(n$)
1010 Oldpage%=PAGE:PAGE=HIMEM
1030 OSCLI("LOAD "+n$+" "+STR$~HIMEM)
1040 GOSUB &A : PAGE=Oldpage%:ENDPROC
```

If you run the main program (ROOT1), you will hear the disc drive spin into operation each time the overlay is loaded into memory. This simple example uses only one overlay , but it does demonstrate the principle.

Note the use of GOSUB &A in line 1040. We use "&A" rather than "10" so that any renumbering of the root program will not affect this line, as it must always remain 10, the first line of the overlayed program section. Note that the variable Oldpage% temporarily stores the old value of PAGE.

Running this program will prove that the system is fairly slow, due to the loading of the module OVER1 100 times in the loop. The following modification will prevent the same module being reloaded unnecessarily:

```
  15 overname$=""
1030 IF overname$<>n$ THEN OSCLI("LOAD
 "+n$+" "+STR$~HIMEM)
1035 overname$=n$
```

This brings the execution time of the program down very close to standard Basic speeds.

RESTRICTIONS

Of course, this system has a number of restrictions:

1. Each overlay must not exceed the room allocated (in this case &400 bytes).
2. The first line of the overlay must be line 10.
3. The module must end with a RETURN instruction.
4. Care must be taken over procedures and functions called.

When a procedure is called for the first time, Basic searches the current program (from PAGE) for the procedure definition. This means that if you call PROCfred from an overlay when the procedure is defined in the root program, you must first set PAGE to Oldpage% in the overlay to enable Basic to find the procedure definition (then you cannot subsequently use GOTO or GOSUB within the overlay without setting PAGE back to HIMEM first).

On the other hand, if PROCfred has already been used by the root program, then Basic knows where it is (it keeps a list of addresses of functions and procedures) and will jump back into the root program without any trouble at all, so you won't need to alter PAGE.

Obviously, you ought to avoid defining procedures in the overlay segment as they might not be in the overlay area when you want them! All you need in order to use these ideas in your own programs is to observe the restrictions above, retain just the first line of ROOT1, and its 'execute' procedure.

IMPLEMENTING OVERLAYS IN RAM

As an extension of the above idea, we can first load all the overlays we shall need into the overlay area and shift them to sideways RAM. Then they can be downloaded and executed when required. Let us consider the following overlays:

Overlay One:

```
10 PRINT I;
20 RETURN
```

Overlay Two:

```
10 PRINT I^2;SPC5;
20 RETURN
```

Overlay Three:

```
10 PRINT SQR(I)
20 RETURN
```

These should be saved with names given. Our new root program (ROOT2) has to contain code

which will move program segments in and out of sideways RAM, and locate any required module. The modules are stored in blocks of &400 (Osize), so that we could fit 16 modules of this size into a 16k block of sideways RAM.

A program which begins with line 10 always has the same three bytes at the beginning (&0D,&00 and &0A). Since these are constant we can replace them in sideways RAM with the first three letters of the overlay name. This means that we can address the overlay by name in order to search and load from sideways RAM, but it has the drawback that we cannot distinguish beyond the first three letters (i.e. "Three" and "Thrust" would be\ indistinguishable). Make sure that the case of the letters matches when calling PROCLoadOverlay and PROCexec - "Two" is not the same as "TWO".

Have a look at the listing of ROOT2. Firstly, the overlay area "Osize" and the sideways RAM number (&F) is defined. Following assembly of the fast transfer and search code, HIMEM is adjusted as in ROOT1, and a variable to count the number of overlays (Nover) is initialised. Then the three overlays above are loaded into sideways RAM by three calls to PROCLoadOverlay. Execution of the program then continues in the loop, where PROCexec does all the hard work of moving a module down from sideways RAM and executing it. Considering that this loop does 300 transfers of 1K overlays during execution, I think you will find the speed very fast indeed.

CUSTOMISING

To fit your own overlay requirements, the only sections of code you need to keep are lines 100-130 of ROOT2, together with the definitions of PROCassemble, PROCLoadOverlay and PROCexec.

No doubt, there is still some scope for further improvement. You could use the currently-loaded overlay name idea in ROOT1, or allow for separate banks of sideways RAM to be loaded and downloaded. However, both examples given here provide practical ways of creating larger programs than memory will normally allow.That has to be good.

```
  10 REM Program  Overlay Demo ROOT2
  20 REM Version  B1.1
  30 REM Author   Bernard Hill
  40 REM BEEBUG   April 1988
  50 REM Program  subject to copyright
  60 :
 100 Osize=&400:swr=&F
 110 PROCassemble
 120 HIMEM=HIMEM-Osize
 130 Nover=0
 140 :
 150 PROCLoadOverlay("ONE")
 160 PROCLoadOverlay("TWO")
 170 PROCLoadOverlay("THREE")
 180 :
 190 FOR I=0 TO 100
 200 PROCexec("ONE")
 210 PROCexec("TWO")
 220 PROCexec("THREE")
 230 NEXT
 240 END
 250 :
1000 DEFPROCassemble
1010 DIM shift 160
1020 FOR opt=0 TO 2 STEP 2
1030 P%=shift
1040 [ OPT opt: LDA &F4 : PHA
1050 LDA #swr : STA &F4 : STA &FE30
1060 LDX #Osize DIV 256: .lp1 LDY #0
1070 .lp2 LDA (&70),Y:STA (&72),Y:INY
1080 BNE lp2:INC &71:INC &73:DEX
1090 BNE lp1:PLA:STA &F4:STA &FE30:RTS
1100 .load
1110 LDA #&80:STA &71:LDY #0:STY &70
1120 LDA &F4:PHA:LDA #swr:STA &F4
1130 STA &FE30:.loop3 LDY #0
1140 .loop4 LDA (&70),Y : CMP &100,Y
1150 BNE notfound
1160 INY :  LDA (&70),Y : CMP &100,Y
1170 BNE notfound
1180 INY :  LDA (&70),Y : CMP &100,Y
1190 BNE notfound : LDA 7 : STA &73
1200 LDA #0:STA &72 : PLA : STA &F4
1210 STA &FE30:JSR shift:LDY #0
1220 LDA #13 : STA (6),Y
1230 INY : LDA #0 : STA (6),Y
1240 INY : LDA #10 : STA (6),Y : RTS
1250 .notfound
1260 LDA &71 : CLC : ADC #Osize DIV 256
1270 CMP #&C0:BEQ notexist
1280 STA &71 : JMP loop3
1290 .notexist
```

```
1300 PLA : STA &F4 : STA &FE30
1310 BRK
1320 EQUB 0
1330 EQUS "Overlay not found":EQUB 0
1340 ]
1350 NEXT
1360 ENDPROC
1370 :
1380 DEFPROCLoadOverlay(n$)
1390 LOCAL I
1400 OSCLI("LOAD "+n$+" "+STR$~HIMEM)
1410 IF LENn$<3 THEN n$=n$+LEFT$("    ",
3-LENn$)
```

```
1420 FOR I=0 TO 2
1430  I?HIMEM=ASCMID$(n$,I+1):NEXT
1440 !&70=HIMEM
1450 ?&73=&80+Nover*Osize DIV 256
1460 CALL shift:Nover=Nover+1:ENDPROC
1470 :
1480 DEFPROCexec(n$)
1490 IF LENn$<3 THEN n$=n$+LEFT$("    ",
3-LENn$)
1500 $&100=n$ : CALL load
1510 Oldpage%=PAGE:PAGE=HIMEM:GOSUB &A
1520 PAGE=Oldpage%:ENDPROC
```

STOCK MARKET ANALYSIS (continued from page 24)

Graphs can be either linear or semi-logarithmic, with variable x and y axis scaling. A grid can be overlaid to aid coarse value reading, but a pointer facility allows a precise reading to be taken from any part of the graph.

Any graph can be overlaid by any other available (such as the FT-30) either absolutely, re-based, relatively adjusted, or indexed. The graphs can be smoothed using any weighting constant, and can be overlaid with any moving averages, either centred or lagged, or High and Low curves. Even lines of best fit can be plotted, or ruler lines and text placed by the user. Rate of Change and Point and Figure charts can also be produced with many of the variable options available for the graphs.

There are only one or two minor points of irritation: the need to enter the date including the century, even though the program ignores it, the inconsistent means of leaving routines (51,9,Q, or Return), and the impossibility of aborting lengthy printer output. There is much more that could be said given the space.

Suffice to say, that Sharemaster is a most comprehensive and impressive package, useable by a wide range of investors to a depth ranging from the trivial to the most advanced. It does not give an opinion, but enough information is available to satisfy every level of use. Those prepared to study the techniques open to them will be able to learn to exploit the program to the full.

COMPARISON

To use chartism to buy shares at the right time involves having a considerable pre-existing database on which to draw. Unless you have a clear idea of which shares to watch, this will involve a vast amount of time, effort, and possibly money. Sharematic gives a far wider database than Sharemaster, but this greater width is hardly significant unless your field of interest lies within the shares broadcast. Sharemaster allows you to concentrate on those shares of interest, but therefore assumes a hunch on your part.

Assuming that you have a share whose price history exists in both programs, both will assist in guessing the selling time. Both can use moving averages, which is the most popular indicator. Sharematic offers a buy/sell recommendation, which Sharemaster, with vastly greater analysis potential, wisely avoids. Personally, I would prefer to learn more interpretive skills, and make my own decision.

If portfolio management is required, then Sharemaster must be the option. To get an easy, free, Blue-Chip and Indices overview, then Sharematic is your choice. The two are simply so different that they are just not in competition.

1st COURSE

Character Control (part 2)

Mike Williams examines ways of using more of Basic's string functions.

Let's start this month by learning how to use some additional string functions. In fact, one of these was mentioned in passing towards the end of the previous article, and that is the ASC function. This returns the ASCII code of any character, and if applied to a string, it returns the ASCII code of the first character in the string. For example, ASC("A") will generate the value 65, which is the ASCII code for A (in decimal). So also will ASC("ABC").

Fortunately, most of the characters, like the letters of the alphabet, are allocated codes in a sensible way, so after A, B has a code value of 66, C is 67 and so on. Likewise, the numeric characters (i.e. the digits from 0 to 9) have consecutive codes from 48 to 57). This is a fact which we can make use of as we'll see in a moment. First, here is another short example of the use of the ASC function. See if you can work out what this program will do if you were to enter the name "BASINGSTOKE".

```
100 INPUT word$
110 length=LEN(word$)
120 FOR count = 1 TO length
130 char$=MID$(string$,1)
140 PRINT char$,ASC(char$)
150 NEXT
```

If you type the program in as listed above and then run it, you will soon see what it does. It lists one by one the letters of the word specified, with the corresponding ASCII code next to each. Note the use of MID$ to extract each character in turn as we saw last time.

MENU CHOICES

Many menu displays, for example, use the letters A, B, C etc to mark the options available. It can often be convenient to convert these into the numbers 1, 2, 3 etc. This is easily done:

```
100 INPUT option$
110 choice=ASC(option$)-64
```

Subtracting 64 from the ASCII code of the letter input gives the desired result. As a tip, it is often helpful, when a single character only is to be input to choose a menu option, to use GET or GET$ to avoid the need to press Return as well. Simply replace INPUT option$ with option$=GET$. In fact, by using GET rather than GET$ the whole thing becomes really simple:

```
100 choice=GET-64
```

And no string functions in sight!

Now you might be thinking that if we numbered our option choices from 1 upwards we could avoid the need to convert our input altogether. Well yes, you could write:

```
100 INPUT choice
```

but if you use GET for your input you would still need to write:

```
100 choice=GET-48
```

in order to convert the ASCII codes for the digits 1, 2, 3 etc to their numerical equivalents (since ASCII "1" = 48 etc).

Some words of warning are in order. It is vitally important to understand and appreciate the distinction between the numbers 1, 2, 3 etc, the characters "1", "2", "3", etc and the ASCII codes for these characters. It is only too easy to feel confused over this. If so I suggest that you read this last section again. Secondly, do note that in the above examples we have assumed that input is single key, and a letter or a digit as appropriate. In practice it would probably be worth checking the value or character. In particular, we have assumed that all letters are entered in upper case. To cater for the possibility of lower case, the best solution is to use something like:

```
100 choice=(GET AND &DF)-64
```

This forces any alphabetic letter input into the ASCII code for its upper case variety regardless of its case.

ASCII CODES

The ASCII codes are all listed conveniently in the back of the User Guide. If this is not at hand, then the following short program will do the job for you:

```
100 FOR count = 32 TO 127
110 PRINT count,CHR$(count)
120 NEXT
```

This uses another common string function, CHR$. This generates a character from an ASCII code and as such can be thought of as the opposite of the ASC function. Now although the three line program above gives us the desired result, the ASCII codes and corresponding characters are adjacent to each other in the display and difficult to read. This is because the value of the variable *count* is a number, and automatically right justified, while CHR$ gives a character which is automatically left justified.

Suppose we want to improve on the appearance with both number and character left justified. The simplest way of achieving this is to replace line 120 above with:

```
110 PRINT STR$(count),CHR$(count)
```

The STR$ is a new function which takes a number and converts it into an identical looking string of characters. Thus the number 112 (for example) becomes the string "112", and is now left justified because it is treated as a string. The STR$ function can be very useful as we shall see.

Just as ASC and CHR$ can be thought of as opposites, STR$ has its inverse in the VAL function. This takes a number in string form and converts it into a corresponding value.

JUSTIFICATION

As we saw in the example above, correct control of justification is important in achieving good looking displays of information. Although we can rely on the system's own automatic justification, most experienced programmers prefer to do this themselves for complete control.

Justification uses the concept of a field, that is the number of consecutive positions available for the display or printing of a string, which could then be left or right justified within a field. The following procedure will display a specified string either right or left justified in a field of width w:

```
1000 DEFPROCjustify(string$,x,y,w,mode$)
1010 LOCAL length,pad$
1020 length=LEN(string$)
1030 IF length>w PRINTTAB(x,y)LEFT$(stri
ng$,w):ENDPROC
1040 PRINTTAB(x,y)SPCw
1050 pad$=STRING$((w-length),CHR$32)
1060 IF mode$="L" string$=string$+pad$
1070 IF mode$="R" string$=pad$+string$
1080 PRINTTAB(x,y)string$
1090 ENDPROC
```

Justification requires that the program calculates the difference between the length of the string and the width of the available space, and then adds that number of spaces to the left or right of the string to be printed. The procedure listed above is quite comprehensive, and in practice some lines could be omitted.

Apart from w, the other parameters used by the procedure are the string to be displayed, the x,y position of the field on the screen, and a mode letter to indicate left or right justification. We need to know the length of the string to start with, and this is calculated with the help of the LEN function in line 1020. Before proceeding further the procedure checks to see if the string is wider than the field width specified. If so, the string is truncated to the right and displayed before leaving the procedure. If there is no possibility of this happening then this line could be omitted.

Line 1040 removes any text already occupying the field on the screen, and may be omitted if this is blank anyway (a possibility if paged mode is used, but unlikely if a scrolling screen is in operation). The instruction here simply fills the field with spaces using SPC to do the job.

At line 1050, pad$ is assigned the required number of spaces (the difference between the length of the string and the width of the field). The STRING$ function used here is the last string function we shall meet. Its purpose is to generate a string of repeated characters, here spaces (CHR$32). In the following lines these spaces are added to the left or to the right of the string before display.

The procedure could be called as in:

```
PROCjustify("BASINGSTOKE",10,2,15,"L")
```

to left justify the name "BASINGSTOKE" in a field of width 15 starting in position (10,2). The procedure could be readily modified to cater for the centring of text within the field. Just add line 1075:

```
1075 IF mode$="C" PRINTTAB(x,y) LEFT$
(pad $,(w-length)DIV2)+string$
```

In this instance we have to add half of the previously calculated number of spaces onto the front of the number before printing.

Clearly, numbers in string format could be left or right justified using the same procedure. However, it is often useful to print or display numbers lined up by their decimal points

(known as decimal justification). Here is a procedure, using string functions again, which may be used to do just that.

```
   1000 DEFPROCdecimal(number,x,y,w,d)
   1010 LOCAL length,pad$,num$
   1020 num$=STR$(number)
   1030 IF INSTR(num$,"E")=0 THEN
        pad$=CHR$32 ELSE pad$="*"
   1040 PRINTTAB(x,y)STRING$(w,pad$)
   1050 IF pad$="*" THEN ENDPROC
   1060 length=LEN(num$)
   1070 p=INSTR(num$,".")
   1080 IF p=0 THEN PRINTTAB(x,y)FNdec1
        ELSE PRINTTAB(x,y)FNdec2
   1090 ENDPROC
   1100:
   1110 DEF FNdec1:LOCAL value$
   1120 IF length>w-(d+1) THEN
              value$=STRING $(w,"*")
        ELSE value$=STRING$(w-(d+1)-
length,
              CHR$32)+num$
   1130 =value$
   1140:
   1150 DEF FNdec2:LOCAL value$
   1160 IF length-p>d THEN
              num$=LEFT$(num$,p+d):
              length=LEN(num$)
   1170 IF p+d>w THEN
   1180 =value$
```

To try and make matters clearer I have written the procedure with two supporting functions called FNdec1 and FNdec2. Let's take a look at what is happening. The procedure, called PROCdecimal, has five parameters. These are the number to be displayed, the x,y position of the start of the field, the width (w) of the field, and the number of decimal places (d). One of the reasons why the coding may look more complex than you might have expected is that I have written the procedure to cope with cases where the number will not fit in the specified field (even when decimal places are truncated according to the value of d), and to recognise when the number would be displayed in exponent form (using the letter E), which the procedure cannot cope with. In these cases the field is filled with a row of asterisks.

At the start of the procedure, the number is converted into string form, and this in turn is checked for the exponent E (line 1030). If found asterisks are printed and the procedure terminated, otherwise the field is filled with spaces as before.

The procedure then determines the length of the number (in characters) and searches for the position of a decimal point which is assigned to p. If p is zero, then the number is an integer and the function FNdec1 pads it out and displays it correctly. If a decimal point is found, we have a real number and FNdec2 does this job.

You may find it easier to follow the workings of the two functions by referring to the diagram I used when writing the procedure in the first place (Figure 1). In FNdec1 we check to see if the integer will fit in the space available. If it will, we add spaces to the front of the number (w-(d+1)-length), otherwise a row of asterisks is returned.

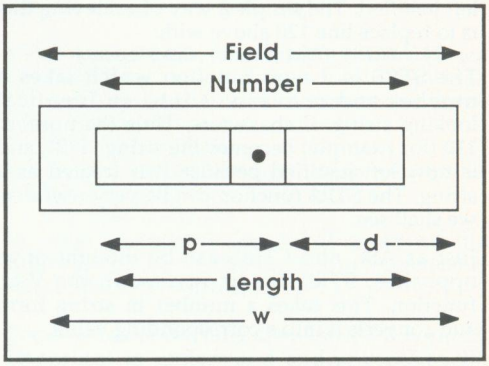


Fig 1 Decimal Justification

With FNdec2 we first check whether we need to truncate the decimal places (line 1160) and then proceed as before to see if the integer part of the number will fit. If not, asterisks are again returned, otherwise the appropriate number of spaces is added to the front of the number.

To test out the procedure try the following short program:

```
   100 MODE 7
   110 PROCdecimal(123,10,1,20,5)
   120 PROCdecimal(123.456,10,2,20,5)
   130 PROCdecimal(123.456789,10,3,20,5)
   140
PROCdecimal(123123123123,10,4,20,5)
   150 PROCdecimal(-456789,10,5,20,5)
   160 PROCdecimal(0.0023456,10,6,20,5)
   170 END
```

Of course, you are just as welcome to try some examples of your own. If you use the procedure (and supporting functions) in your own programs you may also be able to simplify the coding. Next month I will give some final examples of the use of Basic's string functions.

ACORN IN ACTION

A totally new experience for show visitors

at the
Royal Horticultural Hall,
Westminster, London SW1

ELECTRON & BBC MICRO USER SHOW

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

You'll find the very latest software and peripherals for the complete Acorn range at the Electron & BBC Micro User Show. But this time there'll be so much more to enjoy. Acorn In Action will demonstrate some of the truly amazing projects currently involving the machines . . .

SEE

★ **A spectacular laser light show controlled by a BBC Micro. (Saturday only)**

★ **The research work on the BBC Micro that has helped to bring new hope to sufferers of the eye disease glaucoma. (Friday and Sunday)**

★ **A program developed by an amateur astronomer to locate distant galaxies. (Saturday and Sunday)**

★ **The Beeb system being used by doctors at Guy's Hospital to provide a breakthrough in the treatment of arterial disease. (Saturday and Sunday)**

PLUS **watch your own heartbeats displayed, measure your manual dexterity and hear your own voice backwards – all courtesy of a BBC Micro.**

Take your seat in the Archimedes Demonstration Theatre run by Acorn's own experts. Thirty minute special introductory courses to the new machine will be held on the hour, every hour throughout the three days. Price just £1.

It all adds up to a fantastic day out for the whole family!

Avoid the queues! Get your ticket in advance – and SAVE £1 A HEAD!

Post to: **Electron & BBC Micro User Show Tickets, Europa House, Adlington Park, Adlington, Macclesfield SK10 4NP.**

ELECTRON & BBC MICRO USER SHOW

Royal Horticultural Hall
Westminster
London SW1
May 13-15, 1988

Advance ticket orders must be received by Wednesday, May 4, 1988.

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

Please supply:

___ Adult tickets at £2 (save £1) £_____
(Order four adult tickets, get the fifth FREE!)

___ Under 16s tickets at £1 (save £1) £_____
(Order four under-16s tickets, get the fifth FREE!)

Total £_____

☐ I enclose a cheque made payable to Database Exhibitions

☐ Please debit my Access/Visa card no:

Expiry date: /

Name _____ Signed _____

Address _____

_____

Postcode _____

A355

PHONE ORDERS: Ring Show Hotline: 0625 879920
PRESTEL ORDERS: Key *89 then 614568383
MICROLINK ORDERS: Mailbox 72:MAG001

Please quote credit card number and full address

BBC USER GROUP INDEX (continued from Vol.6 No.9)

STAFFORDSHIRE
"POTBUG", BBC Micro User Group, 50 Cliff St, Smallthorne, **Stoke-on-Trent**, Staffs ST6 1SQ. Tel. 818430.

STAFFORDSHIRE
A.C.Beckett, 57 Adonis Cl., **Tamworth,** Staffs B79 8TY.

SOUTHEND
Anybody interested in starting a Beeb User Group in Southend area please contact L.Martin at 42 Stuart Road, **Southend-On-Sea**, Essex, SS2 5JT. Tel. (0702) 611794

SUFFOLK (WEST)
BBC Micro Users Group. Contact the Club Secretary: Mr.A.Hurden, 14 Plovers Way, **Bury St Edmunds**, Suffolk. Tel. (0284) 5946.

SURREY
Canberley Computer Users Club
Meets on the 1st and 3rd Monday of each month, from 7.30 to 9.30pm in room 14 of the Adult Education Centre, France Hill Drive, Camberley, Surrey. ContactJohn Lyons on Camberley (0276) 65275.

SURREY
Sutton Library Computer Club
Contact David Wilkins (secretary), 21 Village Row, Sutton, Surrey SM2 6JZ. Tel.01-642-3102.

SURREY (WEST)
The **West Surrey Computer Club**

SUSSEX
BRIGHTON, HOVE and District Computer Club
The Hon Secretary, Mr.Cyril Osborne, 27 Woodland Court, Dyke Road Avenue, Hove, E.Sussex, BN3 6DP.

SUSSEX
BOGNOR CompUter Group (BUG)
Beeb sub-group. Meets twice a month at the R.A.F.A Club, Waterloo Sq., (opposite Pier), Bognor Regis, Sussex at 7.30pm on the second and fourth Thursday of the month. All Beeb users welcomed.

SUSSEX (WEST)
Midhurst & District Computer User Group
Meets 2nd and 4th Thursday of each month from 7p.m. to 9p.m. Contact Val Weston on Midhurst 3876, or Robert Armes on Midhurst 3279 or at Cullens, North Street, Midhurst.

SUSSEX
BRIGHTON Area
Jim Price, Bedford House, 27/28 St Georges Rd., Brighton, Sussex.

SWANSEA
Mr.R.P.Baines is very interested in forming a local user group for BBC micro users in the Swansea area. Anyone interested in such a group should contact Mr.Baines at 20 New Road, Trebanos, Pontardawe, Swansea SA8 4DL.

TYNE and WEAR
NEWBUG. c/o N.Kidd, Multi-Purpose Centre, Ox Close Village, Washington, Tyne and Wear. Tel. 051 4166407.

WALES (SOUTH)
Cardiff BBC Computer Club (CBCC)
Meets alternate Wednesdays at University College, Cardiff. Further information from: Geoff Barker. Tel. Penarth 701023.

WALES (SOUTH)
Nicholas Goodwin, 22, Gendros Drive, Gendros, **Swansea.**

WARWICKSHIRE
Stratford Computer Club Meets on the second Wednesday of each month at the Wesley Hall, 7pm - 9pm. Chris Parry (secretary), 15 Kipling Road, Stratford-on-Avon, Warwickshire. Tel. (0789) 68080

WILTSHIRE
Salisbury
Contact Alastair Lack on 0722 77303

WILTSHIRE
R.W Packer, 10 Wheatlands, Haydonliegh, **Swindon,** Wiltshire.

Anyone interested in forming a BBC user group in this area please call **Swindon** 695296 or 695364.

WORCESTERSHIRE
Club meets every Tuesday from 7pm to 9pm at the South Street Annexe, Redditch Youth Centre, Ipsley Street, Redditch. Contact Mr.Anthony Green, 14 Radway Close, **Redditch,** Worcestershire B98 8RZ. or Tel. (0527) 61434.

YORKSHIRE

Meets fortnightly in Hornsea library. Contact Niel Willerton, **Hornsea** 4149.

YORKSHIRE (WEST)
Colin Price, **Keighley Computer Club,** Keighley Technical College.
Tel. (0535) 603133

Continued on page 38

DABS PRESS

Dabhand User News

Devoted to the Serious Expert User

David Atherton and **Bruce Smith** bring you the latest in Expert Software and Dabhand Guides for your BBC and Master micros.

If you have a printer then we believe HyperDriver will be one of the most significant purchases you can ever make.

Here's what **Beebug** said about our products in the March 1988 issue:

HyperDriver: *"...an ingenious blessing.. a million other good design features..."*

MOS Plus: *"..an excellent product."*

Master Emulation ROM: *"...the whole system FEELS just like a Master...the most impressive implementation is an almost complete emulation of the temporary filing system."*

Send or phone for our free 32 page catalogue.

HyperDriver: Printer Power

Possibly the most significant purchase you can make

HyperDriver isn't just another printer ROM - it's the *ultimate* one. It's absurdly easy to use and provides you with so many of the facilities missing from your current printer orientated software including: on-screen preview, CRT graphics, NLQ font, VIEW printer driver and user definable macros to name but a few.

No matter what you use your printer for, word processing, spreadsheets, databases, programming, you will have in excess of 80 * commands instantly available.

We really don't know how to cram all the facts in here, so read Geoff Bains review of HyperDriver in the March 1988 issue, pages 47/48, or ask for our latest catalogue.

Master Emulation ROM

Amazing as it may seem, if you own a BBC B or B+ then installing MER will give it virtually all the software features of a Master 128! See the Beebug review in the March 1988 issue, pages 28/29.

MOS Plus

MOS Plus is a must for all Master 128 owners not least because it fixes those irritating bugs in the OS and adds many essential extras such as *SRLOAD, *FORMAT and *VERIFY in ROM. See Beebug March 1988 pages 44/45.

Prices

Beebug members can obtain their normal 5% discount on these Dabs Press products simply by quoting their membership number (prices in brackets).

HyperDriver: ROM £29.95(£28.52), SWR £24.95 (£23.76)

MOS Plus: ROM £12.95 (£12.33), SWR £7.95 (£7.57)

MER: ROM £19.95 (£19), SWR £14.95 (£14.24)

Orders

Send cheques, POs, official orders to the address below, or quote your Access/Visa card number and expiry date. Credit card orders accepted by phone or mailbox. P&P free in UK. Elsewhere add £2 or £10 airmail. 3.5" ADFS disc £2 extra please state if required.

Dabs Press, 76 Gardner Road, Prestwich, Manchester, M25 7HU
Phone: 061-773-2413
Prestel: 942876210
BT Gold: 72:MAG11596

DABS PRESS

Personal Ads

Torch Z80 second processor ZEP100, with Perfect Software suite, ROMs and leads £75, some business software. 2 BBC Disc Drives, cased and new £50 each. 4 EMR MIDI software packages, Editor, Notator, Composer, Vu-Music, unused £80.10 BBC Educational Cassettes, cost £100+ new, £28. Tel. (037387) 459.

BBC B, with DFS, 32K sideways RAM, speech processor, ROM extension board, ROMs, light pen, digitiser, much software. All for £300. Tel. 01-767 5207 eves.

40 Track, single-sided 'Options' disc drive for sale. Fine working condition - owner upgrading. £50 o.n.o. Tel (0272) 623095.

WANTED Comms enthusiast in London/Kent area with Modem, Command software and know-how to help me set up a Bulletin Board. Please telephone 01-851 3072 evenings or weekends.

Acorn 6502 Second Processor £65. 'Genie System (PMS)' £40. Wordwise Plus £25. Wordwise £8. 'Wordex' ROM £10. View Version 3.0 £40. ViewIndex £5. ViewSpell ROM £10. From BEEBUG: 'Help' ROM £8, Spellcheck 2 £8, Wordease £8. Gemini 'Life & Business Organiser' £4, Beebcalc £5 and Mailist £4. STARdataBASE .ROM/disc manual £18. 'Fleet Street Editor' £18. Acorn database (disc) £2. More business software on disc and tape, all with manuals and inc postage. Tel. 01-399 2865.

BBC Model B with Acorn DFS, Microvitec CUB colour monitor, Teac dual 40/80 disk drives (400K), Wordwise Plus ROM, BBC tape recorder, games software (inc Snapper, Arcadians, Monster, White Knight), Joystick and 10 floppy disks. All very good condition. £725 o.n.o. Tel. 01-234 6180 (day).

BBC Model B, Watford DFS, Watford 32K RAM Board, APTL ROM Board + RAM chips and write protect switch, Computer desk, many programming books and manuals, Wordwise Plus, Disc Doctor, Toolkit Plus, Romit, Gremlin ROMs. All sorts of software + Digimouse with graphics software, all for £220. Microvitec 1451 AP/PAL/RGB Medium Resolution Monitor £180. Viglen Dual 40/80 DS Drives with internal power supply £130. Tel. Hythe Southampton (0703) 845295.

Beebug Help II £10, Romit £15, Watford's Word-Aid chip £10 or swap any for Sleuth or WYSIWYG. All original. Tel. Bedford (0235) 49052.

Magazines, Acorn User and Micro User. Complete sets, offers 'File Plus' programmable database £10. 'ROMSPELL' £10. 20 assorted early BBC cassettes £10. Tel 01-693 2922.

Thrust disc (40/80T) £6, Citadel disc 40T £6.00 and The Secret Diary Of Adrian Mole disc (80T DS) £6. All originals. Tel. Redhill 763482 or write to 18 Hillfield Road, Redhill, Surrey, RH1 4AP.

Wanted Viglan ROM Cartridge System for BBC B and PMS NTQ ROM. Tel. Derby (0332) 556381.

Master 128+ Manuals £320, 40/80 DS Drive £70, Interword £35, ADT £22, MAX £12, AMX Mouse and SuperArt (Master) £50, Office Mate and Master £15, Joystick (Analogue) £10. Games, other discs and box £50. Tel (0642) 786468.

Acorn speech (2 ROMS) £20, Beebug MUROM £15, CC Graphics ROM £15, Voltmace 14 B Joystick + interface £8 + £8, Quickshot Joystick £5, Beebug Paintbox + Sprites £5 each. Original tape software from Elite to Andraid Attack and lots more £4 each. Books - Understanding Basic £2, Basic Handbook £5, Basic for Beginners, Friendly Computer Book £4 each. Bush cassette recorder £5. C15 tapes 50p each. All prices negotiable. Tel. (051426) 0937

Archimedes A310M with latest version (1.09) PC Emulator, no monitor. Boxed as new £780. Also Master software: System Delta with Card Index and Mailshot £45, Printer buffer and Machine Code Monitor on disc £5 ea. Advanced User, Disc-User, ROM User Guides half-price or offer. Tel. (0272) 736237.

DFS upgrade kit (1770) for BBC with Manual £25. Replay 1770 and ADFS version £15. Fleet Street Editor £8. Tel. (0624) 71353.

Command Beebugsoft ROM, completely unused £20. Acorn Prestel adapter + ROM £40. Prism 1000 modem £30. Tel. (041889) 0144 eves.

BBC B 32K, issue 7, Watford ROM expansion board, 2 x 40/80 drives (one own PSU). Z80 second processor + software. Hi-def B&W monitor. All little used. Original cost c£900. Accept offers around £400. Tel. 01-404 4444 ext 3500, office hours.

BBC Master 128 £335, Mitsubishi Twin Disc drive plus power supply £115, Philips CM8833 colour monitor 3 weeks old £200. Computer Concepts Mega 3 ROM, contains Inter-Word, Sheet and Chart £45. BEEBUG Master ROM £16. Tel. (0244) 812331 ext 282 or (0978) 757295 eves.

Interword ROM with reference manual in original packing £25. Watford 32K shadow RAM and manual £35. Mini office II 40T Disc for BBC B £5. Tel (041942) 7197.

Acorn Cambridge Workstation 32016 second processor including all software, Panos, Lisp, Fortran 77, C, Iso Pascal. £400 ono. Penman Plotter £150 ono. Tel. 01-992 4935.

Software list database only £7 inc P&P and VAT. Offers full search facilities and print out option. Send SAE for details to Matthew Reardon, 43 Lesford Road, Coley Park, Reading, Berks, RG1 6DX or phone (0734) 576587.

Programs on disc for Master 128/Model B. Masterfile II DFS £5, Revs £4, Stykers Run £4, Acornsoft Hits Volume 1 £4, Miscue Analysis £5, The Artist ROM plus Wigmore Mouse £25. All o.n.o. Tel 01-341 2187.

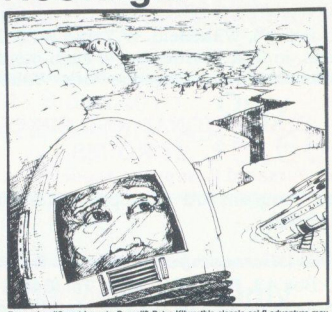
Wanted A.T.P.L board, also any 16K or 32K EPROMs. Write to Charles, 47 Clumber Drive, Gomersal, Cleckheaton, BD19 4RP.

Continued on page 40

ADVERTISING IN BEEBUG

For advertising details, please contact Yolanda Turuelo on (0727) 40303 or write to Dolphin Place, Holywell Hill, St Albans, Herts.AL1 1EX

Nostalgia



Remember "Countdown to Doom"? Peter Kilworth's classic sci-fi adventure may well have been the first game you ever fought. Well, RETURN TO DOOM - Part 2 of his Doom trilogy- is now available from Topologika:

You are flying through the universe, minding your own business, when a distress call comes in. "Mayday! The Galapoxi, taking the Ambassador of Regina on an important mission to Flaxo, has just crashed on Doom, Rescue needed! Heading for Cleft..." As the only person ever to have survived Doom, you steer once more for that dangerous planet. This could be your finest hour.

Or maybe even longer.....

As tough as Countdown - but a different sort of challenge - RETURN TO DOOM is sure to bring back memories. £12.95 inc disc, manual,VAT and P&P.

TOPOLOGIKA

Tel: (24 hours ACCESS) 0733 244682

FREEPOST PO Box 39, Stilton PETERBOROUGH

Business Ads

OMNIROM - the ultimate Utility ROM for the MASTER and COMPACT. Adds 3 new CONFIGURATION Commands - FOREGROUND and BACKGROUND COLOURS and FONT (which integrates 3 new Fonts), ADFS/DFS Menu System, plus many other utilities, such as 65C12 Disassambler, Memory and Disc Editors, BASIC STATUS and much more. Supplied on 16K ROM. Price £12.95 (p&p incl).

THE SCRAMBLER for BBC B, MASTER & COMPACT ALSO FOR ARCHIMEDES.

PROTECT your files for as little as £19.95 (p&p incl). The system to help you comply with the DATA PROTECTION ACT. ENCODES any file held on disc (ADFS/DFS). Menu driven. Unique PASSWORD. Supplied on ROM. ARCHIMEDES module version on disc at £25 (p&p incl). Ideal for schools, colleges and small firms.

Number 3 Software, 3 Dairy Farm Court, Attleborough, Norfolk, NR17 2BT.

BEEB-PPLANNER, Version 8, CPA program, Time Analyse 250 activities, calculates project cost, three calendars, various reports, uses sideways RAM, £39.95. ***E J Sheffield, 8 Langdon Close, Camberley, Surrey, GU15 1AQ.*** *Send SAE for details.*

BBC USER GROUP INDEX (continued from page 34)

YORKSHIRE (WEST)
Wakefield BBC Micro User Group, The Secretary, P.O.Box 65, Wakefield, W.Yorks WF2 6YZ. Meets on first Wednesday of each month at Holmfield House, Clarence Park, Wakefield at 7.30.

EDUCATIONAL AND SPECIALIST GROUPS

RAF Personal Computer Association.
Cyril Sampson RAF Brampton **Cambridgeshire** PE18 8QL.

MUSE (Microcomputer Users in EDUCATION)
P.O.Box 43, **Hull** HU1 2HD. Tel.(0482) 20268. Membership £15 pa (£13 by standing order). MUSE publishes six journals a year and has a considerable software catalogue of educational BBC programs.

CHELTENHAM
P.J. Heslip, **Stroud Teachers' Centre**, Arthur Dye School, Springbank Road, Cheltenham, Glos.

LIVERPOOL
J.R Hughes would be interested in hearing from BEEB users in special Education with a view to forming a special Sub-group (mainly by post and phone) at The Advisory Centre, 6 Chatsworth Street, Liverpool L7 6PT.

LONDON
W.E.Hunt, 143 Montague Road, Leytonstone, London E11 3EW. (Currently working on **mathematics, and science** programs).

INTERNATIONAL ADVENTURE CLUB (IAC)
For details of the club, write to:
The International Adventure Club, 10 Ennis Close, **Harpenden,** Herts AL5 1SS. (England).

PRIMARY HEALTH CARE Computer Group
Acorn/BBC user Subgroup. Open to all interested in use of computers in Primary Health Care. (e.g. Doctors, Nurses, practice managers). Contact Dr. Ken Walton, 7 Leighton Avenue, Pinner, **Middlesex** HA5 3BW. Tel. 01 429 0835.

British LOGO User Group (BLUG)
Enquiries to Richard Olney, BLUG, London New Technology Network, 86-100 St.Pancras Way, **London** NW1 9ES.

National Acorn Second Processor U.G.
Dave Sumner, **NASPUG,** 25 Cheshire Close, Bilton, **Rugby** CV22 7JU.

PROLOG Interest Group
Teacher's Centre, Barlow Moor Road, West Didsbury, **Manchester** M20 8PW. Tel. 061-434-3421.

OVERSEAS

AUSTRALIA

AUSTRALIA BBC Teacher's Group,
C/o D.Arbuckle, Perth College, P.O.Box 25, Mt.Lawley. 6050. Western Australia. Telephone: (09) 272 1222 Dialcom/Minerva 07:PEC001

BEEBNET. P.O.Box 262 Kingswood S.Australia 5062

DAPTO BBC USER GROUP or DBUG.
Meet on the last Wednesday of each month at 7pm at: 3/74 Princes Highway, Dapto New South Wales. Contact: K.Nicholls, P.O.Box 447 Dapto.

Educational User Group (AEUG)
Melbourne Grammar School, Domain Road, South Yarra 3141, Victoria, Australia.

The Sydney BBC Micro User Group,
c/o Mr S.McCann, 500 Miller Street, Cammeray, N.S.W. 2062, Australia. (Tel: Sydney 923-1137)

TASBEEB BBC Users Group
Box 25, P.O., North Hobart, **Tasmania,** Australia.

VICBUG (The Victorian Acorn User's Group)
Ted Robinson, 31 Curtin Ave, West Brunswick 3055, Victoria, Australia. Tel. (03) 386 7529.

BELGIUM

Acorn Computer Users Club
Boulevard du Prince de Liege 222 1070- Brussels, Belgium.

Jean-Louis Meerts. **Acorn Computer Users Club** BP.125, 1000- Brussels, Belgium.

Georges Wathlet. **Acorn User Club**. Rue des Clematites 9 bte 1 B-1080 Brussels Belgium. Tel: 02/425 1220

HOLLAND

Asterix, P.O.Box 177, 4760 AD, Zevenbergen The Netherlands.

Please note the changes :

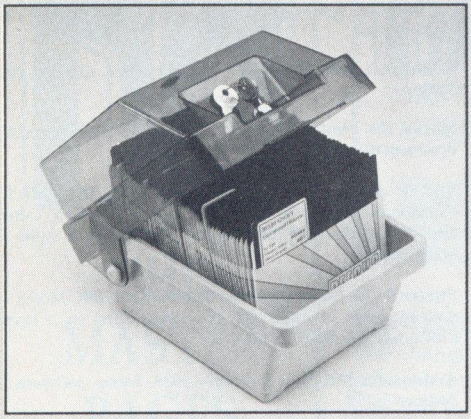
Birmingham User Group will have a new address from the 12 April: Unit 16, 60 Regent Place, off Caroline Street, Birmingham 1.

West London Personal Computer Club. Contact: Blue on 01-579 5415 or Peter on 01-997 1218

To be continued

BEEBUG Discs - The Ultimate in Quality & Reliability

PRICES REDUCED!



50 discs with free lockable storage bock

50 Single Sided 40T Double Density Discs — Only **£35.60**

50 Double Sided 80T Quad Density Discs — Only **£39.90**

Prices shown are members prices and include VAT.

BEEBUG discs are manufactured to the highest specifications and are fully guaranteed.

Our Guarantee

We confidently offer a lifetime data guarantee and will replace any disc with which you encounter problems. We have found that the standard of quality control at the factory makes this necessity very rare.

40 Track Single Sided Double Density

	Price	Members Price	Order Code
10	£9.37	£8.90	0657
25	£23.00	£21.85	0661
50	£37.50	£35.60	0665

80 Track Double Sided Quad Density

	Price	Members Price	Order Code
10	£10.42	£9.90	0660
25	£26.20	£24.90	0664
50	£42.00	£39.90	0668

Please send me _____ (qty) _____ (stock code) at £_____ (unit price)

UK post 10 £1, 25/50 £3.75. Overseas send same price inc. UK post & VAT

I enclose a cheque for £_____ /Please debit my Access/Visa card £_____

Expiry date: ☐☐

Name _____ Memb No. _____

Address _____

_____

BEEBUG
Dolphin Place,
Holywell Hill,
St. Albans,
Herts.
AL1 1EX

☎ (0727) 40303

Personal Ads (continued from page 36)

BBC Master 128 £335. Philips CM8833 colour **monitor** 3 weeks old £200. Computer Concepts MEGA 3 ROM, contains Inter-Word, Sheet and Chart £45. Tel (0978) 757295.

MP165 Dot Matrix Printer, v.g.c with handbook £98. Philips Monochrome Monitor, good condition with lead £49. Angled printer stand, 80 col, new, unused £20. 2 Viglen Master Cartridge Systems (includes 2 spare ROM carriers and stand), boxed, never used £19.50. Telephone Cheltenham (0242) 36690.

BBC B with DFS, Solidisk SWR128 (+ software) with 65C02 processor and ZIF socket £300. CUB colour monitor £160. Challenger 3 in 1 disk drive £210. Care electronics P.S.U. (2 outlets) £20. Acorn Teletext Adaptor £100. Facit 4510 printer (120 cps) £200. Solidisk EPROM programmer + eraser £20. Plus software £10 each. Various other software + ROMs. Tel. 01-504 8467.

BBC C computer with twin disc 'Torch graduate' business system, with spreadsheet, wordprocessor, database, graphics (pie/bar chart, etc). All Psion software. 3 manuals cost £1,200, bargain at £400. Tel. (0521) 42456.

Nidd Valley Digimouse with Grafik (for BBC B or Master) and Chauffeur (for Master 128). Both hardly used. Value £59.80, price £25. Tel. (04243) 4500 Sussex.

Software Bargains! All boxes and documentation supplied. AMS Pagemaker £26.50. AMX Extra Extra £11.50. Beebug Toolkit Plus £14.50. Beebug Spellcheck III £14.50. Play It Again Sam £6.50. Life of Repton £6.50. Studio 8 £8.50. Tel. (04024) 41136 eves.

BBC Master 128 £260. Turbo upgrade £60. 512 upgrade £110. UNI processor £45. Acorn teletext £60. Acorn Prestel £50. Mono monitor £50. Colour monitor £150. Dual 80T drive, own PS4, switch cable, £150. 30MB Winchester £400. Overview £50. ADI £10. ADT £10. ADFS ROM £10. Manuals 1 and 2 £10. Tel. (0533) 312661.

BBC B with DFS, 40/80T disc drive, Phillips monochrome monitor, Wordwise, Quickcalc, Masterfile 2, Joystick, other software and user guide £375. Tel. 01-903 1673.

6502 second processor, DNFS, software and manual. As new £90 o.n.o. 40T S/S uncased Tandon drive £15. Oxford Pascal, GAC, Game Core unused £10 each. Tel. Dartford (0322) 22327.

Turbo board £50. 22 x 40 trk S/S £3 the lot. Acorn SWROM cartridges £4 each. 16 x 80 trk S/S £3 the lot. Sealed box 10 branded 80T D/S £6. Model B 128K solidisk + 32K swe RAM, unused £10. 27128s £1.50 each. All prices include postage. Tel. Stirling (0786) 833541.

Printer Shinwa CPA 80, very good condition, Centronics parallel interface. Complete with manual £100. Tel. (0438) 354385.

22 Beebug monthly magazine cassettes from vol 4/09 to current. Can be easily transferred to disc £15 + pp. Tel (0274) 571406.

Electron + 1, cassette, power leads, manuals, games, etc. 1 yr old. Accept £80 o.n.o. Phone Lincoln 703087.

Juki 5510 Matrix Printer not working due to printhead failure but useful for spares or possible repair. Offers - please phone (0455) 635587.

Beebugsoft Printwise 8 pt. Used only once. £15. Tel. (0446) 52088.

Master 128. Mint condition. Only just out-of-guarantee £295. Bracknell (0344) 51308.

BBC B, Acorn DFS, Teletext adaptor, 40T 100K, 80T 200K Cumana drives, 16K SWR, 65C102 second processor, software and Panasonic KX-P1081 printer £750 o.n.o. or split. Tel. (0283) 65635.

Printer Smith Corona fast text 80 £100 o.n.o. BBC Model with data recorder, manuals and 14 games; Elite, etc + Joystick £180 o.n.o. Tel. (0753) 75141.

Archimedes 310. New February £625. Some software. Tel. Bedford 56139.

Taxan Kaga Supervision 11 620 Colour monitor, BBC, IBM, Apple compatible. Excellent condition, boxed with leads and instructions. £190 o.n.o. Akter 40/80 double sided disc drive, excellent condition, boxed with leads and instructions. £70 o.n.o. Tel. 01-341 2187.

BBC hardware, books, software on tape, discs and ROMs all for cheap. For list please send SAE to: 3 Greatwood Street, Batley, West Yorkshire, WF17 7NJ.

Wanted 80186 co-processor or solidisk PC-500 for BBC Master as soon as possible. Send price and details for quick reply to: 3 Greatwood Street, Batley, West Yorkshire, WF17 7NJ.

WANTED for retired enthusiast, EPSON MX 80 printer or similar for BBC B. Reasonably priced. Tel. (0670) 716109.

WANTED Project Planner by Brainpower (Colinsoft). Purchase or copy as no longer available. Tel. 01-894 6449.

Opus dual 40T SSDD drive, Opus DDOS, Opus utilities disk, disc manuals and an external disc drive power supply, all as new £100. Tel. (0962) 734367 after 6 pm.

Beebug, Mirco User, Acorn User. Full set of each from issue 1, most bound. Offers? Tel. (0372) 58655.

Sleuth ROM 1.06 with manual in original packing £10. Tel. (0202) 693734.

Magazines - Acorn User, Dec '82 - Dec '87 £40 the lot + transport at cost. Micro User, March '83 - Jan '88 £35 the lot + transport at cost. Tel. (0789) 68080 eves.

EPSON RX 80 F/T printer £150 ono. Tel. (0205) 68276 after 6 pm.

Interword £25, Educad £30. Both boxed as new. Tel 01-500 5701.

THE MASTER PAGES

Devoted to the Master Series Computers

One of the most useful programs we have published for the Master is the Printer Buffer Utility by Tim Powys-Lybbe (Vol.5 No.7). This month Derek Baron presents a number of major enhancements to the original program to improve its usefulness even further.

Bernard Hill describes a short program to provide a date-stamping facility for all your files, and we conclude with some further hints and tips for Master and Compact users.

MASTER SERIES

ENHANCED PRINTER BUFFER

Derek Baron's modifications to the Printer Buffer, first published in BEEBUG Vol.5 No.7, allow easy control of many printer effects.

Having eventually typed in Tim Powys-Lybbe's printer buffer utility I immediately found it extremely useful, saving the usual wait for the printer. I have now made some additions to the program which I hope will be of benefit to more BEEBUG members especially those who use Interword, Wordwise, ViewStore, or ViewSheet.

USING THE NEW STAR COMMANDS

The changes provide additional star commands to set various printer effects, all of which are listed in table 1. For example, if you are in ViewSheet or ViewStore command mode, typing *CONDENSED followed by *ELITE allows the use of sheets or reports of up to 160 characters across the paper instead of the normal maximum of 80.

The new star commands supported by the extended printer buffer are:

*DEFAULT	Reset printer
*NLQ	Select NLQ mode
*EMPHASISED	Set emphasised text
*CONDENSED	Set condensed text
*ENLARGED	Set double width text
*DOUBLE	Set double strike printing
*ELITE	Set elite pitch
*ITALIC	Select italic printing
*SUPERSCRIPT	Select superscripts
*SUBSCRIPT	Select subscripts
*REVERSE	Set white on black
*VERTICAL	Set double height printing

*All of these, with the exception of *DEFAULT, can be followed by a zero to turn the effect off (for example *NLQ 0).*

*LFEED n	Sets the line spacing to n/72 inches
*PROPORTION	Sets the extra space between each character
*DEFCHAR	Allows the use of the re-defined characters
*CHAR <char>	Redefines a character where <char> is one of the characters in table 2.

Table 1

Another new feature offered by the modified program is a set of user defined characters for common symbols, such as a vertical line and arrows. These new characters are defined both in the computer for use on screen, and at a higher resolution on the

printer. A full list of the new characters is given in table 2. For example, if you wish to print grids with joined vertical lines as well as horizontal lines from your word processor, then *DEFCHAR, *CHAR !, will turn the ! character into a vertical line. *LFEED.8 (note the dot before the number) will cause the lines to be joined vertically, by reducing the line spacing to exactly one character. I found that using *PROPORTION.1 in condensed mode using the maximum 120 characters allowed in Interword gave the correct spacing for typing a grid for use as a Keystrip. Using the commands in word processors which allow star commands to be embedded in the text, for example Wordwise, provides very easy control of the various printer effects.

*The following characters can be redefined, on a printer supporting user defined characters, using *CHAR.*

!	becomes	|
"	becomes	||
#	becomes	Δ
$	becomes	β
%	becomes	⇌
&	becomes	α
>	becomes	→
<	becomes	←

Table 2

The program was written to run with a Citizen 120D printer, which is Epson FX80 compatible, but with a few extra options. For example, the 120D has a white on black print option which is invoked by ESC ~ 2 followed by 0 for off and 1 for on. This is catered for in the program by the *REVERSE command. Similarly, *VERTICAL selects double height printing using ESC h to turn it on and ESC u to turn it off. Another command added to the printer buffer but not supported on most printers is *PROPORTION which uses ESC <space> <n> to add additional space at n/120th of an inch between characters (in printers jargon this space is called the tracking of the character). A further area which might cause problems is the selection of near letter quality mode with *NLQ. This command uses ESC x followed by 0 or 1 to turn NLQ off and on. This will work on most printers but not all. It will therefore be necessary to consult your printer manual for the actual codes used by a particular printer.

INSTALLING THE EXTENDED PRINTER BUFFER

To enter the extra lines that make up the modifications you should load in the original program first. It is very important that the line numbering in this is exactly as it was in the original article. The new lines listed here should then be typed in exactly as they are printed. Some of the new lines are additional to the original listing, while others replace lines already there. Once all the changes have been made, the program should be saved (under a different name from the original just in case something disastrous happens), and then run as described in the original article, to install the software ready for use.

As written, the new version of the program cannot be put into an EPROM because line 1985 defines a flag that is changed when the program is running. Obviously this flag can't be changed if it is in EPROM. To overcome this it is necessary to change the address of this flag to somewhere in main memory. To do this first delete line 1985 and then add the line:

```
25 onflag = &8F
```

The choice of address here is up to you, but &8F is not normally used by any software. If this causes any problems the address can be changed to that of any other safe location in RAM. For example, location &60 is safe if the program will only ever be entered from Basic. Once this modification has been made the whole program can be put into EPROM and permanently installed in your computer.

Each command implemented in the program has a separate routine to handle it, and in most cases another routine to turn the effect off again. For example the *ENLARGE command is

handled by lines 1900 and 1905. To cater for printers which need different control codes it is necessary to change each of these individual routines to send the alternative codes to the printer.

```
  34 .Copyright BRK:EQUS"(C) BEEBUG 198
8 Additions by D.N.Baron.":BRK
  530 EQUS"  Buffoff":EQUB 13:EQUS " Cle
arbuff":EQUB 13:EQUS " TestBuff":EQUB 13
  532 EQUS"  Default":EQUB 13:EQUS "  NL
Q":EQUB 13:EQUS "  Emphasised":EQUB 13:E
QUS "  Condensed":EQUB 13:EQUS "  Enlarg
ed":EQUB 13
  534 EQUS"  Double":EQUB 13:EQUS "  Eli
te":EQUB 13:EQUS "  Italic":EQUB 13:EQUS
 "  Superscript":EQUB 13:EQUS "  SubScri
pt":EQUB 13:EQUS "  Reverse":EQUB 13:EQU
S "  Vertical":EQUB 13:EQUS "  LFeed n":
EQUB 13:EQUS "  Proportion n"
  536 EQUB 13:EQUS"  Char":EQUB 13:EQUS
"  DefChar":EQUW &D
  642 EQUW default:EQUW nlq:EQUW emphasi
se:EQUW condensed:EQUW enlarged:EQUW dou
ble:EQUW elite:EQUW italic:EQUW super:EQ
UW subscript:EQUW reverse:EQUW vertical:
EQUW linefeed:EQUW proportion
  644 EQUW char:EQUW defchar
 1860 .default JSR send:LDA #64:JSR Oswr
ch:JMP endsend
 1870 .nlq JSR checkoff:BCS nlqoff:JSR s
end:LDA #ASC"x":JSR Oswrch:LDA #1:JSR Os
wrch:LDA #49:JSR Oswrch:JMP endsend
 1875 .nlqoff JSR send:LDA #ASC"x":JSR O
swrch:LDA #1:JSR Oswrch:LDA #48:JSR Oswr
ch:JMP endsend
 1880 .emphasise JSR checkoff:BCS emphas
iseoff:JSR send:LDA #ASC"E":JSR Oswrch:J
MP endsend
 1885 .emphasiseoff JSR send:LDA #ASC"F"
:JSR Oswrch:JMP endsend
 1890 .condensed JSR checkoff:BCS conden
sedoff:JSR checkprinteron:LDA #2:JSR Osw
rch:LDA #1:JSR Oswrch:LDA #15:JSR Oswrch
:JMP endsend
 1895.condensedoffJSRcheckprinteron:LDA
#2:JSR Oswrch:LDA #1:JSR Oswrch:LDA  #18
:JSR Oswrch:JMP endsend
 1900 .enlarged JSR checkoff:BCS enlarge
doff:JSR send:LDA #ASC"W":JSR Oswrch:LDA
 #1:JSR Oswrch:LDA #49:JSR Oswrch:JMP en
dsend
 1905 .enlargedoff JSR send:LDA #ASC"W":
JSR Oswrch:LDA #1:JSR Oswrch:LDA #48:JSR
 Oswrch:JMP endsend
 1910 .double JSR checkoff:BCS doubleoff
:JSR send:LDA #ASC"G":JSR Oswrch:JMP end
send
 1915 .doubleoff JSR send:LDA #ASC"H":JS
R Oswrch:JMP endsend
 1920 .elite JSR checkoff:BCS eliteoff:J
SR send:LDA #ASC"M":JSR Oswrch:JMP endse
nd
 1925 .eliteoff JSR send:LDA #ASC"P":JSR
 Oswrch:JMP endsend
 1930 .italic JSR checkoff:BCS italicoff
:JSR send:LDA #ASC"4":JSR Oswrch:JMP end
send
 1935 .italicoff JSR send:LDA #ASC"5":JS
R Oswrch:JMP endsend
 1940 .super JSR checkoff:BCS suboff:JSR
 send:LDA #ASC"S":JSR Oswrch:LDA #1:JSR
Oswrch:LDA #48:JSR Oswrch:JMP endsend
 1945 .subscript JSR checkoff:BCS suboff
:JSR send:LDA #ASC"S":JSR Oswrch:LDA #1:
JSR Oswrch:LDA #49:JSR Oswrch:JMP endsen
d
 1950 .suboff JSR send:LDA #ASC"T":JSR O
swrch:JMP endsend
 1951 .reverse JSR checkoff:BCS reverseo
ff:JSR send:LDA #ASC"r":JSR Oswrch:JMP e
ndsend
 1952 .reverseoff JSR send:LDA #ASC"t":J
SR Oswrch:JMP endsend
 1955 .vertical JSR checkoff:BCS vertica
loff:JSR send:LDA #ASC"h":JSR Oswrch:JMP
 endsend
 1960 .verticaloff JSR send:LDA #ASC"u":
JSR Oswrch:JMP endsend
 1965 .linefeed JSR number:CMP #0:BEQ li
nefeedoff:PHA:JSR send:LDA #ASC"A":JSR O
swrch:LDA #1:JSR Oswrch:PLA:JSR Oswrch:J
MP endsend
 1970 .linefeedoff JSR send:LDA #ASC"A":
JSR Oswrch:LDA #1:JSR Oswrch:LDA #12:JSR
 Oswrch:JMP endsend
 1975 .send JSR checkprinteron:LDA #2:JS
R Oswrch:LDA #1:JSR Oswrch:LDA #27:JSR O
swrch:LDA #1:JSR Oswrch:RTS
 1980 .checkprinteron LDA #117:JSR Osbyt
```

```
e:TXA:AND #1:STA onflag:RTS
 1985 .onflag EQUB 0F
 1990 .endsend LDA onflag:BNE endsend1:L
DA #3:JSR Oswrch:.endsend1 JMP EndZ
 1995 .checkoff INY:LDA (Oscomd),Y:CMP #
ASC" ":BEQ checkoff:CMP #&D:BEQ endcheck
off:CMP #ASC"0":BNE endcheckoff:SEC:RTS:
.endcheckoff CLC:RTS
 2000 .char INY:LDA (Oscomd),Y:CMP #ASC"
 ":BEQ char:CMP #ASC"!":BNE char1:LDX #0
:JSR printer:LDX #0:JSR comp:JMP EndZ
 2010 .char1 CMP #ASC">":BNE char2:LDX #
11:JSR printer:LDX #8:JSR comp:JMP EndZ
 2020 .char2 CMP #34:BNE char3:LDX #22:J
SR printer:LDX #16:JSR comp:JMP EndZ
 2030 .char3 CMP #ASC"#":BNE char4:LDX #
33:JSR printer:LDX #24:JSR comp:JMP EndZ
 2040 .char4 CMP #ASC"%":BNE char5:LDX #
44:JSR printer:LDX #32:JSR comp:JMP EndZ
 2050 .char5 CMP #ASC"&":BNE char6:LDX #
55:JSR printer:LDX #40:JSR comp:JMP EndZ
 2060 .char6 CMP #ASC"$":BNE char7:LDX #
66:JSR printer:LDX #48:JSR comp:JMP EndZ
 2070 .char7 CMP #ASC"<":BNE char8:LDX #
77:JSR printer:LDX #56:JSR comp:JMP EndZ
 2080 .char8
 2100 LDA #7:JSR Oswrch
 2200 .charend JMP EndZ
 2300 .printer:PHA:TXA:PHA:JSR checkprin
teron:PLA:TAX:LDA #2:JSR Oswrch:LDA #1:J
SR Oswrch:LDA #27:JSR Oswrch:LDA #1:JSR
Oswrch:LDA #38:JSR Oswrch:LDA #1:JSR Osw
rch:LDA #0:JSR Oswrch:LDA #1:JSR Oswrch:
PLA:PHA:JSR Oswrch:LDA #1:JSROswrch
 2305 PLA:PHA:JSR Oswrch:LDA #1:JSR Oswr
ch:LDA #139:JSR Oswrch
 2310 LDY #11:.printer1:LDA #1:JSR Oswrc
h:LDA prindata,X:JSR Oswrch:INX:DEY:BNE
printer1:LDA onflag:BNE printer2:LDA #3:
JSR Oswrch:.printer2 PLA:RTS
 2400 .compPHA:TXA:PHA:LDA#0:LDX#1:JSROs
byte:CPX#3:BNEnotmaster:LDA#23:JSR Oswrc
h:PLA:TAX:PLA:JSR Oswrch:LDY#8:.comp1 LD
Acompdata,X:JSROswrch:INX:DEY:BNE comp1:
RTS:.notmasterPLA:PLA:RTS
 2450 .compdata EQUS "        ":EQUS CHR
$0+CHR$8+CHR$4+"~~"+CHR$4+CHR$8+CHR$0
 2460 EQUS "$$$$$$$$":EQUS CHR$0+CHR$24+
CHR$24+"$B"+CHR$129+CHR$255+CHR$0
 2470 EQUS CHR$4+CHR$2+CHR$255+CHR$0+CHR
$255+"@ "+CHR$0
 2480 EQUS CHR$0+"2M"+CHR$136+CHR$136+"M
2"+CHR$0:EQUS "xHHxHHx@"
 2490 EQUS CHR$0+" @"+CHR$254+CHR$254+"@
 "+CHR$0
 2500 .prindata EQUS STRING$(5,CHR$0)+CH
R$255+STRING$(5,CHR$0)
 2510 EQUS STRING$(3,CHR$24+CHR$0)+CHR$2
4+CHR$129+"Z$"+CHR$24
 2520 EQUS STRING$(2,CHR$0)+STRING$(2,CH
R$255)+STRING$(3,CHR$0)+STRING$(2,CHR$25
5)+STRING$(2,CHR$0)
 2530 EQUS CHR$3+CHR$4+CHR$9+CHR$16+CHR$
33+CHR$64+CHR$33+CHR$16+CHR$9+CHR$4+CHR$
3
 2540 EQUS CHR$0+CHR$20+CHR$2+CHR$21+CHR
$0+CHR$20+CHR$0+"T "+CHR$20+CHR$0
 2550 EQUS CHR$0+CHR$24+"$B$"+CHR$24+"$B
$"+CHR$0+CHR$0
 2560 EQUS CHR$0+CHR$0+CHR$255+CHR$0+CHR
$146+CHR$0+CHR$146+CHR$12+CHR$96+CHR$0+C
HR$0
 2570 EQUS CHR$24+"$Z"+CHR$129+CHR$24+ST
RING$(3,CHR$0+CHR$24)
 2800 .defchar JSR checkprinteron:LDA #2
:JSR Oswrch:LDY #9:LDX #0:.defchar1 LDA
#1:JSR Oswrch:LDA defdata,X:JSR Oswrch:I
NX:DEY:BNE defchar1:LDA onflag:BNE defch
ar2:LDA #3:JSR Oswrch:.defchar2 JMP EndZ
 2810 .defdata EQUS CHR$27+":"+STRING$(3
,CHR$0):EQUS CHR$27+"%"+CHR$1+CHR$0
 2900 .proportion JSR number:PHA:JSR sen
d:LDA #ASC" ":JSR Oswrch:LDA #1:JSR Oswr
ch:PLA:JSR Oswrch:JMP endsend
 2950 .number INY:LDA (Oscomd),Y:CMP #AS
C" ":BEQ number:CMP #ASC"0":BCC number0:
CMP #58:BCS number0:SEC:SBC #48:TAX:INY:
LDA (Oscomd),Y:CMP #&D:BEQ onenum:CMP #A
SC"0":BCC number0:CMP #58:BCS number0:SE
C:SBC #48:CLC:ADC nums,X:RTS
 2952 .onenum TXA:RTS
 2960 .number0 LDA #0:RTS
 2970 .nums EQUB 0:EQUB 10:EQUB 20:EQUB
30:EQUB 40:EQUB 50:EQUB 60:EQUB 70:EQUB
80:EQUB 90
 3000 ]:NEXT:*K.9*SRLOAD Buffer 8000 4|M
 3010 *K.8OSCLI("S.Buffer "+STR$~Q%+" "+
STR$~O%+" 0 FFFF8000")|M
```



Use this short routine by Bernard Hill to date-stamp your data files.

There are many reasons for wanting to date-stamp a data file. My need arose because I use the "|D" option in View to automatically insert the current date into the heading of all correspondence. This is fine except when you want to find out the date of a particular file, since all you have is the "|D" code, which tells you nothing.

The accompanying program gènerates a short piece of code called "STAMP" which will sit in the Library directory of your disc until you have need of it. After saving a correspondence file, just type:

```
*STAMP filename
```

either from Basic, or from within View or wherever, and your file will automatically be date-stamped. If you subsequently type:

```
*INFO filename
```

the redundant load and execution addresses will be replaced by the date, as the accompanying example illustrates.

```
>*INFO LETTER
 $.Letter  004556  014F4F  00064F  01D
>*STAMP LETTER
>*INFO LETTER
 $.Letter  001504  001988  00064F  01D
```

Date stamping a letter on 15th April 1988

Wordwise users who make use of the automatic dating routines published some time ago in these pages (BEEBUG Vol. 6 Nos. 2 and 6) will also find the routine helpful. And it will work just as well for dating datafiles.

```
 10 REM Program Date Stamper
 20 REM Version B 0.4 .>DateStamp4
 30 REM Author  Bernard Hill
 40 REM BEEBUG  April 1988
 50 REM Program subject to copyright
 60 :
 70 zpg=&A8
 80 FOR opt=0 TO 3
 90 P%=&900
100 [ OPT opt
110 LDA #1:LDX #zpg:LDY #0
120 JSR &FFDA
130 LDA #14:LDX #tpb MOD 256
140 LDY #tpb DIV 256
150 JSR &FFF1
160 LDA tpb+1:STA load
170 LDA tpb+2:STA load+1
180 LDX #ospb MOD 256
190 LDY #ospb DIV 256
200 LDA zpg:STA ospb
210 LDA zpg+1:STA ospb+1
220 LDA #2
230 JSR &FFDD
240 LDA tpb:STA exec
250 LDA #&19:STA exec+1
260 LDA zpg:STA ospb
270 LDA zpg+1:STA ospb+1
280 LDA #3:JSR &FFDD
290 RTS
300 :
310 .tpb
320 EQUD 1
330 EQUD 0
340 .ospb
350 EQUW 0
360 .load
370 EQUD 0
380 .exec
390 EQUD 0
400 EQUD 0
410 EQUD 0
420 ]:NEXT
430 OSCLI("SAVE STAMP 900 "+STR$~P%)
```

Hints Hints Hints

LARGE CURSOR FOR THE EDITOR

Yo Tomita

If you have found the Master's Editor cursor too small, then this tip will help. The following EXEC file will set up a command called ECursor which can be called in command line mode after pressing function key f1. It will provide a large block cursor, and change the text colour to yellow. Once installed the routine can also be called by pressing key f1 twice. This is useful, as the cursor and colour are lost after a mode change. When creating the file, you can use the Editor itself, and it is a good idea to save the file in the Library directory of your disc, for easy access.

```
*| >ECursor
*| Cursor for Editor
*KEY1"|W|@|J|@|@|@|@|@|@|@|@|S|G|C|@|@|@|@
|@|@|W|A|C|@|@|@|@|@|@|@"
*FX138,0,129
```

FREE ROM FOR MASTER COMPACT OWNERS

M.D.Chesher

Tucked away in the Library directory of the Compact Welcome disc is a file called Spriter. If it is loaded into sideways RAM using say:

```
*SRLOAD Spriter 8000 4 Q
```

you will find that you now have the missing Acorn Sprites in the form of SPRITE ROM 0.6.

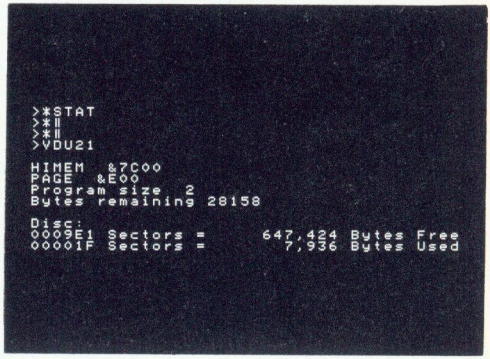
*HELP will show that additional information is available under SPRITES and GRAPHICS. Using *HELP on these words reveals that the new commands are identical to those in the GXR ROM, except that this version stores its sprite definitions within its own RAM area, so there is no need for reserving sprite space with *SSPACE. To get the most out of this new "ROM", you ideally need the GXR manual, but failing this, the detail from *HELP will get you by.

IMPLEMENTING *STATUS

Lee Calcraft

If you save the following EXEC file in your Library directory under the name STAT (not STATUS, since this name is already used), then every time that you type the command *STAT, you will get a display similar to that shown.

```
*| STATUS DISPLAY
*|
VDU21
VDU6:P.'"HIMEM &";~HIMEM:P."PAGE &";~PAGE:
P."Program size ";TOP-PAGE:P."Bytes remain
ing ";HIMEM-TOP''"Disc :":OS.("FREE")
```



PREVENTING UNWANTED DISC READS

Dennis Weaver

When a star command is issued, the operating system passes it first of all to each sideways ROM in turn, and if none claims it, it is offered to the current filing system. In most cases this will be the DFS or ADFS, and you will hear the disc drive whirr. If this search is unfruitful, an error message is given. The additional disc search can be avoided if the star command is given in an abbreviated form, and terminated with a full stop. This can be useful under certain circumstances. B

THE SOLIDISK REAL TIME CLOCK

Bernard Hill follows his appraisal last month of the DABS Press Master Emulation ROM with a quick look at the complementary Real Time Clock module from Solidisk.

Product	Real Time Clock
Supplier	Solidisk Technology 17 Sweyne Avenue, Southend-on-Sea, Essex SS2 6JQ.
Price	£29.00 inc VAT

In the Master Emulation ROM (MER) review last month I mentioned MER's use of Solidisk's RTC. What about this product? Does it make a good investment when bought alone? What would it give you?

With a (rather meagre) manual, the RTC is supplied as a small board about 1.5" x 3" which plugs into a vacant ROM socket (plug it in the one on the right on the BBC board and it will overhang the other three without interfering with them). It has a trailing connection to a Nicad PCB battery, which Solidisk suggest you can solder into position on the board if you have the headroom. The board contains an HD146818 Real Time Clock module, a 16K EPROM and some support circuitry.

Not only does it offer the user a normal *TIME function - and again TIME$ (Basic IV) and ⁞D (View3), but also a full *CONFIGURE and *UNPLUG/*INSERT/*ROMS suite all by itself.

CONFIGURATION BYTES

The RTC module, as in the Master, contains 50 bytes, but the exact meaning of each of these bytes is left to the software designer. Dabs Press wisely chose the same meanings as Acorn's Master, but Solidisk have not. This means that when disabling MER with *MODELB, RTC then takes over the configuration bytes, can't make sense of them and so reconfigures them to their item defaults. You must thus reconfigure them all over again. You could of course keep an EXEC file available for use with both systems, but one slight complication is that Solidisk's RTC expects *CONFIGURE FILE to be followed by the Filing System Number, as opposed to the Master's Filing System ROM Number. For a reason I don't understand, neither MER nor RTC is compatible with the PANEL options of BEEBUG's Master ROM either.

OTHER FEATURES

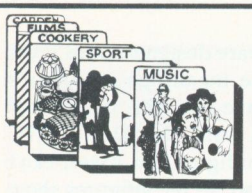
In addition to the above features, the RTC ROM also accepts star commands to make various types of alarm available. At a set time you can have a simple beep, print out the contents of a file or execute a command of up to 34 characters (held in CMOS RAM). These will occur even if other programs are running!

Finally, you have a command to leave a digital clock running at the top right of the screen as well as support for full low-level OSBYTE 161/162 and OSWORD 14/15 calls.

CONCLUSIONS

So what are your options? £30 for RTC, £15-£20 for MER, or both? That depends on what you are looking for. If it's secure ROM unplugging and configuration then RTC gives you that, plus a real time clock and alarm. But if you make frequent use of multiple filing systems then the temporary filing system of MER is a godsend. To buy both is expensive, and you are left with an unconfigured Beeb when you disable MER. It's your choice. B

NOTE:
At the time of going to press we have been informed that Solidisk are discontinuing the Real Time Clock module. Some supplies should still be available for a short while though.



VIDEO CATALOGUER

(Part 2)

Keith Sumner adds the final touches to his Video Cassette Cataloguer with routines to handle searching, sorting and printing of records.

Here we present the additional procedures to add to the video cassette cataloguer published last month. Load the program from last month and type in the additional instructions listed here before resaving. If you managed to catalogue your collection of cassettes last month, this month's procedures will now allow you to search and sort the entries, and print lists and labels to make quick referencing of video tapes easier. Although each feature is reasonably self explanatory a quick description is given here.

```
                SEARCH


You may search :

1. By Film title
2. By Section of title
3. By Tape number
4. By Film category
5. By Min. time left on a tape

6. Exit to main menu

Option no. :
```

OPTION 3 - SEARCHING

The search option will allow the tapes to be searched using any one of the five fields. When you have selected this option you will be presented with a menu. Press the number keys 1 to 5 to choose the relevant field or key 6 to return to the main menu. You will then be asked for the search data. If a match is found it will displayed. You can then search again or select 6 to return to the main menu.

OPTION 4 - SORTING

When this option is selected you will need to be patient while all the entries are sorted into numeric order by the tape numbers. This may take anything up to a minute depending upon the number of entries in the catalogue. You will then be brought back to the main menu automatically.

OPTION 8 - HARD COPY

When this option is selected you will be given the chance to get you printer ready. When all is well press 'Y' and a complete list of all entries will be printed. You will then be returned to the main menu.

No.	Type	Counter	Dur	Cat	Time	Title
001	240	2115-5518	120	SPY	120	The ODESSA File
003	180	0000-1650	090	FLM	000	Magic
003	180	1650-4295	090	FLM	000	Ministry of Fear
004	180	0000-1650	090	FLM	000	Pendulum
004	180	1650-4295	090	FLM	000	House of Wax
005	120	0000-3238	120	FLM	000	Charley Varrick
006	180	0000-4295	180	WAR	000	The Deer Hunter
007	240	0000-1765	103	SPY	000	Our man in Havana
007	240	1767-5518	137	FLM	000	Khartoum
008	180	0000-1780	096	FLM	084	The Domino Principle
009	180	0000-0465	028	HUM	152	Butterflies
010	240	2115-5518	120	FLM	120	The Eagle has landed
011	180	0000-3585	162	FLM	018	Tess
012	240	0000-2110	120	FLM	000	Winning
012	240	2110-5518	120	FLM	000	"10"

OPTION 9-PRINT LABEL

Use this option to print a small label (a standard 36mm by 89mm label will surfice) to stick to each cassette. Make sure that the label is aligned correctly in the printer before entering a three digit tape number. The label will be printed in the smallest text possible. Just press Return to return to the main menu.

If you continue to update the video cataloguer regularly you will be able to make the most of your video tape collection, and this will prove to be a very useful program.

```
Tape 004 (E-180)
0000-1650 090 Pendulum  (FLM)
1650-4295 090 House of Wax  (FLM)
Time remaining : 000
```

```
 1175 IFK=3 PROCsearch ELSE IF K=4 PROCs
ort
 1195 IFK=8 PROCprinter ELSE IF K=9 PROC
label
 3920 :
 3930 DEFPROCsearch
 3940 FO=0:PROChead(13,mg$+"SEARCH")
 3950 PRINT'''"You may search :"'
 3960 FORZ=1TO5:PRINT;Z;". By ";B$(Z-1):
NEXT
 3970 PRINT'"6. Exit to main menu"
 3980 PRINT'"Option no. : ";:Z=GET-48
 3990 IFZ<10RZ>6 PROCsearch
 4000 IFZ=6 ENDPROC
 4010 PRINT''B$(Z-1);:INPUTA$:FORA%=1TOZ
%
 4020 IFZ=1 AND A$=LEFT$(T$(A%),LENT$(A%
)-3) PROCa
 4030 IFZ=2 AND INSTR(LEFT$(T$(A%),LENT$
(A%)-3),A$)<>0 PROCa
 4040 IFZ=3 AND LEFT$(N$(A%),3)=A$ PROCa
 4050 IFZ=4 AND RIGHT$(T$(A%),3)=A$ PROC
a
 4060 IFZ=5 AND VALA$<=VALRIGHT$(N$(A%),
3) PROCa
 4070 NEXT:PROCnomatch:PROCsearch
 4080 ENDPROC
 4090 :
 4100 DEFPROCsort:PROChead(8,fl$+cy$+"SO
RT")
 4110 LOCALF%,I%,S%,T%,Z$,P%
 4120 S%=2^INT(LOG(Z%-1)/LOG(2))
 4130 FORP%=1TO2:IFP%=2 S%=1
 4140 REPEAT:T%=Z%-S%:REPEAT
 4150 F%=FALSE:FORI%=1TOT%
 4160 IFP%=1 AND LEFT$(N$(I%),3)>LEFT$(N
$(I%+S%),3)PROCswap
 4170 IFP%=2 AND LEFT$(N$(I%),3)=LEFT$(N
$(I%+S%),3) AND MID$(N$(I%),9,4)>MID$(N$
(I%+S%),9,4)PROCswap
 4180 NEXT
```

```
 4190 T%=T%-1:UNTILNOTF%:S%=S%DIV2
 4200 UNTILS%=0:NEXT:ENDPROC
 4210 :
 4220 DEFPROCprinter
 4230 PROChead(12,gr$+"PRINTOUT")
 4240 PRINT"Is the printer on line (Y/N)
 ?":*FX21,0
 4250 REPEAT:A$=GET$:UNTILINSTR("Yy",A$)
>0
 4260 VDU2:PRINTTAB(20)"VIDEO INDEX PRIN
TOUT"'''
 4270 PRINT"  No.  Type  Counter    Dur
 Cat  Time  Title"
 4280 FORA%=1TOZ%:PROCsplit:t$=LEFT$(t$,
50)
 4290 PRINTTAB(2)N$+"  "+L$+"    "+R$+"
"+d$+"  "+C$+"  "+D$+"   "+t$
 4300 NEXT:VDU3
 4310 ENDPROC
 4320 :
 4330 DEF PROClabel
 4340 REPEAT
 4350 PROChead(11,bl$+"Print Label")
 4360 PRINT''"Tape No. ";:a$=FNinpt(1,3,
0): IF a$="" UNTIL TRUE: ENDPROC
 4370 E=0
 4380 FOR A%=1 TO Z%
 4390 IF LEFT$(N$(A%),3)=a$ E=E+1: A(E)=
A%
 4400 NEXT
 4410 IF E=0 PROCnomatch: ENDPROC
 4420 VDU 2,1,15,1,27,1,83,1,0
 4430 A%=A(1): PROCsplit
 4440 PRINT '"Tape ";N$;" (E-";L$;")"
 4450 FOR X = 1 TO E
 4460 A%=A(X):PROCsplit
 4470 PRINT R$;" ";d$;" ";t$;" (";C$;")"
 4480 NEXT
 4490 PRINT "Time remaining : ";D$
 4500 VDU 1,27,1,64,3
 4510 UNTIL FALSE
```

B

ARM WRESTLING (continued from page 11)

As it is now unlikely that we shall see further software development of significance for Acorn's 8-bit machines, there is increasing pressure to upgrade to more powerful systems. Undeniably, there are problems and niggles to overcome when upgrading from the 8-bit workhorse we all have come to love (and hate at times), but in the Archimedes, Acorn have provided us with an eminently programmable and accessible machine. It offers both familiarity coupled with the power to emulate other computer systems. The Archimedes has the potential to carry on well into the 1990s - a potential which software houses are beginning to exploit.

B

SUPPLIERS	
Acorn Computers	(0223) 214411
BEEBUG	(0727) 40303
Brainsoft	01-486 0321
Clares Micro Supplies	(0606)48511
Computer Concepts	(0442) 63933
Resource	(0302) 63800
Watford Electronics	(0923) 37774

BEEBUG WORKSHOP Using Printers (pt.2)

We continue our series of Workshops on the use of printers with an examination of Escape sequences and their function.

In last month's Workshop we published a table of the printer codes 0 to 31 with their associated (standard) names. Five of the printer codes, namely CR, LF, FF, SI, and ESC, were explained in some detail. Two of the remaining codes provide important facilities so we will deal with these now.

The large majority of Epson compatible printers support the ASCII codes 8 (BS or backspace) and 14 (SO or shift out). The most common use for the backspace code is to provide an alternative method of underlining text to that which we used last month. This is the method that will have to be adopted if your printer does not have an automatic underline facility. Try the following on your printer:

```
10 VDU2
20 PRINT "Normal ";
30 PRINT "/ Underlined";
30 FOR I% = 1 TO 10
40 VDU 1,8: NEXT I%
40 PRINT STRING$(10,"_")
50 VDU3
```

The word 'Underlined' has, of course, been underlined. Ten backspace characters have been sent to the printer using the VDU 1 command so that the print head moves backwards to the beginning of the word 'Underlined'. Ten underline characters are then printed from this position creating the underlined effect. Some daisy wheel printers use this method to provide automatic underlining, but in that case each letter is underlined immediately it has been printed by following the character itself with a backspace /underline sequence.

The SO character (on most Espon compatible printers) temporarily puts the printer into a double width mode of print. The following example demonstrates this:

```
10 VDU2
20 PRINT "Normal ";
30 VDU1,14
40 PRINT "Double width"
50 VDU3
```

Half of the line is in normal print, the rest is in double width print. The printer is only put into double width mode temporarily. It is cancelled either by the beginning of a new line, or by the code 20 (cancel double width) being sent.

USING ESCAPE SEQUENCES

The idea of Escape sequences was introduced last month. On most printers, Escape sequences can be used to provide many different effects. Each Escape sequence uses ASCII 27 (Escape) followed by a code to specify the particular effect, and maybe one or two additional parameters. Here we shall look at Escape sequences to create bold or italic text, select different fonts, print double height characters and select NLQ (near letter quality) print.

NLQ (near letter quality) printing is a printing mode where text is printed in two passes. The quality of print is significantly increased and it is likely that the final copy of most letters and documents will be printed in this mode. The only drawback with this mode is that it is slow, and will wear the ribbon more quickly. Most Epson compatible printers support NLQ with the single exception of the FX-80 itself.

Let us consider how we can ensure printer output is in NLQ mode when required. The manual for the Epson LX800 gives the following information regarding NLQ mode.

ESCx		Select NLQ or draft		
format:	ASCII	ESC	x	n
	DECIMAL	27	120	n
	HEXADECIMAL	1B	78	n
	KEYBOARD	Ctrl [	x	n

The manual states that n should be 1 to select NLQ, or 0 to select draft mode. In this case the relevant codes that need to be sent to the printer to select NLQ mode are quite clear (i.e. 27,120,1), but not all manuals are this easy to read. A common format used in printer manuals is:

```
LPRINT CHR$(27)+"x"+CHR$(1)
```

If this is how your manual documents each Escape sequence, the necessary codes can be extracted easily if you just remember to use the number in the brackets following CHR$ statements, and look up the ASCII code for any character enclosed in quotes. A list of characters and ASCII numbers can be found in the back of either your printer manual or the BBC User Guide.

An alternative way of determining the ASCII code for a character is by typing the following on your BBC:

```
PRINT ASC("x")
```

Where x can be replaced by any printable character. The Escape sequence for NLQ mode is included in the following listing to illustrate how the final BBC coding will look. Make sure that your printer is 'on line' and that there is paper in it before running the program.

```
 10  VDU2
 20  REM Codes for bold text.
 30    VDU1,27,1,69: PRINT "Bold text"
 40    VDU1,27,1,70
 50  REM Codes for italic text.
 60    VDU1,27,1,52: PRINT "Italic text"
 70    VDU1,27,1,53
 80  REM Codes for Elite and Pica font.
 90    PRINT "Pica font"
100    VDU1,27,1,77: PRINT "Elite font"
110    VDU1,27,1,80: PRINT "Pica font again"
120  REM Codes for Superscript chars.
130    VDU1,27,1,83,1,0: PRINT "Superscript"
140    VDU1,27,1,84
150  REM Codes for Near Letter Quality.
160    VDU1,27,1,120,1,1
170    PRINT "Near Letter Quality"
180    VDU1,27,1,120,1,0
190    PRINT "End of demonstration"
200  VDU3
```

The resultant printout will demonstrate each Escape sequence in turn. If a particular control sequence does not appear to work on your printer it may be because that feature is unavailable. However, it is more likely that your printer expects different control codes to those above. You should consult your printer manual and correct the listing appropriately. The program has been organised so that it is easy to determine the Escape sequences that turn the feature on and off. Most Epson compatibles support the fonts Elite and Pica, of which Pica is the default.

A rather neat feature that is provided on many printers, including the Kaga Taxan, is the ability to perform a line feed in the reverse direction. If you have a printer that supports reverse line feeding, try the following program.

```
10 VDU2
20 FOR I = 10 TO 20
30 PRINT TAB(I);"*"
40 NEXT
50 PRINT
50 FOR I = 21 TO 33
60 VDU 1,27,1,106,1,36
70 PRINT TAB(I);"*";
80 NEXT
90 VDU3
```

The resulting printout will resemble a large letter 'V' but you should notice that the program prints the first stroke in the downwards direction, and then the second slant in the upwards direction (using the reverse line feed). The program is extremely simple, but notice the codes required to line feed backwards. The Escape sequence is as you would expect, but the last number n must specify the line spacing in the form n/216 of an inch. Thus the number 36 represents one character position because there are 6 lines per inch in standard printing mode. You might like to try other effects using reverse line feed.

In part three of this series of workshops we will examine the DIP switch settings inside the printer, deal with other national character sets, and show how to combine various effects to produce attractive documents.



THE COMMS SPOT

This month in the Comms Spot,Peter Rochford takes a look at the VASSCOM network, high speed modems, Epnitex, what's new on Prestel and Micronet, and our own BEEBUG database within Micronet.

VASSCOM

Firstly, a few words about the much talked about VASSCOM network which BT is setting up. This has yet to come on-line officially. The VASSCOM network will link all BT's on-line information and mess-aging services together and allow users to dial just one local number to gain access to any service. It will also allow users to communicate at a variety of baud rates, 300/300 (V21), 1200/75 (V23), 1200/1200 (V22) and 2400/2400 (V22bis).

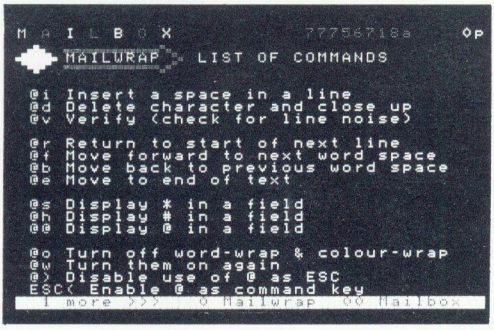
The high speed service is something I personally await eagerly. Some Prestel nodes are already working with multi-baud rates. You can check for this by dialling up your local Prestel access number and listening for the changing carrier tones.

V22BIS MODEMS

You may feel that there is not much point getting all excited about V22bis, when most of the modems around with this feature cost in excess of £500. Well, the good news is that the price of V22bis modems is about to fall dramatically. Several companies have announced recently that they are to produce a low-cost V22bis modem. Amstrad in particular are going to release a modem card with V22bis capability for their own PC and other IBM compatibles. This will retail at an astonishing £199+VAT. Not much good to Beeb owners though you may well say. Well, knowing Amstrad's past record of producing a good product for a ready market and selling it at a cut-throat price, it will not be long I am sure before Mr Sugar releases a stand-alone version that can be used with any computer.

PRESTEL

The keyword search facility is now fully operational and makes using Prestel that much quicker and simpler. Keyword searching allows you to access a particular page or area without knowing its page number. So, for example, to access BEEBUG's front page you would normally have to obtain its page number from the Prestel directory, or, search for the front page by working your way through a series of menus from an index. If you have the number, you would normally key *800909#. With keyword searching, all you have to do is key *BEEBUG#. This system does not search the whole database to find the page you have asked for but relies on an indexing system. When you key *BEEBUG#, Prestel checks the keyword against its index and routes you to the appropriate page. The system must therefore have that keyword in its index, otherwise it will not be recognised. All the major services and information prov- iders on Prestel are now included in the index with keywords to access their front page or other parts of their database area. You may also, whilst on-line, temporarily name any frame you want, using *S name#, and then recall it with *F name# at any time during that session. Full instructions are found on line by keying *19005#



The much criticised and rather archaic mailboxing facility on Prestel has undergone a revamp recently. In particular, it appears less prone to breakdown than it used to be - which has to be good. If you read my article on the Epnitex on-line system in an earlier Comms Spot you will remember how I waxed lyrically

about the mailboxing facility provided there and its word-processor editing and so forth. Prestel have now decided to go ahead and update their mailboxing facility and introduce new features.

Wordwrap is now provided on all mailbox and response frames (joy of joys!). Those of you who use word processors will know about wordwrap where, if you get to the end of a line and the word does not fit, the computer will shift the whole word automatically to the next line .Previously, on Prestel mailbox frames, when you got to the end of a line and your word did not fit, you had to delete what you had typed, cursor key to the next line and re-type it, which is frustrating, time consuming and boring, or simply leave a messy screen with words split across lines. Now, with the new system, you can type away with speed and abandon and watch your text being perfectly formatted. Bliss!

Other features available now are character insert/delete and the ability to move around the text with simple commands making editing so much easier. All these facilities use command sequences starting with an '@', and details of them can be found by keying *77756#. Let's hope they don't stop there but instead follow the example of Epnitex with their excellent mailbox facilities.

EPNITEX

Whilst on the subject of Epnitex, this system has now been officially launched and is active. Although when I reported on Epnitex in a past Comms Spot I was rather excited about this system, I now have to say that I am disappointed that it has taken so long to materialise. Furthermore, the subscription rate for private users is £100 a year, and there are time charges when used between certain hours. This fact and the lack of national local call access will deter many private individuals from subscribing I feel, myself included.

MICRONET

Micronet has made many changes recently to its pages. Not least of these is a vastly improved news service. The multi-user game Shades is proving very popular now, although it got off to a shaky start. The next major introduction is Teletalk which is described as a'telecon-ferencing system'. In effect it is a chatline type bulletin board where several users can get together and chat by sending messages. This will allow the users to bar other people from their 'room' and anyone wishing to join in will have to 'knock' on the door to request entry. Sounds interesting.

BEEBUG DATABASE ON MICRONET

Some nine months ago I took over editing the BEEBUG area on Micronet from Lee Calcraft. Lee was responsible for the initial setting up of the database and put in some very hard work. Since I took over, the database has continued to develop, but not as fast as I would have liked. We now have the magazine contents for both BEEBUG and RISC USER appearing each month along with two or three reviews taken from the current issue of BEEBUG.

The telesoftware available currently stands at ten programs, and these are changed at the rate of two a month. The reaction to our telesoftware from both members and non-members has been very good indeed, and we currently have two programs in Micronet's Top Ten telesoftware chart. Most of the programs are free of charge for BEEBUG members, whilst non-members have to pay to see a frame containing a password which is needed to run the programs

Our retail section on Micronet has undergone many updates recently, and now caters for owners of the Archimedes. We have a huge range of products available for all machines. These can be ordered via our pages using your credit card, and of course with the usual discount to members

You can join or renew your membership via our pages too. A special extra response frame separate from the retail one has now been set up. It is simple to fill in and covers BEEBUG, RISC User and monthly disc and cassette subscriptions.

I have to admit that the BEEBUG database is not as lively as I would like and in recent months, due particularly to the the launch of RISC User pages, it has not been updated or expanded as much as was originally intended.

Continued on page 67

EXPLORING ASSEMBLER

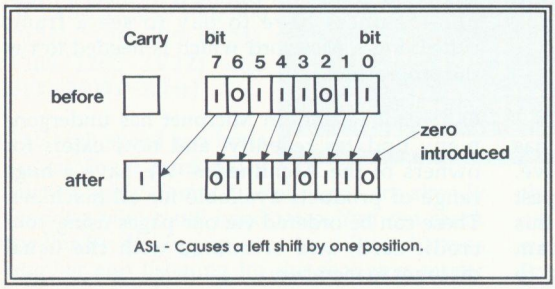


A series for beginners to machine code by Lee Calcraft

This month: Shift, Rotate and Multiply Routines

SHIFT AND ROTATE

In common with most 8-bit processors, the 6502 has no dedicated multiplication or division instructions, and the programmer must implement his own routines to perform these functions. This month we will take a look at multiplication. But before doing so we need to introduce four new instructions. These are ASL and LSR (Arithmetic Shift Left and Logical Shift Right), and ROL and ROR (ROtate Left and ROtate Right). We will take the first pair first.



ASL - Causes a left shift by one position.

ASL AND LSR

The first thing to remember about these two instructions is that although their mnemonics seem to imply that they perform different functions, they are actually identical except for the direction of shift. Both logically shift the binary contents of memory or the accumulator by one position (or one bit) in the corresponding direction. Thus performing an LSR on the binary number 01001100 would yield the result 00100110, while ASL would give 10011000.

All four instructions can take any of five different addressing modes, as follows:

ASL A	Accumulator
ASL &70	Zero Page
ASL &1234	Absolute
ASL &70,X	Zero Page X-Indexed
ASL &1234, X	Absolute X-Indexed

The first case is a little unusual in that the accumulator must be specified using the letter "A", whereas normally the accumulator is implied in other types of instruction. For example the CMP, CPX, CPY set of instructions. It is therefore imperative to avoid the use of the variable **A**, since an instruction to shift its contents would be taken by the assembler to imply an accumulator shift. The other addressing modes are quite straightforward. But you should note when using indexed addressing that the Y register may not be used as an index, even though many other instructions allow both X and Y indexing in Absolute Indexed mode.

Returning to the function of the first two instructions ASL and LSR, there is one further complication. It involves the carry flag, which is treated in both the shift operations as a ninth bit, receiving the contents of the bit shifted out of the register by either instruction. Thus in an ASL, the carry flag receives the *top* bit of the operand, while in an LSR it receives the contents of bit *zero*. .At the opposite end of the register, a zero bit is always shifted in. Thus after 8 successive shifts, any register will contain a zero byte. Both the zero and negative flags are also set as a result of a shift operation, but these behave as they normally do. The zero flag would be set if the result of a shift was zero, and the negative flag would be set if a shift resulted in bit 7 being set.

To take a further example, the binary number 11001100 (204 dec) would become 01100110 (102 dec) after an LSR. The carry flag would be unset, since the lowest bit of the operand is zero, and this is shifted into the carry flag. The

zero flag would be unset, because the result of the operation is greater than zero, and the negative flag would be unset because bit 7 of the result is zero. As you may have noted from the decimal equivalents given here, the result of the shift is exactly half of the operand. Each shift right by one bit position halves the operand (with rounding down), while each shift left doubles it. This fact forms the basis of binary multiplication (and division) as we shall see in a moment. But before this we will take a look at a simpler example of the use of the shift instruction.

BINARY CONVERSION

It is sometimes convenient to be able to display the contents of a register or memory location in binary form. This is easily achieved using the ASL instruction. Take a look at listing 1. If you run this program it will repeatedly ask for a single-byte number, and display its binary equivalent. It works quite simply by loading the number input by the user into the accumulator, and then shifting it left (using ASL in line 150) 8 times in succession. After each shift, the instruction ADC #48 is performed. Since the accumulator is cleared before each shift of the location **value** (line 140), the result of this is to leave the value 49 in the accumulator if the carry flag is set, and 48 if it is not. These two numbers (48 and 49) are the ASCII values of the characters zero and one. So now if we call the operating system routine OSWRCH (line 170), a zero or a one will be displayed according to the value of the currently tested bit. You may remember from the first part of the series that OSWRCH sends the contents of the accumulator to the VDU as an ASCII code. Thus LDA #65:JSR oswrch would display the letter "A" (ASCII 65).

Listing 1

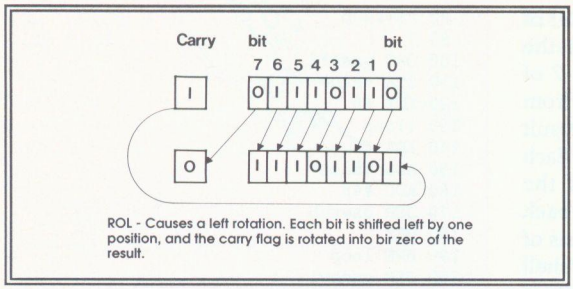
```
 10 REM Dec to Binary
 20 REM Author  Lee Calcraft
 30 REM Version B 0.2
 40 :
 50 value=&70:oswrch=&FFEE:osnewl=&FFE
7
 60 MODE0
 70 FOR pass=0 TO 1
 80 P%=&900
 90 [
100 OPT pass*3
110 STA value
120 LDX #8
130 .loop
140 LDA #0
150 ASL value
160 ADC #48
170 JSR oswrch
180 DEX
190 BNE loop
200 JSR osnewl
210 RTS
220 ]
230 NEXT
240 :
250 REPEAT
260 INPUT'"Number (0-255) "number
270 A%=number
200 JSR osnewl
280 CALL &900
290 UNTIL FALSE
```

```
210 RTS
220 ]
230 NEXT
240 :
250 REPEAT
260 INPUT'"Number (0-255) "number
270 A%=number
280 CALL &900
290 UNTIL FALSE
```

The result, as the routine loops 8 times, is a succession of zeros and ones which reflects the binary equivalent of the number placed in the accumulator, and thus of the number selected by the user. As a final touch, the operating system routine OSNEWL is called (line 200) to terminate the output (OSNEWL sends a carriage return line feed sequence). In many ways the process employed here is similar to that used in the generation of a serial data stream in a serial port such as the Beeb's RS423 port. In that case an 8 bit word is received as a single byte, and shifted out one bit at a time to produce a serial signal - though this particular shifting process is carried out in hardware by a device called a shift register within the 6850 serial interface chip.

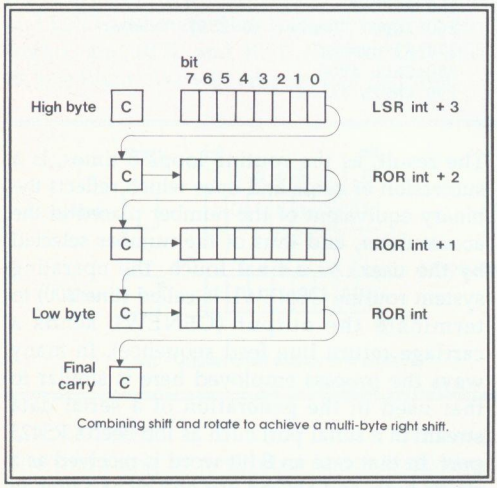
ROR AND ROL

The two sister instructions ROR and ROL are very similar to ASL and LSR except that in addition to the bit lost by the shift being placed



ROL - Causes a left rotation. Each bit is shifted left by one position, and the carry flag is rotated into bir zero of the result.

in the carry flag, the contents of the carry flag are transferred to the other end of the shifted register. Thus if the carry flag is set before an ROL, then the bottom bit of the result will also be set. In an equivalent case using ASL, the bottom bit would always take the value zero. There is thus a complete *rotation* of bits around what is effectively a 9 bit register, and after 9 consecutive rotations, the register will again hold its original contents.



Combining shift and rotate to achieve a multi-byte right shift.

The 6502's shift and rotate instructions complement each other well, as we can see from the example of a multi-byte shift. To shift a four-byte integer held low-byte first at **int** to **int+3**, we can use the following:

```
LSR int+3
ROR int+2
ROR int+1
ROR int
```

With simplicity and elegance, the first instruction shifts the top byte by one bit to the right, and the lowest bit is moved out into the carry flag. Then the second instruction ROR int+2 rotates the next byte. This will shift all bits one place to the right, and place the bottom bit into the carry flag, just as with LSR, but it also picks up the previous contents of the carry flag (containing the lost bit from LSR int+3), and shifts this into its top bit position. The chain is thus maintained, and at the end of the four instructions, the whole four-byte integer will have been shifted one bit to the right. On termination the carry flag will contain the single bit lost by the operation, and this could be used to indicate whether the resultant division by 2 gave a remainder or not.

MULTIPLICATION

As we have seen, multiplying by powers of two is an extremely easy process. It just involves shifting the operand by the appropriate number of places to the left. For example the sequence:

```
ASL A:ASL A:ASL A
```

will multiply the contents of the accumulator by 8. However, creating a general algorithm which can cope with any multiplier requires quite a bit more thought, and the best way to see exactly what is needed is to look at the way in which a long decimal multiplication might be performed on paper. Suppose we want to multiply 125 by 13

```
  125   Multiplicand
  x13   Multiplier
  375   Partial Product 1
 1250   Partial Product 2
 1625   Final product
```

We first of all take the rightmost digit of our multiplier (i.e. 3), and form a so-called partial product. We then repeat this for the next digit (i.e. we multiply by 1), shifting the new partial product left by one column (so that the 125 appears as 1250). Then we add up the partial products to get the final result. We could express this as follows:

```
125x13=(3x10^0x125)+(1x10^1x125).
```

If we are to convert this into an algorithm suitable for the 6502, we must re-work it in binary terms, since all the computer's registers are binary devices. To keep it simple, we will multiply 6 (110) by 5 (101)

```
  110 Multiplicand
 x101 Multiplier
 ----
  110 Partial Product 1
 0000 Partial Product 2
11000 Partial Product 3
-----
11110 Final Product
```

Following exactly the same procedure, we get the binary result 11110, or 30 decimal.

So how would we perform this on the 6502? First of all, it will be easier to add up the partial product as we go along, otherwise we will have to store as many as 8 partial products along the way, and then recall them all just to add them up. As you can see from the simple example above, each partial product is nothing more than a shifted copy of the multiplicand if the corresponding multiplier bit is one, or a string of zeros if it is not. We could therefore use the following sequence:

```
LDA #0
LDX #8
.loop
LSR multiplier
BCC zero
CLC
ADC multiplicand
.zero
ASL multiplicand
DEX
BNE loop
```

It works as follows. The accumulator will hold the running total of partial products which will eventually give the final result. This is cleared at the outset, and the X register, which will act as the loop counter, is set to 8. The multiplier is then shifted right by one bit, and if the carry flag is clear the program skips the next couple of lines. If not, it means that the current bit of the multiplier is high, so we must add in a partial product for this bit. The two instructions **CLC** and **ADC multiplicand** achieve this, by adding the multiplicand to the accumulator. Next the multiplicand is shifted left by one place. This ensures that when the multiplicand is next added in as a partial product, it will be of the correct power.

Listing 2

```
  10 REM Short Multiply
  20 REM Author  Lee Calcraft
  30 REM Version B 0.5
  40 :
  50 multiplicand=&70:multiplier=&71
  60 MODE0
  70 FOR pass=0 TO 1
  80 P%=&900
  90 [
 100 OPT pass*3
 110 STA multiplicand
 120 STX multiplier
 130 LDA #0
 140 LDX #8
 150 .loop
 160 LSR multiplier
 170 BCC zero
 180 CLC
 190 ADC multiplicand
 200 .zero
 210 ASL multiplicand
 220 DEX
 230 BNE loop
 240 RTS
 240 RTS
 250 ]
 260 NEXT
 270 :
 280 REPEAT
 290 INPUT'"Two numbers (0-15) "mult1,m
ult2
 300 A%=mult1:X%=mult2
 310 PRINT USR(&900) AND &FF
 320 UNTIL FALSE
```

The program in listing 2 implements this routine, multiplying a pair of numbers input at the keyboard. As you will quickly perceive, it will only work correctly in cases where the product is less than 256. This is because the product is held in the accumulator, and when this overflows, part of the result is lost.

16 BIT RESULT

If we are to implement a routine which will multiply any pair of 8-bit numbers to generate a 16-bit result, we must use a method which holds only the *active* part of the accumulating final product in the accumulator. This is usually achieved by shifting the accumulating product

right after each stage, rather than shifting the multiplicand *left*. The net result is the same, but now the active part of the accumulating result in the accumulator remains in range. The bit lost by each right shift of the accumulator is itself shifted into a memory location to become part of the least significant byte of the final product. To see how this works out in practice, take a look at listing 3. This again requests two numbers, and displays their product, but now it will work correctly even when both numbers take the full range from 0 to 255.

The central routine is also very similar, except that there is now a location called **lobyte**, which is used to hold the low byte of the result. The new part of the code begins at line 220. First of all the accumulator is rotated right. Because of the way in which ROR works, the bottom bit of the accumulator is shifted into the carry flag. Then **lobyte** is rotated right. This picks up the contents of the carry flag, and inserts it into the top bit of **lobyte**.

As the routine repeats 8 times, the low bits lost from the accumulator ripple through into **lobyte**, until it holds the full value of the low byte of the product. We have no need to zero **lobyte** at the start of the routine since its original contents will be lost after the 8 operations. There is also an important point to note about the instruction ROR A on line 220. At first sight, one might think that LSR A could have been used in its place. But with large multiplicands, there can be overflow errors. This arises because if the operation ADC multiplicand in line 200 overflows into the carry flag, the overflow will be lost if LSR A is used at line 220. But ROR A usefully picks up the carry and inserts it back into the accumulator as the ROR is performed.

Thus on exit from the routine, the accumulator holds the high byte of the product, and the location **lobyte** the low byte. The contents of the accumulator are then stored at **lobyte+1**, and the routine terminates. The Basic program calling the machine code simply displays the combined contents using the indirection operator. If you test the program you should see that it gives correct results for all multiplicands and multipliers between 0 and 255.

As a final thought, you might contemplate the following instruction:

```
MUL R0,R1,R2
```

This single ARM instruction on the Archimedes multiplies the contents of register R1 by register R2, and places the result (which may be as large as 32 bits wide) into register R0.

Listing 3

```
  10 REM 8-bit Multiply
  20 REM Author  Lee Calcraft
  30 REM Version B 0.4
  40 :
  50 multiplicand=&70:multiplier=&71
  60 lobyte=&72:hibyte=&73
  70 MODE0
  80 FOR pass=0 TO 1
  90 P%=&900
  90 P%=&900
 100 [
 110 OPT pass*3
 110 OPT pass*3
 120 STA multiplicand
 130 STX multiplier
 140 LDA #0
 150 LDX #8
 160 .loop
 170 LSR multiplier
 180 BCC zero
 190 CLC
 200 ADC multiplicand
 210 .zero
 220 ROR A
 230 ROR lobyte
 240 DEX
 250 BNE loop
 260 STA hibyte
 270 RTS
 280 ]
 290 NEXT
 300 :
 310 !lobyte=0
 320 REPEAT
 330 INPUT'"Two numbers         "mult1,m
ult2
 340 A%=mult1:X%=mult2
 350 CALL &900
 360 PRINT!lobyte
 370 UNTIL FALSE
```

Next month we will tackle the problem of division. B

MS-DOS TRANSFERS

[Part 1]

Bernard Hill presents an excellent utility that will prove extremely useful for those people who use MS-DOS* at work and BBC micros at home, indeed for any BBC user with access to IBM PCs. The program listed here will transfer disc files from an IBM PC floppy disc to an Acorn DFS disc. Next month we shall be presenting a similar utility that performs the transfer from DFS back to IBM format.

The format of an MS-DOS disc differs considerably from that of a DFS disc. MS-DOS discs are recorded in a double density 40 track format. For this reason your machine must be fitted with a 1770 disc interface to use this program. The Master series and BBC B+ machines are fitted with the 1770 interface as standard. The 1770 interface is an additional upgrade for an ordinary BBC B machine. Another major difference is that MS-DOS formatted discs can be non-contiguous i.e. files are fragmented over the disc surfaces. Furthermore they are organised into clusters of two sectors, one sector on each side of the disc. When the disc is addressed, the computer reads tracks on alternate sides of the disc. This is completely different to filing systems, such as the Beeb's ADFS which access all the tracks on one side of a disc before moving to the second side.

The program presented here is versatile, and may be used on a BBC B, B+, Master and Compact using a single or twin 40 or 80 track disc drive. Since MS-DOS discs use both sides, you will be unable to read IBM PC discs without double sided drives. Because of the complexity of the MS-DOS filing system, and to keep the program reasonably short, I have assumed the following about the source disc:

1. The files to be transferred must be in the root directory (called "\").

2. The disc is assumed to be double sided, 9 sectors per track. If the disc is the obsolete 8-sector type, the optional single-sided, or quad density 15 sector (AT compatible) type, then the program will stop with a warning message.

3. The MS-DOS disc is assumed to be in a self-consistent form, i.e. the MS-DOS utility CHKDSK reports no errors.

CUSTOMISING

To configure the program for your own system you may need to alter some of the statements in lines 120 to 150:

a. Line 120 contains a speed specification. If you have the modern high speed drives then you can set the speed to 0 (the fastest). The slowest value is speed=3. If in doubt try speed=0 and be prepared to alter this should the drives prove unreliable.

b. The DFS disc is assumed to be in drive 0. If you have twin drives, then leave line 140 as D=1 and place the IBM disc in drive 1. If you have a single drive then set D=0 in line 140 and the program will prompt you for disc changes.

c. Line 150 is set for 80 track drives. If you have 40 track then change this line to read track80=FALSE.

d. MS-DOS text files contain a Carriage Return and a Line Feed (&0D, &0A) at the end of every line, whereas BBC text files usually contain only a Carriage Return (&0D). If you are transferring text files set asc% (in line 130) to be TRUE so that the line feeds (&0A) are not copied over. Should you be transferring binary data files then you will need to set asc% to FALSE. Naturally, you will be unable to use programs from MS-DOS on a BBC because the machines have different processors. It is possible to transfer simple Basic programs using this utility but they must be treated as text files and must not be a tokenised format.

RUNNING THE PROGRAM

Having typed in the program (and saved it) place your BBC disc in drive 0 and (assuming you are using twin drives) the IBM disc in drive



1. It will then read the directory and disc map (called the File Allocation Table - FAT) of the IBM format source disc, and present you with a menu of filenames and lengths. Each file name will have an index number. When you are prompted for the appropriate file, enter its index number and it will be transferred. The filename used on the BBC disc will consist of the first 7 characters of the IBM filename provided that the BBC disc does not contain a file of that name already. If it does, the program automatically renames the DFS file by changing the first letter of the name by one character. Press Escape when you have finished transferring, and you will find the IBM files on your DFS disc. If you should have trouble with control characters in IBM files (such as ASCII 12 - form feed) then you may like to filter them out by changing line 380 to:

```
380 IF NOT asc% OR (ch%>31 AND ch%<127)
    OR ch%=9 THEN BPUT#f,ch%
```

This will filter out all non-printing characters with the exception of the Tab character (ASCII 9) which is very common in word processed documents.

Finally, it should be pointed out that the program can also read MS-DOS discs that have been formatted on an Archimedes. If you have a five inch disc drive connected to your Arc, or if you have a three inch drive connected to your BBC, this utility will provide an easy way to transfer PC (MS-DOS) software between the two machines.

MS-DOS is a registered trade mark of the MicroSoft Corporation.

```
   10 REM Program  IBM to BBC transfer
   20 REM Version  B0.5
   30 REM Author   Bernard Hill
   40 REM Beebug   April 1988
   50 REM Program  subject to copyright
   60 :
  100 ON ERROR GOTO 430
  110 MODE7
  120 speed=0
  130 asc%=TRUE
  140 D=1
  150 track80=TRUE: A%=0: X%=1
  160 A%=(USR&FFF4 DIV 256)AND &FF
  170 IF A%>2 Master=TRUE ELSE Master=FA
LSE
  180 bufsiz=10*1024
  190 M=112:DIMsiz%(M),cl%(M),N$(M)
  200 DIM buf% bufsiz,fat 2048,dir 3584
  210 L$="":PROCswitch("IBM")
  220 PROCinitpc(Master)
  230 PROCsetradr(fat):PROCgetsec(1)
  240 IF fat?21=&F9 THEN track80=FALSE:d
irsec=8 ELSE dirsec=6
  250 IF fat?21<>&FD AND fat?21<>&F9 THE
N PRINT "Not IBM format disk":END
  260 REPEAT PROCswitch("IBM")
  270 PROCtitle:PROCdirfat:REPEAT
  280 INPUT"File number (0 to stop): "n%
  290 UNTIL n%>=0 AND n%<=nf
  300 IF n%=0 THEN 410
  310 PROCswitch("BBC"):n$=FNname(n%)
  320 c=cl%(n%)
  330 OSCLI("SAVE "+n$+" 0 "+STR$~siz%(n
%))
  340 f=OPENOUTn$
  350 L%=siz%(n%)-1:IF asc% THEN L%=L%-1
  360 FOR I%=0 TO L%:M%=I% MOD bufsiz
  370 IF M%=0 THEN c=FNfillbuff(c)
  380 ch%=buf%?M%
  390 IF NOT asc% OR ch%<>10 THEN BPUT#f
,ch%
  400 NEXT:CLOSE#f
  410 UNTIL n%=0:MODE 7:END
  420 :
  430 IF ERR<>17 THEN MODE7:REPORT:PRINT
" at line "ERL:END
  440 PRINT:END
  450 :
 1000 DEFPROCtitle:VDU26,12
 1010 T$=CHR$132+CHR$157+CHR$131+CHR$141
+"   BEEBUG PC to BBC transfer"
 1020 PRINTT$'T$'LEFT$(T$,2)+CHR$130+"
     (Drive "+STR$~D+" to drive 0)"
 1030 PRINTCHR$134+" No";TAB(10);"Name";
TAB(27);"Size"
 1040 VDU28,0,24,39,4:ENDPROC
 1050 :
 1060 DEFFNname(n):LOCAL N$,f,c
```

```
 1070 N$=LEFT$(N$(n),7):REPEAT
 1080 f=OPENINN$
 1090 IF f>0 THEN CLOSE#f:N$=CHR$(ASCN$+
1)+MID$(N$,2)
 1100 UNTIL f=0:PRINT"File saved as "N$
 1110 =N$
 1120 :
 1130 DEFFNu(v,@%):v$=STR$v
 1140 IF LENv$<@% THEN v$=STRING$(@%-LEN
v$," ")+v$
 1150 =v$
 1160 :
 1170 DEFFNfillbuff(c):CLOSE#f
 1180 PROCswitch("IBM"):PROCsetradr(buf%
)
 1190 n=0:max=bufsiz DIV 1024
 1200 REPEAT n=n+1:s=FNsecno(c)
 1210 PROCgetsec(s):PROCgetsec(s+1)
 1220 c=FNnclus(c):UNTIL c=0 OR n=max
 1230 PROCswitch("BBC"):f=OPENUPn$
 1240 PTR#f=EXT#f:=c
 1250 :
 1260 DEFFNnclus(n)
 1270 !&70=fat!(3*(n DIV 2)):!&73=0
 1280 IF n MOD 2=0 THEN v=!&70 AND &FFF
ELSE v=(!&71 DIV 16) AND &FFF
 1290 IF v>=&FF7 THEN =0 ELSE =v
 1300 :
 1310 DEFFNsecno(c)=2*c+dirsec+3
 1320 :
 1330 DEFPROCdirfat:PROCsetradr(fat)
 1340 PROCgetsec(2):PROCgetsec(3):PROCge
tsec(4)
 1350 PROCsetradr(dir)
 1360 FOR s=dirsec TO dirsec+6
 1370 PROCgetsec(s):NEXT
 1380 loc=dir-32:nf=0:REPEAT loc=loc+32
 1390 t=loc?11:IF ?loc=0 OR ?loc=&2E OR
?loc=229 OR t AND &18 THEN 1430
 1400 nf=nf+1:loc?11=13:N$(nf)=LEFT$($lo
c,8)+"."+RIGHT$($loc,3):loc?11=t
 1410 siz%(nf)=loc!&1C:cl%(nf)=loc!&1A A
ND &FFFF
 1420 PRINT FNu(nf,3);TAB(10);N$(nf);TAB
(25);FNu(siz%(nf),6)
 1430 UNTIL ?loc=0 OR nf=M:PRINT:ENDPROC
 1440 :
 1450 DEFPROCgetsec(N)
 1460 T=(N-1) DIV 9:S=(N-1) MOD 9 + 1
```

```
 1470 PROCrwsec(T,S,TRUE):ENDPROC
 1480 :
 1490 DEFPROCrwsec(T,S,read)
 1500 *FX143,12,255
 1510 val=rst:IF D=0 THEN val=val OR 1 E
LSE val=val OR 2
 1520 IF T MOD 2=1 THEN val=val OR sel
 1530 ?ctrl=val
 1540 ?flag=1:?cmd=&C+speed:PROCwait
 1550 IF track80 THEN ?datareg=(T DIV 2)
*2 ELSE ?datareg=T DIV 2
 1560 ?flag=1:?cmd=&18+speed : REM seek
 1570 PROCwait
 1580 ?trackreg=T DIV 2:?secreg=S
 1590 ?flag=1:IF read THEN ?cmd=&84 ELSE
 ?cmd=&A6 :REM read/write
 1600 PROCwait:*DISC
 1610 ENDPROC
 1620 :
 1630 DEFPROCwait
 1640 REPEAT UNTIL ?flag=0
 1650 IF (?cmd AND &10)<>0 THEN PRINT"Re
ad error drive "D" track "T" sector "S:E
ND ELSE ENDPROC
 1660 :
 1670 DEFPROCinitpc(Master)
 1680 IF Master THEN wd=&FE28:ctrl=&FE24
:sel=16:dden=&20:rst=4 ELSE wd=&FE84:ctr
l=&FE80:sel=4:dden=8:rst=&20
 1690 cmd=wd:status=wd:trackreg=wd+1
 1700 secreg=wd+2:datareg=wd+3:S=0:T=0
 1710 ?ctrl=rst+D+1:ENDPROC
 1720 :
 1730 DEFPROCsetradr(a)
 1740 FOR opt=0 TO 2 STEP 2
 1750 P%=&D00:[OPT opt : pha
 1760 lda status : and #1 : sta flag
 1770 lda status : and #&1F : cmp#3
 1780 bne exit : lda datareg
 1790 .dest sta a : inc dest+1
 1800 bne exit : inc dest+2
 1810 .exit pla : rti
 1820 .flag brk
 1830 ]:NEXT:S=0:T=0:ENDPROC
 1840 :
 1850 DEFPROCswitch(a$):*FX15,1
 1860 IF D=0 AND L$<>a$ THEN PRINT"Inser
t "a$" disk : press a key":IF GET
 1870 L$=a$:ENDPROC
```

B

POINTS ARISING..POINTS ARISING..POINTS ARISING..POINTS ARISING..

VIDEO CASSETTE CATALOGUER Vol.6 No.9

The following lines were regretably omitted from the end of this program:

```
 3930 DEFPROCde
 3940 temp$=LEFT$(i$,(Y-1)*20+X-2):tem$=
RIGHT$(i$,80-(Y-1)*20-X+1):i$=LEFT$(temp
$+tem$+" ",80):X=X-1:IFX=0 ANDY>1 X=20:Y
=Y-1 ELSEIF X=0 ANDY=1 X=1
 3950 ENDPROC
```

B



Knight Quest

Matt Eastmond has come up with yet another action packed arcade game of superior quality. Venture into ancient times when Knights ruled the land and Sorcery was at its worst.

This is an arcade adventure where you take the role of a Knight on a quest to find the lost crown. The crown is hidden deep in the ruins of a weird and wacky castle inhabited by dangerous monsters and protected by a greedy guardian. Freeing the crown is not an easy task. There are many puzzles to solve, and the monsters must be avoided at all costs.

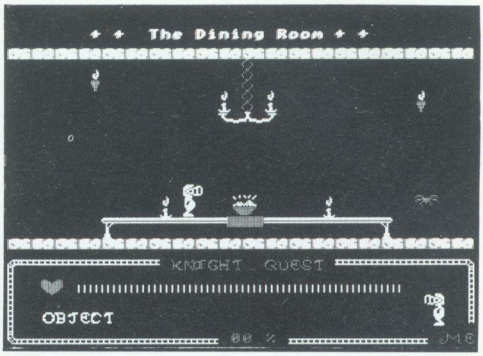
The Knight may be guided from one screen to another using the Z and X keys to move left and right, and the Return key to jump. Objects may be picked up and dropped using the Spacebar but you may only hold one object at a time, and objects may only be dropped or picked up if they are on a red rectangle. Should you be unfortunate enough to touch a wandering monster your energy will be severely reduced. The yellow bar at the bottom of the screen shows how much energy you have left.

The program consists of two listings, each listing should be entered and saved. Note the use of the underline character in some of the DATA statements from line 3110 onwards of the second program. Be careful to save the second listing under the file name 'QUEST1' so that the first listing will CHAIN it correctly. If you do not have shadow memory, try not to insert any more spaces in the listings than is strictly necessary because space is very tight. So tight in fact, that it will not be possible to run the game on a BBC model B under ADFS unless the machine has shadow memory.

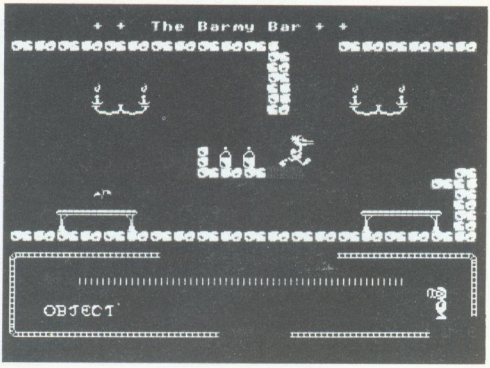
Although the program is somewhat longer than those we usually publish, in our opinion it is well worth the effort of typing it in. It is very seldom that you see such an excellent game with such stunningly smooth graphics written entirely in Basic.

```
   10 REM Program  Knight Quest
   20 REM Version  B1.02
   30 REM Author   Matt Eastmond
   40 REM BEEBUG   April 1988
   50 REM Program  subject to copyright
   60 :
  100 MODE 1:VDU23,1;0;0;0;0;
  110 ?&36D=&B
  120 ENVELOPE 1,1,1,1,1,5,5,5,-5,-1,-1,
-1,120,120
  130 ENVELOPE 2,1,1,1,1,1,1,1,0,0,0,-5,
120,120
  140 ENVELOPE 3,129,13,13,13,100,100,10
0,127,0,0,-2,126,0
  150 ENVELOPE 4,1,0,0,0,0,0,0,126,-1,-1
,-1,126,96
  160 VDU23,192,31,41,73,175,152,240,144
,144,23,193,248,36,38,233,17,15,9,15,23,
194,240,144,144,232,143,82,34,31,23,195,
9,15,9,23,241,74,68,248
  170  VDU23,196,255,68,68,255,0,0,0,0,2
3,197,0,0,0,0,255,68,68,255,23,198,144,1
44,240,144,144,144,240,144,23,199,9,9,15
,9,9,9,15,9
  180 VDU17,2,31,0,23,192,31,39,23,193,3
1,0,30,194
  190 FORT=1TO38:VDU31,T,23,196:NEXT
  200 FORT=1TO35:VDU31,T,30,197:NEXT
  210 FORT=24TO29:VDU31,0,T,198:NEXT
  220 FORT=24TO28:VDU31,39,T,199:NEXT
  230 VDU23,192,128,70,76,88,112,88,76,1
35
  240 VDU23,193,2,101,81,81,73,69,69,66
  250 VDU23,194,15,120,152,24,24,26,124,
128
  260 VDU23,195,30,51,96,64,70,99,35,30
  270 VDU23,196,71,66,66,126,126,66,66,1
03
  280 VDU23,198,60,98,193,129,140,134,67
,57
  290 VDU23,199,0,66,33,33,33,33,33,30
  300 VDU23,200,30,33,36,60,36,32,17,14
  310 VDU23,201,30,33,33,32,144,78,33,30
  320 VDU23,202,31,60,36,4,4,4,4,14
  330 VDU23,203,4,6,5,4,68,164,136,112
  340 VDU23,204,4,12,20,164,68,4,4,2
  350 VDU17,1,31,13,23,32,192,193,194,19
5,196,202,32,32,198,199,200,201,202,32
  360 VDU31,37,30,203,204,200
  370 VDU23,208,0,56,126,127,127,127,127
,63
  380 VDU23,209,0,14,63,127,255,255,255,
254
  390 VDU23,210,31,15,7,1,0,0,0,0
  400 VDU23,211,252,248,240,192,128,0,0,
0
  410 VDU5:GCOL0,1:MOVE90,228:VDU208,209
,10,8,8,210,211,4
```

```
  420 VDU23,213,60,66,195,129,129,195,66
,60
  430 VDU23,214,252,66,34,34,62,34,34,12
4
  440 VDU23,215,15,60,100,4,4,4,36,24
  450 VDU23,216,60,114,97,64,64,64,98,60
  460 VDU17,3,31,3,28,213,214,215,200,21
6,202
  470 VDU23,192,31,127,127,79,71,103,60,
0
  480 VDU23,193,28,254,240,248,254,126,1
26,0
  490 VDU23,194,31,126,126,126,127,127,6
0,0
  500 VDU23,195,28,62,254,254,198,204,12
4,0
  510 VDU23,200,15,63,110,215,219,231,25
5,64
  520 VDU23,201,224,224,24,220,60,254,23
0,254
  530 VDU23,202,30,31,47,76,152,176,240,
112
  540 PROCinst
  550 VDU23,203,70,60,24,0,0,0,0,0
  560 VDU23,204,70,60,24,0,0,0,0,0
  570 VDU23,205,24,32,66,129,129,66,4,24
  580 VDU23,206,0,252,48,48,24,15,7,0
  590 VDU23,207,0,0,0,0,0,195,255,60,0
  600 VDU23,208,60,24,24,24,60,231,231,0
  610 VDU23,209,0,63,12,12,24,240,224,0
  620 VDU23,210,127,255,128,127,17,18,20
,24
  630 VDU23,211,255,255,0,255,0,0,0,0
  640 VDU23,212,254,255,1,254,136,72,40,
24
  650 VDU23,213,24,24,24,24,60,60,60,126
  660 VDU23,214,9,136,64,0,0,20,42,221
  670 VDU23,215,17,162,4,0,48,88,180,95
  680 VDU23,216,0,255,127,63,31,31,7,0
  690 VDU23,217,0,255,254,252,248,248,22
4,0
  700 VDU23,218,255,191,159,143,143,78,6
0,24
  710 VDU23,219,126,126,126,60,60,24,24,
24
  720 VDU23,220,1,0,8,28,62,33,1,1
  730 VDU23,221,0,248,156,156,132,0,0,0
  740 VDU23,222,24,24,0,28,34,66,129,129
  750 VDU23,223,255,255,255,191,159,143,
207,126
  760 VDU23,238,120,132,15,119,135,9,16,
8
  770 VDU23,239,30,33,224,222,193,32,16,
32
  780 VDU23,240,0,28,34,33,37,17,14,12
  790 VDU23,241,28,34,33,37,17,14,4,206
  800 VDU23,242,15,57,119,246,246,251,25
5,63
```




```
  810 VDU23,243,231,255,254,112,1,142,24
8,224
  820 VDU23,249,60,102,153,153,165,195,6
6,60
  830 VDU5:PROCknight
  840 MOVE1132,180:VDU231,232,10,8,233,1
0,8,234:MOVE100,800:VDU224,225,10,8,8,22
6,10,8,227:MOVE1132,800:VDU231,232,10,8,
233,10,8,234
  850 PROCweirdo
  860 GCOL0,1:MOVE96,680:VDU224,225,10,8
,8,226,10,8,227:MOVE1132,680:VDU231,232,
10,8,233,10,8,234
  870 PROCduck
  880 GCOL0,2:MOVE96,560:VDU224,225,10,8
,8,226,10,8,227:MOVE1132,560:VDU231,232,
10,8,233,10,8,234
  890 PROCp(330,760,"1.  The Bold Knight
")
  900 PROCp(330,640,"2.  The  Hunch-Man"
)
  910 PROCp(330,520,"3.  The Bionic Bird
")
  920 VDU4:PRINTTAB(5,3);"Choose your ch
aracter to begin"
  930 M%=0:REPEAT
  940 IF INKEY-49 PROCknight:M%=3
  950 IF INKEY-50 PROCweirdo:M%=1
  960 IF INKEY-18 PROCduck:M%=2
  970 UNTIL M%
  980 VDU28,0,22,39,0,12,26
  990 VDU23,196,0,1,3,6,12,56,240,192
 1000 VDU23,197,0,248,12,6,99,195,127,28
 1010 VDU23,198,8,4,12,52,44,56,24,0
 1020 VDU23,199,0,24,56,16,24,24,153,126
 1030 VDU23,244,255,18,12,12,18,33,81,13
8
 1040 VDU23,245,255,72,48,48,72,132,138,
81
 1050 VDU23,246,129,196,66,66,66,66,36,6
0
```




```
 1060 VDU23,247,193,193,227,247,255,255,
255,255
 1070 VDU23,248,131,131,199,129,255,255,
255,255
 1080 B%=-96:D%=512:C%=224
 1090 IF PAGE>&1100 THEN PAGE=&1100
 1100 CHAIN"QUEST1"
 1110 :
 2000 DEF PROCknight
 2010 VDU23,224,62,124,241,206,178,178,2
06,241
 2020 VDU23,225,0,112,200,72,72,72,72,24
0
 2030 VDU23,226,124,60,120,124,102,79,79
,95
 2040 VDU23,227,127,62,60,24,48,96,252,2
54
 2050 VDU23,228,0,3,13,60,112,248,120,48
 2060 VDU23,229,127,62,252,198,3,1,0,0
 2070 VDU23,230,0,24,60,60,120,240,192,0
 2080 VDU23,231,0,14,19,18,18,18,18,5
 2090 VDU23,232,60,62,143,115,77,77,115,
143
 2100 VDU23,233,60,12,14,51,103,103,103,
127
 2110 VDU23,234,127,62,60,24,12,6,63,127
 2120 VDU23,235,0,24,60,60,30,15,3,0
 2130 VDU23,236,254,124,63,99,192,128,0,
0
 2140 VDU23,237,0,192,176,60,14,31,30,12
 2150 ENDPROC
 2160 :
 2170 DEF PROCweirdo
 2180 VDU23,224,0,1,3,3,7,15,28,56,23,22
5,112,248,204,204,244,224,113,30,23,226,
112,112,120,236,206,159,159,254,23,227,2
54,124,56,48,24,48,57,254,23,231,14,31,5
1,51,47,7,142,120,23,232,0,0,128,192,240
,248,28,14
 2190 VDU23,233,14,14,30,55,115,249,249,
127,23,234,127,62,28,12,24,12,156,127
 2200 ENDPROC
 2210 :
 2220 DEF PROCduck
 2230 VDU23,231,1,2,2,15,255,0,255,3,23,
232,244,120,125,254,126,126,252,248,23,2
33,192,96,97,121,247,231,255,255,23,224,
87,124,252,127,92,62,31,15,23,225,0,128,
128,224,255,0,255,224,23,226,3,3,99,247,
254,243,199,126
 2240 ENDPROC
 2250 :
 2260 DEF PROCp(S%,R%,A$)
 2270 MOVE S%,R%:GCOL0,1:PRINTA$
 2280 MOVE S%+4,R%+4:GCOL0,2:PRINTA$
 2290 ENDPROC
 2300 :
 2310 DEF PROCinst
 2320 COLOUR2:VDU4
 2330 PRINTTAB(15,7)"Z ... Left"
 2340 PRINTTAB(15,8)"X ... Right"
 2350 PRINTTAB(10,9)"Return ... Jump"
 2360 PRINTTAB(11,10)"Space ... Pickup/D
rop"
 2370 PRINTTAB(10,4)"By Matthew  Eastmon
d"
 2380 VDU5:PROCp(448,960,"KNIGHT QUEST")
 2390 VDU4:PRINTTAB(9,20)"Press Space to
 continue"
 2400 REPEAT UNTIL INKEY-99
 2410 SOUND1,4,199,1:GCOL0,3
 2420 VDU28,1,20,38,1,12,26
 2430 ENDPROC
```

*****

```
   10 REM Program  QUEST1
   20 REM Version  B1.03
   30 REM Author   Matt Eastmond
   40 REM BEEBUG   April 1988
   50 REM Program  subject to copyright
   60 :
  100 ON ERROR MODE 7:REPORT:PRINT" at l
ine ";ERL:END
  110 DIM ob%(36),ob$(5),c%(5),F%(7)
  120 DIM p%(3):REPEAT
  130 PROCsp("Press  Space",460,660)
  140 REPEAT UNTIL INKEY-99
  150 PROCinit:PROCnew:REPEAT:PROCman
  160 IF X%=-32 OR X%=1248 OR Y%=992 OR
Y%=416 B%=-96:PROCnew
  170 IF X1%=12 PROCcon
  180 IF W%=0 OR W%=2 PROCspider ELSE PR
OCball
  190 IF X%=1088 IF Y%=704 IF X1%=3 PROC
tune:PROCscroll
  200 UNTIL st%=176 OR sc%=100
  210 PROCwipe
  220 IF sc%=100PROCcongrat ELSE PROCdea
th
```

```
  230 UNTIL FALSE
  240 :
 1000 DEF PROCman
 1010 B%=X%:D%=Y%:C%=A%
 1020 IF J%=0 AND A%=224 AND POINT(X%,Y%
-101)=0 Y%=Y%-32:PROCprint:ENDPROC
 1030 IF J%=0 AND A%=231 AND POINT(X%+62
,Y%-101)=0 Y%=Y%-32:PROCprint:ENDPROC
 1040 IF INKEY-74+J%<0 J%=7:SOUND1,1,0,2
 1050 IF J%PROCjump:ENDPROC
 1060 IF INKEY-67 AND POINT(X%+70,Y%-80)
<1X%=X%+32:A%=224:PROCprint:ENDPROC
 1070 IF INKEY-98 AND POINT(X%-8,Y%-80)<
1X%=X%-32:A%=231:PROCprint:ENDPROC
 1080 IF INKEY-99 PROCobject
 1090 ENDPROC
 1100 :
 1110 DEF PROCprint
 1120 GCOL3,M%:MOVE B%,D%:IF C%=224 VDU
&E0&E1&0A&08&08&E2&0A&08 ELSE VDU &E7&E8
&A&8&E9&A&8
 1130 IF F%=0VDU C%+3 ELSE VDU 8,C%+4,C%
+5,C%+6
 1140 F%=F%+1:MOVE X%,Y%:IF F%=2 F%=0
 1150 IF A%=224 VDU A%,A%+1,10,8,8,A%+2,
10,8 ELSE VDU A%,A%+1,10,8,A%+2,10,8
 1160 IF F%=0VDU A%+3 ELSE VDU 8,A%+4,A%
+5,A%+6
 1170 ENDPROC
 1180 :
 1190 DEF PROCjump
 1200 E%=INKEY-98-INKEY-67
 1210 IF E%=1 A%=224
 1220 IF E%=-1A%=231
 1230 IF POINT(X%-8,Y%-80)>0 IF E%=-1 E%
=0
 1240 IF POINT(X%+70,Y%-80)>0 IF E%=1 E%
=0
 1250 X%=X%+32*E%:Y%=Y%+F%(J%)
 1260 J%=J%-1:PROCprint
 1270 IF POINT(X%+28,Y%-101) J%=0
 1280 ENDPROC
 1290 :
 1300 DEF PROCobject
 1310 IF Y%<>(by+96) OR X%-bx>96 OR X%-b
x<-87 ENDPROC
 1320 SOUND1,4,99,1:FORT=0TO60:NEXT
 1330 PROCwipe
 1340 FOR H=0TO5
 1350 IFc%(H)=X1% T=H*6:PROCpob(T):VDU4,
31,11,27,17,ob%(T),ob%(T+1),ob%(T+2),17,
ob%(T+3),10,8,8,ob%(T+4),ob%(T+5),17,1:P
RINT"  The ";ob$(H):c%(H)=-1
 1360 NEXT:VDU5
 1370 FOR I=0TO5
 1380 IF c%(I)=-1 L=I*6:IF L<>T PROCpob(
L):c%(I)=X1%
 1390 NEXT
 1400 IF c%(0)=4 PROCtune:PRINTTAB(4,8)"
Thank":PRINTTAB(5,9)"you":VDU5:PROCpob(0
):c%(0)=0:p%(0)=1
 1410 IF c%(4)=3 AND p%(2)=0 PROCtune:VD
U5:Y%=896:X%=0:PROCprint:SOUND1,1,0,2:p%
(2)=1
 1420 IF c%(5)=5 AND p%(3)=0 PROCtune:VD
U31,19,21,32,32,32,32:VDU5:p%(3)=1
 1430 IF c%(1)=4 AND p%(1)=0 PROCtune:PR
INTTAB(4,8)" Hey":PRINTTAB(3,9)" Presto"
:VDU31,0,10,32,32,10,8,8,32,32,10,8,8,32
,32,5:p%(1)=1
 1440 ENDPROC
 1450 :
 1460 DEF PROCspider
 1470 MOVEZ%,U%:IF W%=0 GCOL3,1:VDU&EE&E
F ELSE GCOL3,3:VDU&DE&A&8:GCOL3,2:VDU&DF
 1480 Z%=Z%+L%
 1490 IF Z%<S%OR Z%>T%L%=-L%
 1500 MOVEZ%,U%:IF W%=0 GCOL3,1:VDU&EE&E
F ELSE GCOL3,3:VDU&DE&A&8:GCOL3,2:VDU&DF
 1510 IF X%=Z% OR X%=Z%+32 IF Y%-U%<96 I
F Y%-U%>-8 PROCenergy
 1520 ENDPROC
 1530 :
 1540 DEFPROCball
 1550 GCOL3,2:MOVEZ%,U%:VDU&F9:U%=U%+L%
 1560 IF U%<S%OR U%>T%L%=-L%:SOUND0,-9,6
,1
 1570 MOVEZ%,U%:VDU&F9
 1580 IF X%=Z% OR X%=Z%-32 IF Y%-U%<96 I
F Y%-U%>-8 PROCenergy
 1590 ENDPROC
 1600 :
 1610 DEF PROCcon
 1620 VDU4,17,128,17,3,31,Q%,15,32,31,Q%
+5,15,32
 1630 Q%=Q%+R%
 1640 IF Q%=7 OR Q%=24 R%=-R%:SOUND0,-15
,6,1
 1650 VDU17,129,31,Q%,15,192,193,194,195
,192,193,17,128,5
 1660 ENDPROC
 1670 :
 1680 DEF PROCnew
 1690 IF X%=-32 X1%=X1%-1:X%=1184
 1700 IF X%=1248 X1%=X1%+1:X%=32
 1710 IF Y%=992 X1%=X1%-4:Y%=448
 1720 IF Y%=416 X1%=X1%+4:Y%=960
 1730 PROCscreen:PROCprint
 1740 FOR LA%=0TO5
 1750 IF c%(LA%)=X1% L=LA%*6:PROCpob(L)
 1760 NEXT
 1770 IF W%=0GCOL3,1:MOVEZ%,U%:VDU238,23
9
 1780 IF W%=2MOVEZ%,U%:GCOL3,3:VDU222,10
,8:GCOL3,2:VDU223
 1790 IF W%=1 GCOL3,2:MOVEZ%,U%:VDU249
```

```
 1800 ENDPROC
 1810 :
 1820 DEFPROCinit
 1830 RESTORE 2000:st%=1072
 1840 VDU19,1,0;0;19,2,0;0;19,3,0;0;
 1850 GCOL0,2
 1860 FORT%=192TOst%STEP16
 1870 MOVE T%,210:DRAW T%,190
 1880 NEXT
 1890 FORT=0TO3:p%(T)=0:NEXT
 1900 FORT=0TO5:READ ob$(T):NEXT
 1910 FORT=0TO7:READ F%(T):NEXT
 1920 FORT=0TO35:READ ob%(T):NEXT
 1930 FORT=0TO5:READ c%(T):NEXT
 1940 X1%=6:V%=192:sc%=0:B%=-96:J%=0
 1950 X%=480:Y%=512:A%=224
 1960 Q%=16:R%=1:F%=1:L%=32:VDU4
 1970 COLOUR1:PRINTTAB(18,30)" 00 % "
 1980 ENDPROC
 1990 :
 2000 DATA Bowl of food,Water Bottle,Spa
re Head,Nasty Plant,Trampoline,Horrible
Hole
 2010 DATA -32,-32,-32,0,0,32,32,32,3,21
4,215,1,216,217,3,222,32,2,223,32,M%,224
,225,M%,226,32,2,220,221,1,216,217,3,32,
32,2,244,245,3,32,32,3,32,246,6,2,8,10,9
,12
 2020 :
 2030 DEF PROCscreen
 2040 RESTORE3110
 2050 IF X1%>1 FOR LA%=1 TO X1%-1:READ Z
%,Z%,Z%,A$,Z%,Z%,A$,Z%,Z%,Z%,Z%,Z%:NEXT
 2060 VDU4,28,0,22,39,0,12,26
 2070 COLOUR129:COLOUR3:LA%=1
 2080 READ A,B,C,A$
 2090 ?&FE00=1:?&FE01=0:m%=0
 2100 FOR W%=1TOA
 2110 PROCsw:Z%=192
 2120 FOR U%=I%TO P%
 2130 VDU31,U%,K%,Z%
 2140 Z%=Z%+1:IF Z%=196 Z%=192:IF m%<81
?&FE00=1:?&FE01=m%:m%=m%+1
 2150 NEXT,
 2160 COLOUR128:FOR W%=1TO B
 2170 IF m%<81 ?&FE00=1:?&FE01=m%:m%=m%+
1
 2180 PROCsw:VDU31,K%,P%
 2190 IF I%=0VDU&11&2&F0&F1&A&8&8&F2&F3
 2200 IFI%=1VDU&11&1&CD&A&8&CD&A&8&CD&A&
8&8&8&11&2&C6&9&11&1&CD&9&11&2&C6&A&8&8&
8&8&8&11&3&C7&9&11&1&CD&9&11&3&C7&A&8&8&
8&8&8&11&2&CE&CF&D0&CF&D1
 2210 IF I%=2 VDU&D2&A&8&D5:FORS=9TO31:V
DU31,S,19,211:NEXT:VDU&D4&A&8&D5
 2220 IF I%=3 VDU&11&2&C6&11&1&A&8&DB
 2230 IF I%=4 VDU&11&2&C6&11&3&A&8&C7
 2240 IF I%=5 VDU&11&3&D3&D3&D3&D3
 2250 IF I%=6 GCOL0,2:VDU17,1,238,239:MO
VE (K%+1)*32,(32-P%)*32:PLOT21,(K%+1)*32
,900
 2260 IF I%=7 VDU&11&2&D5&8&B&D2&D3&D3&D
3&D3&D3&D4&A&8&D5
 2270 IF I%=8 VDU&11&1&C4&C5
 2280 IF I%=9 VDU&11&3&DA&A&8&C7
 2290 IF I%=10 VDU&11&1&C8&C9&A&8&8&11&3
&CA&11&1&CB
 2300 IF I%=11 VDU17,3,222,10,8,17,RND(2
),223
 2310 IF I%=12 VDU&11&2&DC&DD&A&8&8&11&1
&D8&D9
 2320 IF I%=13 VDU&11&2&DC&DD&A&8&8&DC&D
D&A&8&8&DC&DD
 2330 NEXT
 2340 IF C=0GOTO2420
 2350 COLOUR3:COLOUR129:FOR W%=1TO C
 2360 PROCsw:Z%=192
 2370 FOR U%=K%TO P%
 2380 IF m%<81 ?&FE00=1:?&FE01=m%:m%=m%+
1
 2390 VDU31,I%,U%,Z%,Z%+RND(2)
 2400 Z%=Z%+1:IF Z%=194 Z%=192
 2410 NEXT,:COLOUR128
 2420 READ bx,by:GCOL0,129:VDU24,bx;by-2
8;bx+96;by;16,26
 2430 IF m%<81 FOR X=m% TO80:?&FE00=1:?&
FE01=X:NEXT
 2440 READ ROOM$,W%,Z%,U%,S%,T%:PROCsp("
+ +  The "+ROOM$+" + +",220,980)
 2450  VDU19,1,1;0;19,2,3;0;19,3,7;0;4
 2460 IF X1%=3 COLOUR2:PRINTTAB(30,7)"Th
e Crown"
 2470 IF X1%=4 AND p%(0)=0 COLOUR2:PRINT
TAB(4,8)"Feed":PRINTTAB(5,9)"me"
 2480 IF X1%=4 AND p%(0)=1 COLOUR2:PRINT
TAB(4,8)"I am":PRINTTAB(3,9)"thirsty"
 2490 IF X1%=3 VDU31,34,9,247,248
 2500 IF X1%=4 AND p%(1)=1 VDU31,0,10,32
,32,10,8,8,32,32,10,8,8,32,32
 2510 VDU5
 2520 ENDPROC
 2530 :
 2540 DEF PROCpob(LA%)
 2550 MOVE bx+18,by+64:GCOL3,ob%(LA%):VD
U ob%(LA%+1),ob%(LA%+2):GCOL3,ob%(LA%+3)
:VDU10,8,8,ob%(LA%+4),ob%(LA%+5)
 2560 ENDPROC
 2570 :
 2580 DEFPROCwipe
 2590 VDU4,31,11,27,32,32,10,8,8:PRINTSP
C(21):VDU5
 2600 ENDPROC
 2610 :
 2620 DEFPROCdeath
```

```
 2630 SOUND0,4,6,2:PROCscroll
 2640 PROCsp("Game  Over",496,860)
 2650 PROCsp(STR$(sc%)+"% Completed",464
,796)
 2660 PROCdelay(999)
 2670 ENDPROC
 2680 :
 2690 DEF PROCtune
 2700 VDU4:RESTORE3100:FORS=1TO7
 2710 READ P,L:SOUND2,4,P,L:NEXT
 2720 COLOUR1:sc%=sc%+20
 2730 PRINTTAB(18,30)" ";sc%;" % "
 2740 ENDPROC
 2750 :
 2760 DEFPROCcongrat
 2770 PROCsp("WELL DONE !",480,800)
 2780 PROCsp("You have completed Knight-
Quest",150,550)
 2790 B%=-96:X%=900:Y%=700:A%=231
 2800 PROCprint
 2810 ENDPROC
 2820 :
 2830 DEF PROCenergy
 2840 GCOL0,0:FORT=st%-64 TO st%STEP16
 2850 MOVE T,210:DRAW T,180:NEXT
 2860 st%=st%-64:SOUND1,3,RND(99),1
 2870 ENDPROC
 2880 :
 2890 DEF PROCscroll
 2900 FORT=0TO21:VDU4,31,0,31,10
 2910 PROCdelay(30):NEXT
 2920 FORT=0TO21:VDU31,0,0,11
 2930 PROCdelay(30):NEXT
 2940 ENDPROC
 2950 :
 2960 DEF PROCdelay(f%):FORX=0TOf%:NEXT
 2970 ENDPROC
 2980 :
 2990 DEFPROCsw
 3000 I%=ASC(MID$(A$,LA%,1))-65
 3010 K%=ASC(MID$(A$,LA%+1,1))-65
 3020 P%=ASC(MID$(A$,LA%+2,1))-65
 3030 LA%=LA%+3
 3040 ENDPROC
 3050 :
 3060 DEF PROCsp(A$,x,y)
 3070 VDU5:GCOL0,1:MOVE x,y:PRINT A$:GCO
L0,2:MOVE x+4,y+4:PRINT A$
 3080 ENDPROC
 3090 :
 3100 DATA177,4,169,2,161,2,169,3,161,9,
177,4,181,1
 3110 DATA9,2,3,KJLSJT[I£[K£[N£ADhAVhGQb
cRhBdEKQOAEUGIP]IP,600,352,Climbing Fram
e,0,640,416,96,1152
 3120 DATA3,8,2,AVhADhARgH_UAdOAdLAdIBKE
B\EMSPNSMgIUAEN,96,352,SKull Passage,1,9
6,576,544,608
 3130 DATA4,4,2,ADhAVY_VhZNhBTEBJEHLUGOQ
ZEMAIU,96,352,Gym,1,960,448,416,544
 3140 DATA8,7,2,AV£eVhADheSfePfANIRNV]N£
BOEBZEHHUHWUJZRJJRAFLgEUAEM,288,608,Old
Kitchens,1,960,448,416,544
 3150 DATA2,5,1,AVhADhBUEHDUEGRICOEDMADV
,600,352,Holy Room,0,896,416,736,1152
 3160 DATA2,6,0,AVhADhBUECITDHFDdHENRE\R
,600,416,Dining Room,0,896,480,736,1152
 3170 DATA10,4,0,AVW]VhKSNORSPQTSPWUO_eP
hADY_DhGXMGFJGNOGeL,1024,352,Spidery Sta
irs,2,704,416,320,704
 3180 DATA8,4,1,AVaADaAP]^OfaNfeKfeHfeEf
BUEDFRDQRD\RgDV,288,640,Red Bedroom,1,96
0,448,416,512
 3190 DATA8,5,3,AOKcLd\N^ZP\\S^JVhADRWDh
DFIDJIMMTLQTKYTAENgJUJPU,608,416,Rubbish
 Tip,0,640,544,416,704
 3200 DATA4,5,2,_QhADhHMZAVhMTKNTHNJJKQK
KXKAJUFMR,992,512,Jolly Jungle,1,800,448
,416,576
 3210 DATA7,7,2,eQgQOQQNQQPYAVhADW]DhBJE
B£EHEUH_ULSNLUNMHRgPUWEJ,704,544,Barmy B
ar,0,448,448,416,832
 3220 DATA8,4,3,AVhAPGeSgeMgeJgeGgADa_Pf
BJEKJTKPTK_TgDUAQU£EL,600,640,Mouldy Cel
lar,1,128,640,608,864
```

THE COMMS SPOT - *Continued from page 53*

This is now set to change, and we are planning to cater better for all members whether they own Beebs, Masters or Archimedes. One of the sections we want to improve is the feedback area, publishing reader's letters on a regular basis. Also the hints and tips area is in need of updating. For both of these we need a response from YOU! So please get logged on and get writing to us with your ideas, comments, and hints and tips

Having outlined where the improvements need making, let me say that at the moment there is still plenty in our area of interest. In the coming months things will get even better so do keep keying *BEEBUG# after you have logged on

If you want to send any mailboxes to us then the number to use is 819991213 (under the name of BEEBUG's editor). We look forward to hearing from you.

SOFTWARE WRITE PROTECT

Neofitos Pashalis

Sideways RAM boards often have a write protect option. This normally consists of a manual switch across two wires. A rather neat alteration can be made to allow the switch to be driven by software. Simply remove the hardware switch and connect the two wires to the outside pins of the cassette socket. Write protect can then be selected and de-selected by using the *MOTOR command to open and close the relay. Whether the relay needs to be open or closed to write protect the RAM depends upon the sideways RAM in use. Write protect switches with more than two wires cannot be rewired in this way.

VERY HARD RESET

Lester Cook

It is often necessary to be able to reset a BBC machine from within a Basic program. It is easy to simulate a press of the Break key using the command CALL !&FFFC (which calls the reset routine in the operating system), but it is more complicated if you need to achieve a power-on reset. This can be done by executing the extra command *FX151,78,127 before the command CALL !&FFFC. This will totally simulate a power-up, including the complete erasure of memory. Possible uses of this technique include initalisation of protected ROM images in sideways RAM , or even to prevent the recovery of a program once it has finished running. If the command *FX151,78,127 is typed in directly, and the Break key is then pressed, the end result is exactly as if the computer had been turned off and then on again, except that this method causes no wear on the power supply. Incidentally, the keyboard will stop working as soon as the *FX command has been issued so the commands will need to be placed in a program if they are to totally reset the machine, i.e.:

```
10 *FX 151,78,127
20 CALL !&FFFC
```

GREEN SCREENS

Jason Wood

Many colour monitors available on the market nowadays, including those supplied by Acorn, have a Green Screen switch to change the display to green. It is very important to remember that all this switch does is to turn off the blue and red guns in the monitor, leaving only the green component of the picture. While this works fine for a white display, and any other colours containing green, it does mean that colours like blue and red totally vanish.

CHANGING TAB

Jane Taylor

Many programs define function key 10 to perform a specific action when the Break key is pressed. However, there can be problems testing whether you have used the right sequence because the only way of testing is to actually press the Break key itself. An alternative method is to issue a *FX219,138 which will make the Tab key produce the same string as function key 10. Normal use of Tab is restored by entering *FX219,9.

PRINTER TIPS

Peter Mileham

A quick way of sending a form feed to the printer from the keyboard is to hold down the Control key and press the following sequence of letters: B A L C. The advantage of this sequence over the normal Ctrl-L method is that the screen display remains totally unaffected. Another version of this allows a printer to be tested to see if it is on line or not. Hold down the Control key and press this sequence: B A G C. If the printer is on line it will emit a beep. Naturally, this will only work on printers that have a buzzer.

B

POSTBAG

POSTBAG

UPDATED EPROM PROGRAMMER

I have been an avid reader of BEEBUG for four years and constructed the EPROM programmer as described in the Aug/Sept 1985 issue (Vol.4 No.4). After a few teething troubles I managed to get the package working very well and have used it without any further trouble. Imagine my disappointment when I found that most EPROM manufacturers were changing the programming voltage from 21v to 12.5v.

I decided to find out how to alter the circuit so that I could use both 21v and 12.5v EPROMs. The results are very easy to include in the package. The solution lies in soldering a 27K resistor and an on/off switch in parallel with the 20K resistor which connects to pin 1 of the TL497 voltage regulator shown in the original circuit diagram. The effect is to give a change from 21v to 12.5v at the flick of a switch.

E.T.Ryan

The author of the original design, Geoff Bains, confirms that this modification should work as described.

SIMPLY LOAD AND GO

There is a very simple alternative to your Load and Go with Basic utility (BEEBUG Vol.6 No.7). A simple EXEC file in the library directory on the following lines will achieve basically the same result:

```
*DIR $.XXXXX
CHAIN "FRED"
```

Page changes can be incorporated in the EXEC file if required. This means a program can be run regardless of the current directory. I use an EXEC file called "I" to run a modified version of your address book program to maintain a list of all my programs. Having saved a program, a quick *I runs the index program allowing me to record the basic details of my new program.

S.B.Birks

*Mr Birks is quite correct in all that he says, and the system he uses (on an ADFS system by the sound of it) clearly works well. However, using an EXEC file in this way will require a separate EXEC file to chain each Basic program, whereas the BSAVE utility described in the magazine allows a Basic program alone to be saved in such a way that it may subsequently be run by typing *name (or whatever name the program was saved under). Also, since only one file is involved, this approach is likely to be quicker, though not by very much.*

QUART IN A PINT POT

Having recently completed my masterpiece of a 17K Basic program, my model B (with 6502 second processor) was getting dangerously close to capacity when filled with data, and with seemingly little chance of being able to squeeze in more. I was about to give up when I discovered the 'The Advanced Disk User Guide for the BBC Micro' and in chapter 16 page 311 the answer to my problem: break the program into modules and have the Menu, save, Load, Sort, Display, Amend etc saved onto disc as overlays. Using this approach I gained an extra 6K of space which I can fill with data. It appears to me that here is scope for an article or two on a subject that has been sadly neglected by all magazines.

I.S.Crawford

The idea of dividing a program up into several modules any one of which may be loaded into a reserved area of memory is a principle that has been known for a long time (many early mainframe computers had much less memory than a BBC micro). BEEBUG has covered this in two articles: Dynamic Overlaying of Procedures in Vol.3 No.2, and Virtual Arrays in Vol.5 Nos. 8 & 9. These both take a different approach to that referred to by Mr.Crawford. We have also included an article on this subject in this issue, using both disc and sideways RAM, and no doubt we shall return to this topic again. B

BEEBUG MEMBERSHIP

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

BEEBUG SUBSCRIPTION RATES

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

BACK ISSUE PRICES

Volume	Magazine	Cassette	5"Disc	3.5"Disc
1	£0.40	£1.00	-	-
2	£0.50	£1.00	-	-
3	£0.70	£1.50	£3.50	-
4	£0.90	£2.00	£4.00	£4.50
5	£1.20	£2.50	£4.50	£4.75
6	£1.30	£3.00	£4.75	

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 subscriptions)

BEEBUG MAGAZINE is produced by BEEBUG Ltd.

Editor: Mike Williams
Assistant Editor: Kristina Lucas
Technical Editor: David Spencer
Technical Assistant: Lance Allison
Production Assistant: Yolanda Turuelo
Membership secretary: Mandy Mileham
Editorial Consultant : Lee Calcraft
Managing Editor: Sheridan Williams

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) 1988

Printed by Head Office Design (0782) 717161 ISSN - 0263 - 7561

Magazine Disc/Cassette

APRIL 1988 DISC/CASSETTE CONTENTS

AN ADFS MENU FOR VIEW - fast and efficient file access for View users.

RESISTOR CALCULATION - provides a combination of preferred values to suit any requirement.

NOW C HERE (Part 2) - a limerick program demonstrating the use of character strings.

OVERLAYING TECHNIQUES - two separate utilities to expand usable memory using disc or sideways RAM.

FIRST COURSE
CHARACTER CONTROL (Part 2) - two programs demonstrating string and decimal justification.

THE MASTER SERIES
ENHANCED PRINTER BUFFER - complete enhanced printer buffer program including many additional star commands.
DATE-STAMPING FILES - a simple way to date stamp all your files on a Master.

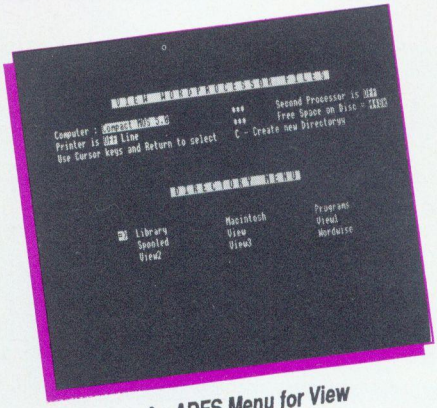
VIDEO CASSETTE CATALOGUER - the whole program including this month's search, sort and hard copy options.

EXPLORING ASSEMBLER (PART 9) - three demo programs showing the use of the shift and rotate instructions.

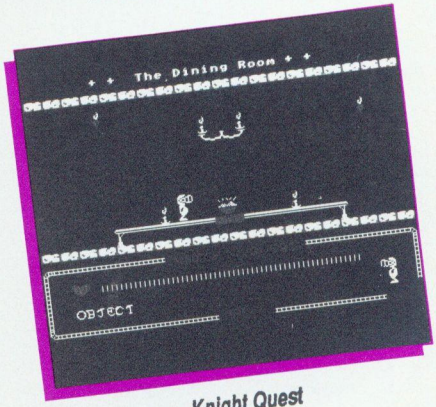
KNIGHT QUEST - a superbly implemented and highly challenging arcade style adventure game. Undoubtedly one of the best games we have published.

IBM TO BBC TRANSFER UTILITY - a utility to transfer files from MS-DOS format discs to BBC format discs, all running on a BBC micro.

MAGSCAN - Bibliography for this issue of BEEBUG, the last issue in volume 6.



An ADFS Menu for View



Knight Quest

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

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

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

Cassette subscriptions can be commuted to a 5.25" or 3.5" disc subscription on receipt of £1.70 per issue of the subscription left to run. All subscriptions and individual orders to:

BEEBUG, Dolphin Place, Holywell Hill, St.Albans, Herts. AL1 1EX.

The Best Deals on Archimedes From BEEBUG

The Archimedes Specialists

1 0% FINANCE

For a limited period we are able to offer 0% APR finance over 9 months on the purchase of any Archimedes. You pay no interest at all. This is a brand new scheme only available from BEEBUG. The deposit and repayments are shown below.

	Deposit	9 Payments		Deposit	9 Payments
A305 Base	£83.85	£80.00	A310 Base	£90.25	£89.00
A305 Mono	£87.35	£86.00	A310 Mono	£102.75	£94.00
A305 Colour	£106.85	£103.00	A310 Colour	£113.25	£112.00
A310M Base	£96.25	£96.00	A440 Base	£267.85	£264.00
A310M Mono	£108.75	£101.00	A440 Mono	£271.35	£270.00
A310M Colour	£119.25	£119.00	A440 Colour	£290.85	£287.00

2 TRADE IN YOUR OLD BBC, MASTER OR COMPACT FOR AN ARCHIMEDES

We will be pleased to accept your old computer (in working condition) as part exchange towards the purchase of an Archimedes. (If you use the finance scheme this will replace your initial deposit on a 305/310, so you pay nothing now). Allowances are as follows:

BBC Issue 4 No DFS	£125
BBC Issue 4 DFS	£175
BBC Issue 7 No DFS	£175
BBC Issue 7 DFS (Or B+)	£225
Master 128	£250
Compact Base System	£215

Please phone for allowances on other Compact and Master systems.

3 FREE DISCS & PC EMULATOR

Join RISC USER, the Archimedes magazine and support group, and purchase your Archimedes by Cheque, Access, Visa, Official Order or 11.5% finance and we will supply you, absolutely free, 10 3.5" discs, a lockable disc storage box, printer lead and the latest version of The PC Emulator from Acorn. Altogether you save more than £142.00.

Prices Including VAT

A305 Base	£803.85	Mono £861.35	Colour £1033.85
A310 Base	£891.25	Mono £948.75	Colour £1121.25
A440 Base	£2643.85	Mono £2701.35	Colour £2873.85

4 11.5% FINANCE OVER 12 TO 36 MONTHS

As a Licensed Credit Broker we are able to offer finance on the purchase of any equipment, including the Archimedes. You still benefit from the free PC Emulator, discs, disc box and printer lead. (Typical APR 23% on the purchase of a 310 Colour system over 36 months.
Deposit £126.25 36 payments of £37.36).

5 DISCOUNTS FOR EDUCATION

We are able to offer attractive discounts to Education Authorities, Schools, Colleges and Health Authorities. Please write with your requirements for a quotation.

TO FIND OUT MORE PHONE OR WRITE NOW. TEL: 0727 40303

We offer a complete service, including Advice, Technical Support, Showroom, Mail Order and Repairs. Our showroom in St. Albans stocks everything available for the Archimedes. Call in for a demonstration.

Please indicate your requirements below.
Subscription to Risc User (£14.50 UK) ☐ Information Pack and Catalogue ☐ 0% Finance Form for 305/310/310M/440 Base/Mono/Colour ☐ 12-36 Months Finance Form for 305/310/310M/440 Base/Mono/Colour ☐ Trade In BBC/Master/Compact ☐ Purchase 305/310/440 Base/Mono/Colour ☐
UK Courier Delivery £7.00. Overseas please ask for a quotation.

I enclose a cheque value £.....

Please debit my Access/Visa/Connect Card No

Expiry...../..... with £.....

Name .....

Address .....

.....

Signature .....

Beebug, Dolphin Place, Holywell Hill, St. Albans, Herts AL1 1EX Tel: 0727 40303

BEEBUG – The Archimedes Specialists