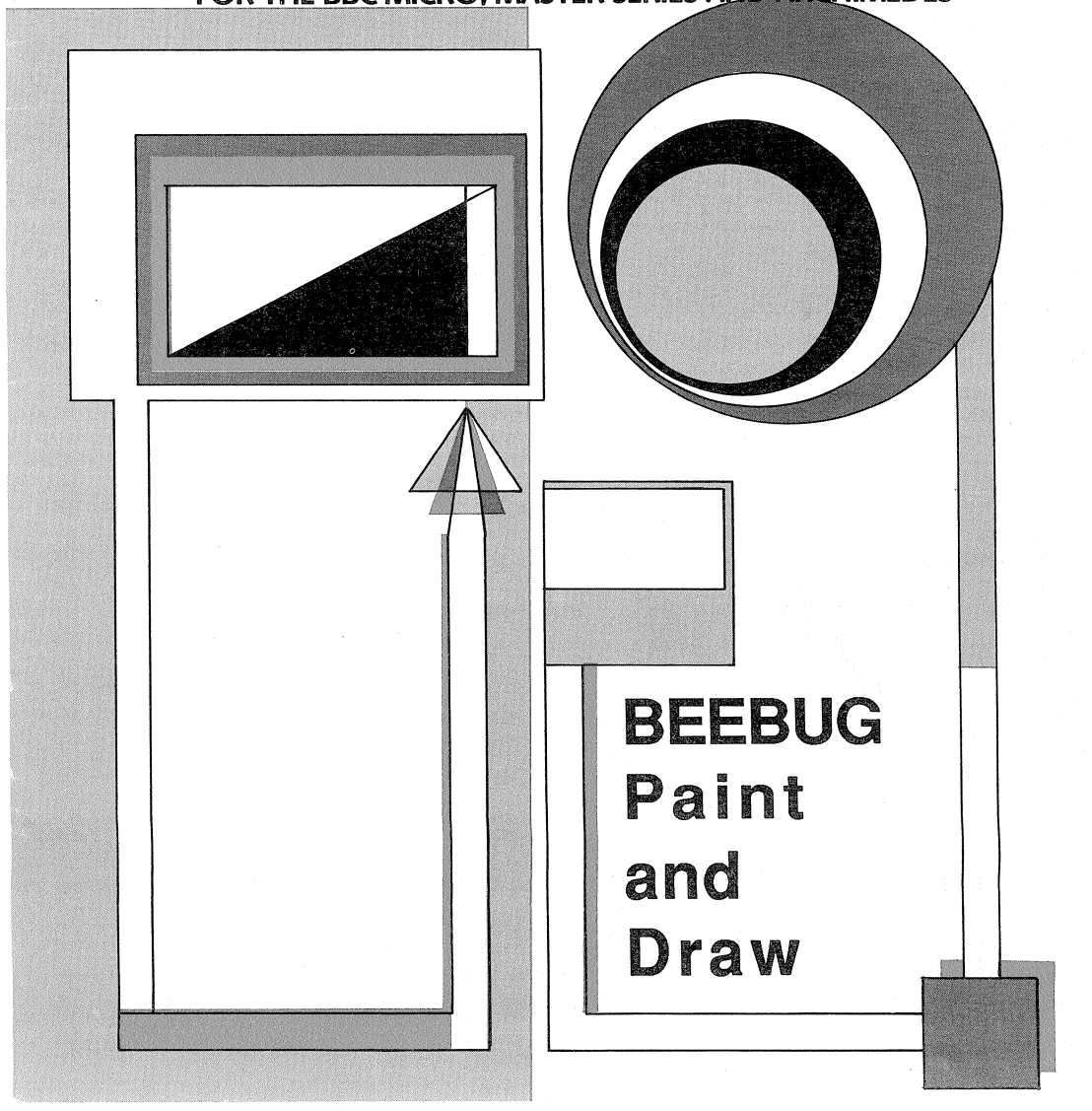
VOLUME 6 NUMBER 4

BEEBUG

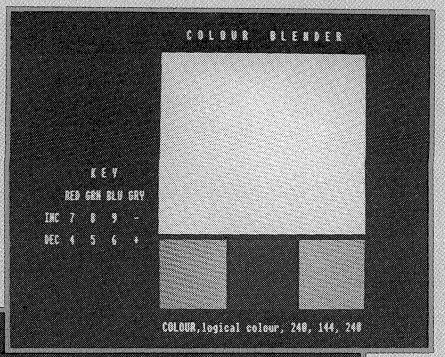
AUG/SEPT 1987

Keystrip Generator

FOR THE BBC MICRO, MASTER SERIES AND ARCHIMEDES



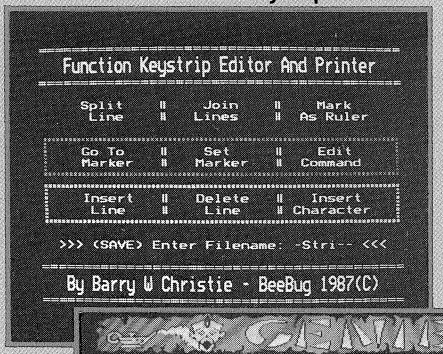
BEEBUG Paint and Draw

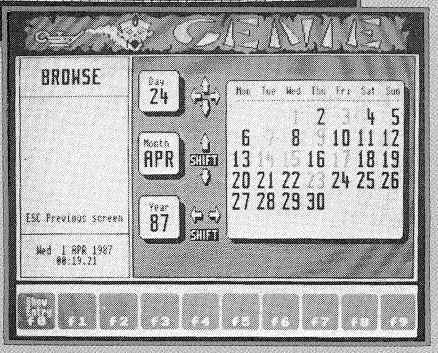


Colouring Archie

Painting & Drawing

Keystrip Generator





Genie

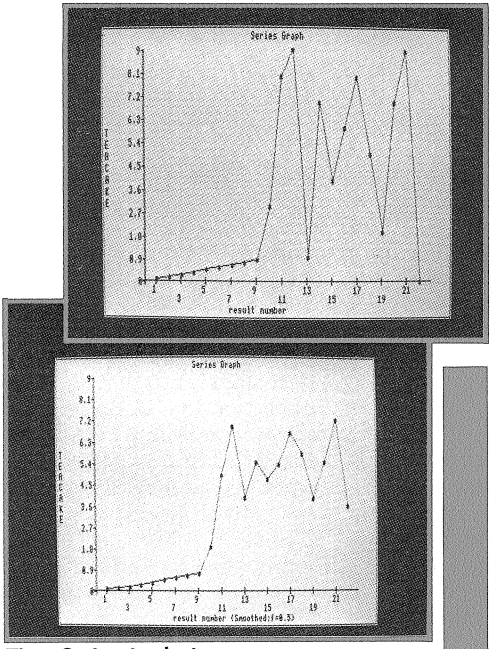
BEEBUG

Volume 6 Number 4
August/September 1987

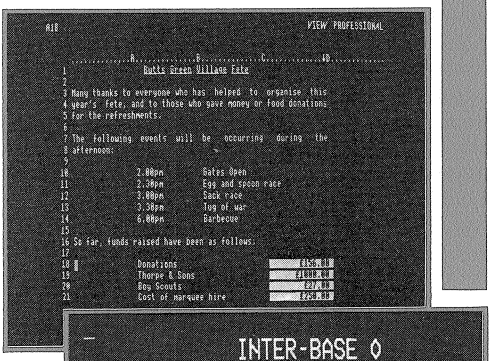
FEATURES

REVIEWS

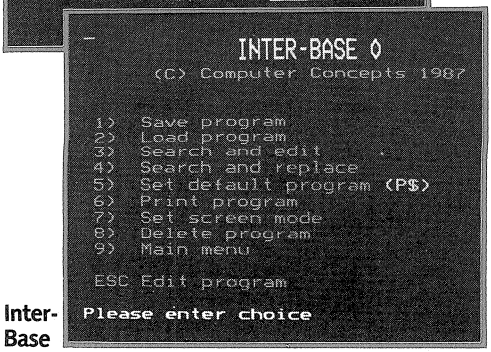
REGULAR ITEMS



Time Series Analysis



View Professional



Inter-Base

EDITORIAL JOTTINGS

BEEBUG SUBSCRIPTIONS

The current subscription rate(for UK members) has been in force since March 1986. But costs do not stand still. Postal rates have risen since then, the cost of printing the magazine has increased, together with service charges and other costs. After careful examination, we have regretfully decided that we must increase the annual subscription rate to BEEBUG with effect from October for NEW subscriptions. We are pleased to be able to maintain the present rates for existing members for a further period of two month (until 1st December 1987). Moreover, even if your renewal date falls later, you may help offset the increase by renewing at the old rate of £12.90, provided you do so before December. Your membership of BEEBUG will simply be extended by a further 12 months. The new rates are as follows:

£7.50 — 6 months (5 issues) UK only
£14.50 — 1 year (10 issues) UK, B.F.P.O., Channel Islands
£20.00 — Rest of Europe
£24.50 — Middle East
£27.00 — Americas and Africa
£29.00 — Elsewhere

Most societies and associations review their subscriptions on an annual basis, and even in these times of lower inflation, subscriptions do continue to rise from year to year. In our case we have been able to maintain the same rate for 18 months. You may feel that our increased coverage of Acorn's Archimedes range has caused this rise, but we can assure you that although this does increase our costs by a small amount, an increase in subscription rates would still be necessary, and the decision to implement this had already been taken.

ARCHIMEDES COMPETITION

Last month we launched our competition to win a brand new Archimedes A305. This is a tremendously exciting and valuable prize, and the competition is open to all BEEBUG members (but excluding staff). You will need to hurry though if you are to meet the August 25th closing date. Full details were given in the July issue of BEEBUG. We anticipate that the results of the competition will be published in the next (October) issue of BEEBUG.

NEW BEEBUGSOFT RELEASE

As visitors to our stand at the Acorn User Show will have seen, BEEBUG is soon to release a 'C' language compiler which will run on all BBC micros. This is a full implementation of 'C' as defined by Kernighan and Ritchie, and in many respects outperforms Acorn's own 'C' language implementation (faster compilation, more compact object code). And it works on a standard model B. Although 'C' is not a language for everyone, it is highly regarded in the computing world, and yet, as far as we know, no version has previously been available for the BBC micro.

News News News News News News New

STRYKER STRYKES AGAIN

Not content with releasing best selling Stryker's Run on the unsuspecting games playing public, Superior Software has now released a sequel entitled 'Codename: Droid, Stryker's Run Part 2'. Departing from the realms of reality, this has dauntless Commander John Stryker in a space adventure as he undertakes a perilous mission to planet Volga. Stryker's Run Part 2 costs £9.95 on cassette (for the BBC micro and Electron), £11.95 for 5.25" disc and £14.95 on 3.5" disc. Earthbound Superior are on (0532) 459453.

CRICKET, LOVELY CRICKET

With this summer's dismal weather, Graham Gooch's Test Cricket from Audiogenic may be your best chance to watch some high class cricket, and maybe take part in the action yourself. Full test match or limited overs, lifelike graphic animation, authentic scoring and sound effects are all available. Graham Gooch's Test Cricket costs £9.95 on cassette and £11.95 on disc. Contact Audiogenic on (0734) 303663.

MOUSE MK III

AMS of mouse fame have now introduced a third variant on their tame rodent, to be included with Stop Press (known previously as Pagemaker until Aldus objected) and with Art Software. Prices for both are £79.99. The Swiss made mouse, claimed as an exclusive by AMS, bears a remarkable similarity to that being sold by Acorn as part of its new Archimedes systems. AMS can be contacted on 061-941 6344.

DIFFERENCES AT WATFORD

Prolific as ever, Watford Electronics has announced a new Mark II version of its DDFS. The new board is claimed to be fully compatible with files created by its predecessor, and the inclusion of official Acorn Tube host code allows a second processor or co-pro adaptor to be used without the need for an Acorn DNFS ROM. Automatic 40/80 track sensing also removes the need for switchable 40/80 track drives. DDFS Mark II costs £39 (plus VAT), and an upgrade deal reduces that to £30 in return for ANY manufacturer's DDFS system.

Another device now available from Watford is the Splitter. This enables two devices to be connected simultaneously to the Beeb's User Port, with switching via a toggle switch. The Splitter costs £15 (plus VAT). Watford is on (0923) 37774.

RED BOXES

The Red Box micro-based control and security system now boasts Red Three to work alongside Red Two, Red One and Red Leader. The new box is a general purpose analogue and digital input/output device for home control systems. Analogue signals from temperature sensors, light meters, humidity detectors and the like are converted into digital signals for transmission to the Red Leader control unit. Feedback from Red Leader can be used to control any appropriate device. For further information on Red Three and the other red boxes contact General Information Systems at Croxton Park, Croxton, Cambs PE19 4SY.

ACORN'S UPS AND DOWNS

Little more than a week after announcing price reductions of £50 on the Master 128 and £100 on the Master Compact, Acorn has announced a price rise of £50 on the Compact. So both machines end up being £50 pounds cheaper (at least until next week!).

Acorn has also released two new Acornsoft packages, a 'C' language system at £89.70 and Termulator at £39.95, both including VAT. Termulator allows any Beeb to act as a terminal to a host mainframe or mini. 'C' looks set to be the latest in high level languages for BBC micros with an Archimedes version due for release later this year. See Editorial Jottings regarding our own 'C' compiler.

INTER-BASE IN CONTROL

Inter-Base, the final link in Computer Concepts' suite of integrated productivity tools, has been released. Peter Rochford investigates this new software, which provides a complete programming language for database design.

Product	**Inter-Base**
Supplier	**Computer Concepts, Gaddesden Place, Hemel Hempstead, Herts HP2 6EX. Tel: (0442) 63933**
Price	**£69.00 inc. VAT and p&p**

Now finally, Inter-Base, Computer Concept's own database management system has arrived, and this completes the Inter-Link series for the BBC Micro and Master. However, there will, I understand, be versions of all the 'Inter' series for the new Archimedes computers, with certain enhancements.

Inter-Base is supplied on a 64K ROM mounted on a small carrier board. This plugs into one of the paged ROM sockets on your Beeb or Master. Master owners should note that the ROM and board are too tall to be used in one of the standard Acorn cartridges, but will work in a Care Electronics cartridge.

The ROM comes with fitting instructions, reference manual, function key strip and a utility disc. The manual comprises some 112 pages and in the main is written to a very high standard. I do have some criticisms of it, which I will refer to later.

The utility disc is supplied on a DFS format single-sided 80 track disc. This contains an example database file of names and addresses, a text file describing the use of the disc, an example file for mail merging the address file with an Inter-Word text file and two programs that will convert both ViewStore and BEEBUG's own Masterfile datafiles to enable them to be read into Inter-Base.

Inter-Base will work with both DFS and the ADFS but cannot work with second processors. The software does make use of any shadow RAM fitted, which is a decided advantage when using 80 column screen modes.

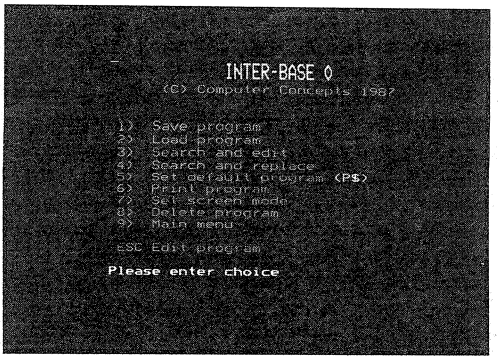
Like the other 'Inter' ROMs, Inter-Base features the ROM-LINK facility, allowing multiple packages to reside in memory at any one time, with the ability to be able to switch between them. In addition, data from other Inter-Link packages can be exchanged with Inter-Base, providing a high degree of integration.

Inter-Base is divided into two distinct sections. These comprise a card index application program, and a programming language. The card index is reached from the main menu once *IBASE has been issued. This is essentially a very simple but flexible card index, allowing searching and sorting, indexing, printing of data and so forth. The number and types of fields may be defined by the user along with positioning of fields on the screen. Any screen mode may be used, and the size of the database is limited only by the available disc storage.

Operation of the card index program is from the 'main' and 'options' menus, or from the Beeb's function keys. It is adequate for simple database work but is not really representative of what Inter-Base is capable.

To release the full power of this package, you need to examine the second and main part of the software. This is the Inter-Base programming language. You have to write your own custom database application using this or get someone to write it for you.

In conversations Computer Concepts have emphasised that if you need more than a simple card index application, Inter-Base will be of little use to you unless you can program it or get someone to do it for you. But, if you can program well in Basic, then programming with Inter-Base should hold no fears.



The Inter-Base language is in fact very much an extension of BBC Basic. All the features of BBC Basic are available to you within Inter-Base, so your database displays can feature graphics and be as sophisticated as you desire. Many of the keywords in Inter-Base are the same as in BBC Basic and commands operate in the same way or with slight variations. Where there are variations, it usually means that the Inter-Base version is more powerful. Alongside those commands common to BBC Basic are many new commands. For example, loop and control statements such as WHILE-ENDWHILE and CASE-ENDCASE are included.String handling is far better supported in Inter-Base than in Basic, as one would expect in a database language. There are commands for upper to lower case matching and conversion, wildcard matching and item selection.

The Inter-Base language is accessed via the main menu which takes you to a program menu. From within the program menu, pressing Escape takes you to a full screen editor which has most of the facilities of a wordprocessor. The Inter-Base language uses no line numbers but relies on the use of labels where necessary. Apart from the language itself, Inter-Base provides both specialised file handling and database commands to manipulate and store data.

With its combined language and database commands, Inter-Base can be used to create some very powerful and complex database applications, and I would not be at all surprised to see the Inter-Base language being used for work other than database applications. Inter-base programs can also be converted to run in sideways RAM.

CONCLUSION

Earlier I said that Inter-Base would be of little use to those who were not able or prepared to do programming. This is true now whilst Inter-Base is newly released, but I am sure that in the future software houses will release applications for it, and no doubt magazines like BEEBUG will feature short programs and utilities.

I must criticise Computer Concepts for two things. Firstly, for not providing an example application written in the Inter-Base language on the utilities disc and, secondly, for not providing a proper programming tutorial in the manual. The tutorial could have been written in such a way as to instruct the user how to customise the application provided to suit his own needs. Computer Concepts say that applications are already being written by several software houses, and that someone is already engaged in writing a book on the programming language. This is fine but adds to the cost for the end user all the time.

When I reviewed Acornsoft's ViewStore in BEEBUG Vol.4 No.6, I described it as the definitive database package for the BBC micro. Also, that all other packages in the future would be judged against it.

In its standard form, as purchased, with only the card index available, Inter-Base is nowhere near as powerful as ViewStore or Masterfile. To get Inter-Base to do what ViewStore can do will require much hard work with the programming language. But, if you are prepared to do that programming or wait until applications are written, Inter-Base can provide a far more powerful and flexible database management system, that can be readily tailored to the user's own specific needs. Couple that with Inter-Base's ability to integrate with and control the other Inter ROMs and you have a productivity software system that has remarkable power and potential.

B

Keeping up with Archimedes

Acorn's newly announced Archimedes range is now building up to full production for the autumn. Mike Williams reports on the latest news and developments.

Last month we covered the launch of Archimedes, concentrating on the main features of the new systems, and the A300 'BBC micro' series in particular. Prices for the A400 series, and for some of the initial add-ons and upgrades, have now been announced (see prices panel).

MS-DOS COMPATIBILITY

I referred in last month's report to the absence of MS-DOS compatibility for the new range, but hinted that maybe Acorn might change their minds. In fact MS-DOS compatibility will be available in two forms. The software emulator will realistically need a 400 series machine to provide sufficient memory space for MS-DOS applications. There will also be a hardware MS-DOS podule with its own on-board processor and memory which will provide MS-DOS emulation on both 300 and 400 series machines.

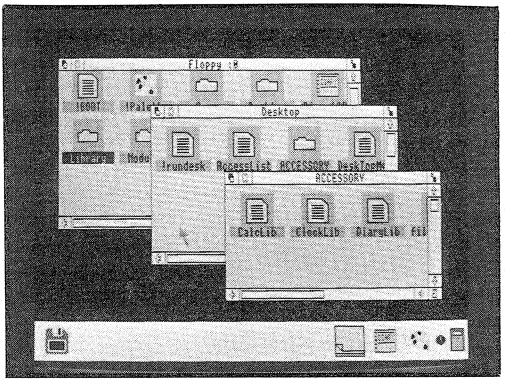
Prices and availability have yet to be announced, but it is certainly good news. If the price is right, and if there is a high degree of compatibility, Acorn's Archimedes may really be able to offer the best of both worlds (Acorn and MS-DOS). However, Acorn have released MS-DOS compatibility in the past (in the form of the 512 co-processor for the Master) with notable lack of support, and consequently success, so we will need to look at this carefully when available.

PODULES

The proposed ROM podule provides five sockets for ROMs and/or RAM, and although some existing ROM based software may work, remember that Archimedes executes ARM machine code and not 6502. Thus any existing ROMs that do work (and Acorn View will do so) will only run under the 6502 emulator. The I/O podule will provide Archimedes with a user port, an A to D (analogue to digital) interface, and a 1MHz bus. Other podules planned for release next year include an Ethernet network card, an SCSI (Small Computer Systems Interface card), modem and IEEE-488 interface.

SOFTWARE

I said last month that I believed the key to Archimedes' ultimate success lay in the development of a wide base of software able to take advantage of the super-fast RISC technology. Apart from BBC Basic V (and built-in ARM assembler), other languages to be released by Acorn are 'C', LISP, ISO-Pascal, Fortran-77, Prolog and Comal. None of these (including 'C' as had been rumoured) will be bundled in with Archimedes. All these languages are scheduled for release in September.



Two other software products of interest to programmers will be the Software Developer's Toolbox and the Software Developer's Debug Tool. The Toolbox includes a number of useful utilities, a comprehensive editor called Twin, an ARM assembler, linker and debugging tool. A variety of terminal style software is also scheduled for release at the end of this year including a full Prestel emulator and a host viewdata system.

All of Acorn's View family of applications software (except ViewPlot) will run on Archimedes via the 6502 emulator, and a

native-mode version of View Professional, the integrated word processor, spreadsheet and database package (see review in this issue), is scheduled for release in November. The popular PC package, Logistix, is another release scheduled for September.

THIRD PARTY SOFTWARE

A number of third party software suppliers, already known to BBC users, are planning software releases for Archimedes. In some cases only the minimum of conversion work has been undertaken to allow the software to run, while others have completely re-written existing software to exploit the full potential of Archimedes. There will also be completely new products made possible by the power of Archimedes. We indicate below some of the more interesting packages likely to be available this year.

MINERVA SYSTEMS

System Delta, the programmable database system, is being completely rewritten for Archimedes, as is System Gamma (see News in Vol.6 No.3) and a new spreadsheet called System Sigma. All three products are fully programmable by the user. All existing, and new, System Delta based applications packages will also be available for Archimedes.

BBC SOFT

A full-feature 32-bit wordprocessor is promised for the autumn, and a comprehensive terminal package.

CLARES MICRO SUPPLIES

Alpha-base and Art Package are being adapted to Archimedes for September, and an Archimedes Toolkit Module is due for release at the same time.

COMPUTER CONCEPTS

Not to be outdone, CC will be providing Disc Doctor II and Word Processor for Archimedes, both running under the WIMP environment. The Word Processor will include a 60,000 word dictionary for instant spelling checks.

BEEBUG

Yes, we are also working in conjunction with Acorn to produce a BEEBUG Modem specifically for Archimedes, together with a new version of our modem software, Command.

AMS and Clares are both working on Desk Top publishing systems for Archimedes, and this is seen as a highly important application area for Acorn's new machine. The ILECC Nimbus graphics library is also being converted to Archimedes, and educational software from Resource, CUP, Longman and SMDP will also become available.

These and other companies will also be releasing adaptations of existing BBC micro software to run under the 6502 emulator on Archimedes. This should provide performance at least as good as the existing Master series. However, this does not necessarily imply that users contemplating upgrading to Archimedes from earlier versions of the BBC micro will be able to use their existing software.

A400 Series Prices

A410	£1399 (£1609)	without monitor
	£1449 (£1666)	with mono. monitor
	£1599 (£1838)	with colour monitor
A440	£2299 (£2644)	without monitor
	£2349 (£2701)	with mono. monitor
	£2499 (£2874)	with colour monitor

Upgrades and Add-ons - Prices

Archimedes/Master Econet	£43.47 (£49.99)
Additional 0.5 Mbytes RAM	£89.00 (£102.35)
Second floppy disc drive	£125.00 (£143.75)
20 Mbyte hard disc & podule	£499.00 (£573.85)
Podule backplane	£35.00 (£40.25)
ROM podule	£39.00
I/O podule	£79.00
MIDI add-on to I/O podule	£29.00
MIDI podule	£59.00
Reference Manual	£19.95

Prices in brackets include VAT.
5% discount on all prices to BEEBUG members.

B

Using the Arm Assembler

Lee Calcraft puts the ARM assembler through its paces, and tests out the RISC processor's 32 bit multiply instruction.

The purpose of this brief section on RISC assembly language is twofold. Firstly it is intended to convey a little of the flavour of programming in ARM assembler, the language of Acorn's 2 micron RISC processor. Secondly, it may help those with access to an Archimedes to get the assembler up and running. The manuals supplied with the Archimedes do not cover ARM assembler; and although the additional Programmer's Manual for the machine contains a wealth of vital information, including details on the use of the assembler, it does not cover the language itself in any detail.

Obviously there is not the space for an exhaustive discussion of each instruction in the present article, but we can certainly put the processor through its paces a little. Our starting point must be the assembler itself. As we said in the last issue, it is used in exactly the same way as that of the Beeb. Listing 1 gives a generalised assembler layout which will work equally well on an Electron, a Model B, Master, Compact or Archimedes. The difference comes when we insert the assembler instructions.

Listing 1

```
10 REM Generalised assembler header for
20 REM Elk, B, B+, Master & Archimedes
30 DIM space 200: REM reserve 200 bytes
40 FOR pass=0 TO 1
50 P%=space
60 [
70 OPT pass*3
80 .start
...
...   assembler code
...
1000 ]
1010 NEXT
```

The first program that we will try is just two instructions in length:

```
90 MOV R0,#&41
210 MOV R15,R14
```

These should be added to listing 1. In order to make the code execute automatically after assembly, you should also add:

```
1050 CALL  start
```

If you now run the program, you should see the assembly listing displayed on the screen, and you should then be returned to Basic. If so, you have successfully run a piece of ARM machine code.

The first instruction, MOV R0,#&41 is a move instruction whose purpose is normally to move the contents of one register into another. In this case we have used it to "move" the constant &41 into register R0. This has a similar effect to the 6502 instruction:

```
LDA #&41
```

except that the ARM has 15 registers at the user's disposal, any of which may be used for arithmetical or logical operations.

The second instruction has the same effect as the 6502 RTS, or ReTurn from Subroutine. In this case, it takes us back to Basic, since that is where the machine code was called from. In fact there is no dedicated RTS instruction in ARM assembler. The instruction that we have used is another version of the move instruction. Its effect is to move the contents of register R14 into R15. This causes a return from subroutine, since register R15 holds the program counter, and when a subroutine is entered, R15 is saved to R14; so the instruction MOV R15,R14 reinstates the program counter to the value which it held immediately before the subroutine was called, so effecting the equivalent of a 6502 RTS.

OS CALLS

Now that we have that rather involved explanation out of the way, we can test out some more instructions. Operating system calls are easily accomplished in ARM assembler, and there is a wider range of routines to choose from. They are implemented using the instruction:

```
SWI n
```

where n defines the type of call. Thus for

example:

```
SWI &05
```

calls OSCLI, and is the direct equivalent of the 6502 call:

```
JSR &FFF7
```

More than 50 SWI calls are listed in the Programmer's Reference Manual, many of which were not available on earlier BBC micros. For example SWI 01 will write a string, SWI &28 will convert a number to a string, and SWI &45 will evaluate an expression.

We can easily test out SWI 0 (OSWRCH - or operating system write character) by inserting it at line 100 in our program, thus:

```
100 SWI 0
```

If you now run the program, you should see a letter "A" sent to the screen. This is because SWI 0 writes to the screen the least significant byte of the contents of register R0; and in the previous line we loaded this with &41, the ASCII code for "A". To print a "Z" in place of the "A", you could change line 90 to:

```
90 MOV R0,ASC("Z")
```

As you can see, the ARM assembler will accept the Basic operator ASC, just as the 6502 assembler in the model B. If you now add the following line:

```
110 SWI 3
```

the routine will terminate with a carriage return, since SWI 03 performs the equivalent of OSNEWL, generating a carriage return and line feed.

SWI MESSAGES

To spice things up a bit, try adding the following lines:

```
120 SWI 1
130 EQUS "ARM Multiply Result"
140 EQUB 7
150 EQUB 0
160 ALIGN
```

These engage the very useful SWI 1 to write a string of text to the screen. At the end of the string we have added EQUB 7 to insert a character 7 into the string, to give a beep (i.e. VDU7), and then an EQUB 0 to tell the operating system that it has reached the end of the string. Line 160 contains the important new assembler directive ALIGN. It serves to align our next instruction to a four-byte boundary, where the ARM processor will expect to find it.

32 BIT MULTIPLY

In order to really put the ARM processor through its paces, we will now try a 32 bit multiply. To keep things simple we will multiply two 16 bit numbers to produce a 32 bit product.

The first potential pitfall is that ARM assembler does not provide an instruction for loading 16 or 32 bit constants into processor registers, except under special circumstances. We will therefore unceremoniously poke our two multiplicands into memory, then load them from memory into registers R2 and R3.

It would of course be possible to load the registers using the CALL statement from Basic, and we will look into this on another occasion.

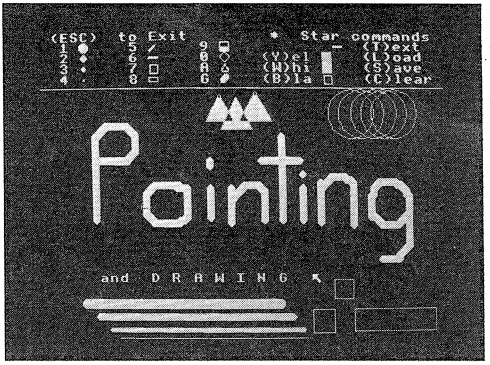
Listing 2

```
 10 REM TEST SWI 0, 1 AND 3  ver 0.5
 20 REM AND PERFORM 32 BIT MULTIPLY
 30 DIM space 200
 40 FOR pass = 0 TO 1
 50 P%=space
 60 [
 70 OPT pass*3
 80 .start
 90 MOV R0,#ASC("Z")
100 SWI 0
110 SWI 3
120 SWI 1
130 EQUS "ARM Multiply Result"
140 EQUB 7
150 EQUB 0
160 ALIGN
170 LDR R1,answer+4
180 LDR R2,answer+8
190 MUL R0,R1,R2
200 STR R0,answer
210 MOV R15,R14
220 .answer
230 EQUD 0
240 EQUD 0
250 EQUD 0
1000 ]
1010 NEXT
1020 :
1030 !(answer+4)=12345
1040 !(answer+8)=33333
1050 CALL start
1060 PRINT!answer
1070 PRINT"CHECK 12345*33333 =";12345*
33333
```

continued on page 15

Beebug Paint and Draw

L. Cullen



With its excellent colour and graphics, the BBC micro has always been a favourite for painting and drawing packages. L.Cullen's program may not turn you into a Picasso or Van Gogh, but it should provide many hours of enjoyment.

The BEEBUG Paint and Draw program uses a mode 1 four-colour screen. The top four lines of text are used to display a key to all the functions of the program, complete with icons in most instances. The remainder of the screen is the drawing area. An arrow pointer indicates a position on the screen, and this may be moved about the drawing area using the cursor keys. Pressing Shift at the same time provides faster pointer movement for covering large areas quickly. Pressing two cursor keys together allows movement in the four diagonal directions.

All the other painting and drawing functions are selected by a single key press, as indicated at the top of the screen, and the arrow pointer will be replaced by the appropriate icon. Once the selected task has been accomplished, pressing the space bar will return the arrow pointer to the screen in the current drawing position ready for your next step. The functions themselves are listed below.

GETTING STARTED

There are two programs to type in (DRAW and DRAW1). The first defines various icons and calls the second (main) program. Do omit the space immediately following each line number (this is included to improve readability only), and do not add any extra spaces, as the second program is a tight fit in the available memory. This program must run with PAGE no higher than &1100 (this may not be feasible with a Watford DFS), and this is set automatically in the last line of the first program.

In the case of the last four options, the drawing colour is always reset to white. In addition, Escape provides (with a check) for an exit from the program, while star commands may also be entered in the top four lines of the screen (use Shift to scroll).

All you have to do now is to get out your electronic painting kit and make a start. If you do produce a masterpiece why not send us a screen dump on tape or disc; if its very good we might even 'hang' it in a future issue of the magazine.

PROGRAM NOTES

The program is well structured with one procedure relating to each of the main functions. Negative INKEY statements are used throughout the program to improve response. In several of these procedures a call is made to the procedure PROCpause to slow down the speed of operation to an acceptable level. Adjusting the parameter will allow you to set this to your own liking.

Master and Compact users can improve some of the facilities in the program because of the additional graphics facilities available on their machines. In particular, a comprehensive colour-flood can be provided (refer to First Course in BEEBUG Vol.6 No.3 for details), which will cope with any shape in one application.

BEEBUG PAINT AND DRAW FUNCTIONS

Options 1, 2, 3 & 4 - Paint 'brushes' in a choice of four sizes. Colour may be changed as you are painting.

Option 5 - A 'pencil' for thin line drawing. Colour may again be changed while drawing.

Option 6 - 'Rubber band' line drawing. Once selected the cursor keys may be used to move one end of a line about the screen. The other, fixed, end is determined by the starting point. Pressing the space bar fixes the line, while Return has the same effect but allows a succession of such lines.

Option 7 - Use this function to draw a square. Moving left or right will control the size. The space bar fixes the result.

Option 8 - Draw a rectangle. Left and right cursor keys control the width, up and down keys the height, of the rectangle.

Option 9 - Colour fill from the current position until a different boundary colour or the edge of the screen is reached. The routine is fairly simple, so more than one application may be needed for filling complicated shapes.

Option 0 - Draw circles with the starting point as the centre. Left and right cursor keys control the size of the radius, and the space bar then fixes the circle.

Option A - For drawing triangles. Left and right cursor keys control the size of the base, and the up and down keys the height.

Option E - This is the rubber. In use it always rubs out to black.

Options R, Y, W, B - Choose one of four colours. The current colour is always marked at the top of the screen for reference.

Option T - Normal mode 1 text may be entered from the current pointer position. Use the Return key in this instance to exit.

Option L - Load a previously saved screen. Any existing screen will be replaced.

Option S - Save the current screen to disc or tape.

Option C - Clear the drawing area. In fact this function paints the screen in the current foreground colour, and may thus be used to provide coloured 'paper'.

Draw

```
   10 REM Program DRAW/Draw 1
   20 REM Version B1.3
   30 REM Author L.Cullen
   40 REM BEEBUG Aug/Sept 1987
   50 REM Program subject to copyright
   60 :
  100 MODE1
  110 VDU23,240,0,124,120,120,124,78,7,2
  120 VDU23,241,60,126,255,255,255,255,1
26,60
  130 VDU23,242,0,24,60,126,126,60,24,0
  140 VDU23,243,0,0,24,60,60,24,0,0
  150 VDU23,244,0,0,0,24,24,0,0,0
  160 VDU23,245,0,2,12,28,56,112,224,64
  170 VDU23,246,0,0,0,255,255,0,0,0
  180 VDU23,247,0,127,65,65,65,65,65,127
  190 VDU23,248,0,0,255,129,129,255,0,0
  200 VDU23,249,255,129,129,129,255,255,
255,255
  210 VDU23,250,24,36,66,129,129,66,36,2
4
  220 VDU23,251,0,24,24,36,102,66,126,0
  230 VDU23,252,28,62,126,254,252,248,24
0,0
  240 VDU23,253,255,255,255,255,255,255,
255,255
  250 IF PAGE>&1100 THEN PAGE=&1100
  260 CHAIN"DRAW1"
```

Draw 1

```
  100 ON ERROR GOTO 3510
  110 DIM os 40,C(4):*FX4,1
  120 C(1)=57:C(2)=48:C(3)=65:C(4)=69
  130 *FX12,4
  140 PROCscreen:PROCvariables:PROCmc
  150 PROCstart
  160 END
  170 :
 1000 DEF PROCvariables
 1010 Left%=0:right%=0:x%=640:y%=512
 1020 colour%=3:posit%=890
 1030 GCOL0,3:MOVE0,855:DRAW1280,855
 1040 MOVE0,855:DRAW1280,855
 1050 PROCprint:ENDPROC
 1060 :
 1070 DEF PROCscreen
 1080 COLOUR3:VDU4:c$=CHR$253
 1090 PRINTTAB(1,0);"(ESC)  to Exit"
 1100 PRINTTAB(23,0);"*  Star commands"
```

```
 1110 FOR Y=1 TO 4:VDU31,2,Y,Y+48,32,Y+2
40:NEXT
 1120 FOR Y=1 TO 4:VDU31,9,Y,Y+52,32,Y+2
44:NEXT
 1130 FOR Y=1 TO 4:VDU31,16,Y,C(Y),32,Y+
248:NEXT
 1140 COLOUR1:PRINTTAB(22,1);"(R)ed ";c$
 1150 COLOUR2:PRINTTAB(22,2);"(Y)el ";c$
 1160 COLOUR3:PRINTTAB(22,3);"(W)hi ";c$
 1170 PRINTTAB(22,4);"(B)la ";CHR$247
 1180 PRINTTAB(32,1);"(T)ext"
 1190 PRINTTAB(32,2);"(L)oad"
 1200 PRINTTAB(32,3);"(S)ave"
 1210 PRINTTAB(32,4);"(C)lear"
 1220 VDU5:ENDPROC
 1230 :
 1240 DEF PROCstart
 1250 REPEAT:a%=INKEY(0):*FX15,1
 1260 IF a%>96 AND a%<123 THEN a%=a% AND
 95
 1270 IF INKEY-1 THEN shift=4 ELSE shift
=1
 1280 IF INKEY-26 AND x%-8*shift>0 THEN
GCOL4,1:MOVEx%,y%:PRINTa$:x%=x%-8*shift:
MOVEx%,y%:PRINTa$
 1290 IF INKEY-122 AND x%+8*shift<1249 T
HEN GCOL4,1:MOVEx%,y%:PRINTa$:x%=x%+8*sh
ift:MOVEx%,y%:PRINTa$
 1300 IF INKEY-58 AND y%+8*shift<850 THE
N GCOL4,1:MOVEx%,y%:PRINTa$:y%=y%+8*shif
t:MOVEx%,y%:PRINTa$
 1310 IF INKEY-42 AND y%-8*shift>32 THEN
 GCOL4,1:MOVEx%,y%:PRINTa$:y%=y%-8*shift
:MOVEx%,y%:PRINTa$
 1320 tx%=x%:ty%=y%
 1330 IF a%=42 THEN PROCstar
 1340 IF a%=49 THEN PROCblob(241,16)
 1350 IF a%=50 THEN PROCblob(242,8)
 1360 IF a%=51 THEN PROCblob(243,4)
 1370 IF a%=52 THEN PROCblob(244,2)
 1380 IF a%=53 THEN PROCpencil
 1390 IF a%=54 THEN PROCline
 1400 IF a%=55 THEN PROCsquare
 1410 IF a%=56 THEN PROCrectangle
 1420 IF a%=57 THEN PROCfill(x%,y%)
 1430 IF a%=65 THEN PROCtriangle
 1440 IF a%=69 THEN PROCrubber
 1450 IF a%=82 THEN colour%=1:PROCmc
 1460 IF a%=89 THEN colour%=2:PROCmc
 1470 IF a%=87 THEN colour%=3:PROCmc
 1480 IF a%=66 THEN colour%=0:PROCmc
 1490 IF a%=48 THEN PROCcircle
 1500 IF a%=84 THEN PROCtext
 1510 IF a%=76 THEN PROCload
 1520 IF a%=83 THEN PROCsave
 1530 IF a%=67 THEN PROCclear
 1540 UNTIL FALSE:ENDPROC
 1550 :
 1560 DEF PROCblob(b,step):exit%=FALSE
 1570 PROCprint:GCOL0,colour%
 1580 a$=CHR$b:MOVE x%,y%:PRINTa$
 1590 REPEAT:PROCpause(step/2-1):PROCcol
our
 1600 IF INKEY-26 AND x%-step>0 x%=x%-st
ep:MOVEx%,y%:PRINTa$
 1610 IF INKEY-122 AND x%+step<1249 THEN
 x%=x%+step:MOVEx%,y%:PRINTa$
 1620 IF INKEY-58 AND y%+step<850 THEN y
%=y%+step:MOVEx%,y%:PRINTa$
 1630 IF INKEY-42 AND y%-step>32 THEN y%
=y%-step:MOVEx%,y%:PRINTa$
 1640 IF INKEY-99 THEN PROCprint:exit%=T
RUE
 1650 UNTIL exit%:ENDPROC
 1660 :
 1670 DEF PROCpencil:exit%=FALSE
 1680 PROCprint:GCOL0,colour%
 1690 MOVE x%,y%:DRAW x%+1,y%
 1700 REPEAT
 1710 MOVEx%,y%:PROCpause(5)
 1720 IF INKEY-26 AND x%-8>0 THEN x%=x%-
8
 1730 IF INKEY-122 AND x%+8<1249 THEN x%
=x%+8
 1740 IF INKEY-58 AND y%+8<850 THEN y%=y
%+8
 1750 IF INKEY-42 AND y%-8>32 THEN y%=y%
-8
 1760 DRAW x%,y%:PROCcolour
 1770 IF INKEY-99 THEN PROCprint:exit%=T
RUE
 1780 UNTIL exit%:ENDPROC
 1790 :
 1800 DEF PROCcolour
 1810 IF INKEY-52 THEN colour%=1:PROCmc
 1820 IF INKEY-34 THEN colour%=3:PROCmc
 1830 IF INKEY-101 THEN colour%=0:PROCmc
 1840 IF INKEY-69 THEN colour%=2:PROCmc
 1850 ENDPROC
 1860 :
 1870 DEF PROCline:exit%=FALSE
 1880 PROCprint:x1%=0:y1%=0:PROCline2
 1890 REPEAT:PROCpause(5)
 1900 IF INKEY-26 AND x%+x1%-16 >0 THEN
PROCline2:x1%=x1%-16:PROCline2
 1910 IF INKEY-122 AND x%+16+x1% <1249 T
HEN PROCline2:x1%=x1%+16:PROCline2
 1920 IF INKEY-58 AND y%+16+y1%<850 THEN
 PROCline2:y1%=y1%+16:PROCline2
 1930 IF INKEY-42 AND y%-16+y1%>32 THEN
PROCline2:y1%=y1%-16:PROCline2
 1940 IF INKEY-74 THEN GCOL0,colour%:MOV
E x%,y%:DRAW x%+x1%,y%+y1%:x%=x%+x1%:y%=
y%+y1%:x1%=0:y1%=0
 1950 IF INKEY-99 THEN GCOL0,colour%:MOV
E x%,y%:DRAW x%+x1%,y%+y1%:x%=x%+x1%:y%=
y%+y1%:PROCprint:exit%=TRUE
 1960 PROCcolour
 1970 UNTIL exit%:ENDPROC
 1980 :
```

```
 1990 DEF PROCsquare:exit%=FALSE
 2000 PROCprint:x1%=10:y1%=10
 2010 x1%=10:y1%=10
 2020 GCOL4,1:PROCsquare2
 2030 REPEAT:PROCpause(1)
 2040 GCOL4,1:PROCsquare2
 2050 IF INKEY-26 AND x%-10+x1% >0 AND y
%+10+y1%>32 THEN x1%=x1%-10:y1%=y1%-10
 2060 IF INKEY-122 AND x%+10+x1% <1249 A
ND y%+10+y1%<850 THEN x1%=x1%+10:y1%=y1%
+10
 2070 GCOL 4,1:PROCsquare2
 2080 IF INKEY-99 THEN GCOL 0,colour%:PR
OCsquare2:PROCprint:exit%=TRUE
 2090 UNTIL exit%:ENDPROC
 2100 :
 2110 DEF PROCrectangle:exit%=FALSE
 2120 GCOL4,1:PROCprint:x1%=50:y1%=25
 2130 PROCrectangle2
 2140 REPEAT:PROCpause(1)
 2150 PROCrectangle2
 2160 IF INKEY-26 AND x%-10+x1%>0 THEN x
1%=x1%-10
 2170 IF INKEY-122 AND x%+10+x1%<1249 TH
EN x1%=x1%+10
 2180 IF INKEY-58 AND y%+10+y1%<850 THEN
 y1%=y1%+10
 2190 IF INKEY-42 AND y%-10+y1%>32 THEN
y1%=y1%-10
 2200 PROCrectangle2
 2210 IF INKEY-99 THEN GCOL0,colour%:PRO
Crectangle2:PROCprint:exit%=TRUE
 2220 UNTIL exit%:ENDPROC
 2230 :
 2240 DEF PROCclear
 2250 VDU4:VDU28,0,4,39,0:CLS
 2260 MOVE100,1020:GCOL0,3:PRINT"Clear t
he Screen (Y/N)? ";:VDU7
 2270 a%=GET AND 95:CLS:VDU26
 2280 IF a%<>89 THEN PROCscreen:PROCmc:E
NDPROC
 2290 VDU24,0;0;1279;848;:GCOL0,colour%+
128:CLG:VDU26
 2300 PROCscreen:PROCvariables:PROCmc
 2310 ENDPROC
 2320 :
 2330 DEF PROCcircle
 2340 PROCprint:x1%=1:PROCcircle2
 2350 REPEAT
 2360 PROCcircle2
 2370 IF INKEY-26 AND x%-8+x1%>0 AND y%-
8+x1%>32 AND y%+8-x1%<850 AND x%+8+x1%<1
249 THEN x1%=x1%-8
 2380 IF INKEY-122 AND x%+8+x1%<1249 AND
 y%+8+x1%<850 AND y%-8-x1%>32 AND x%-8-x
1%>0 THEN x1%=x1%+8
 2390 PROCcircle2
 2400 UNTIL INKEY-99
 2410 PROCcircle2:GCOL0,colour%
 2420 FOR angle=PI/12 TO 2*PI+(PI/12) ST
EP PI/12
 2430 DRAW x%+x1%*COSangle,y%+x1%*SINang
le
 2440 NEXT
 2450 x%=tx%:y%=ty%:PROCprint
 2460 ENDPROC
 2470 :
 2480 DEF PROCtriangle:exit%=FALSE
 2490 x1%=32:y1%=32:PROCprint
 2500 PROCtriangle2:GCOL4,1
 2510 REPEAT
 2520 PROCtriangle2
 2530 IF INKEY-26 AND x%-8+x1%>0 THEN x1
%=x1%-8
 2540 IF INKEY-122 AND x%+8+x1%<1249 THE
N x1%=x1%+8
 2550 IF INKEY-58 AND y%+8+y1%<850 THEN
y1%=y1%+8
 2560 IF INKEY-42 AND y%-8+y1%>32 THEN y
1%=y1%-8
 2570 PROCtriangle2
 2580 IF INKEY-99 THEN GCOL 0,colour%:PR
OCtriangle2:x%=tx%:y%=ty%:PROCprint:exit
%=TRUE
 2590 UNTIL exit%:ENDPROC
 2600 :
 2610 DEF PROCrubber:exit%=FALSE
 2620 PROCprint:GCOL4,1:MOVE x%,y%
 2630 a$=CHR$252:PRINTa$
 2640 REPEAT:PROCpause(5)
 2650 GCOL0,0:MOVE x%,y%:PRINTa$
 2660 IF INKEY-26 AND x%-10 >0 THEN x%=x
%-10
 2670 IF INKEY-122 AND x%+10 <1249 THEN
x%=x%+10
 2680 IF INKEY-42 AND y%-10>32 THEN y%=y
%-10
 2690 IF INKEY-58 AND y%+10<850 THEN y%=
y%+10
 2700 MOVE x%,y%:GCOL4,1:PRINTa$
 2710 IF INKEY-99 THEN GCOL0,0:MOVE x%,y
%:PRINTa$:PROCprint:exit%=TRUE
 2720 UNTIL exit%:ENDPROC
 2730 :
 2740 DEF PROCload
 2750 VDU4:VDU28,0,4,39,0:CLS:VDU26
 2760 COLOUR3:INPUT"Load File: " file$
 2770 file$="LOAD "+file$
 2780 PROCoscli(file$)
 2790 PROCscreen:PROCvariables:PROCmc
 2800 ENDPROC
 2810 :
 2820 DEF PROCsave
 2830 GCOL4,1:MOVE x%,y%:PRINTa$
 2840 VDU4:VDU28,0,4,39,0:CLS
 2850 COLOUR3:INPUT"Save File: " file$
 2860 CLS:VDU26
 2870 file$="SAVE "+file$+" 3000+5000"
 2880 PROCoscli(file$)
 2890 PROCscreen:PROCvariables:PROCmc
```

```
 2900 ENDPROC
 2910 :
 2920 DEF PROCtext:exit%=FALSE
 2930 PROCprint:GCOL0,colour%
 2940 REPEAT:a%=GET
 2950 IF a%=13 THEN PROCprint:exit%=TRUE
:GOTO2970
 2960 MOVEx%,y%:VDU a%:x%=x%+33
 2970 UNTIL exit%:ENDPROC
 2980 :
 2990 DEF PROCprint
 3000 flag=-flag:a$=CHR$240
 3010 GCOL4,1:MOVE x%,y%:PRINTa$
 3020 ENDPROC
 3030 :
 3040 DEF PROCstar
 3050 VDU4,14:VDU28,0,4,39,0:CLS
 3060 INPUT"*" osc$:PROCoscli(osc$)
 3070 PRINT"Press any key to continue";:
REPEAT UNTIL GET
 3080 CLS:VDU26:PROCscreen:PROCmc
 3090 ENDPROC
 3100 :
 3110 DEF PROCpause(t)
 3120 TIME=0:REPEAT UNTIL TIME>t
 3130 ENDPROC
 3140 :
 3150 DEF PROCline2
 3160 GCOL4,1:MOVE x%,y%:DRAW x%+x1%,y%+
y1%
 3170 ENDPROC
 3180 :
 3190 DEF PROCtriangle2
 3200 MOVE x%,y%:DRAW x%+x1%,y%:DRAW x%+
(x1%/2),y%+y1%:DRAW x%,y%
 3210 ENDPROC
 3220 :
 3230 DEF PROCrectangle2
 3240 MOVE x%,y%:DRAW x%+x1%,y%:DRAW x%+
x1%,y%+y1%:DRAW x%,y%+y1%:DRAW x%,y%
 3250 ENDPROC
 3260 :
 3270 DEF PROCsquare2
 3280 MOVE x%,y%:DRAW x%+x1%,y%:DRAW x%+
x1%,y%+y1%:DRAW x%,y%+y1%:DRAW x%,y%
 3290 ENDPROC
 3300 :
 3310 DEF PROCcircle2
 3320 GCOL4,1:MOVE x%,y%:DRAWx%+x1%,y%
 3330 ENDPROC
 3340 :
 3350 DEF PROCfill(x,y)
 3360 PROCprint:GCOL 0,colour%
 3370 background=POINT(x,y)
 3380 GCOL0,background+128
 3390 PROCfill1(x,y,4)
 3400 PROCfill1(x,y-4,-4)
 3410 PROCprint
 3420 ENDPROC
```

continued on page 32

USING THE ARM ASSEMBLER (continued)

The code to accomplish the multiply is given in listing 2, where it follows the short routines discussed earlier. Lines 170 and 180 load the two multiplicands into R1 and R2. Line 190 performs the multiply, setting R0 to the product of R1 and R2. Line 200 then stores the result in memory at the address "answer". After the multiply has been performed, the program checks the result from Basic, and, I am happy *to say, gives the same answer.*

The output from the program is given in listing 3. This usefully illustrates the format of the assembler's output as well as showing that the ARM has got its sums correct. B

Listing 3

```
00008900
00008900
00008900                        OPT pass*3
00008900            .start
00008900 E3A0005A               MOV R0,#ASC("Z")
00008904 EF000000               SWI 0
00008908 EF000003               SWI 3
0000890C EF000001               SWI 1
00008910                        EQUS "ARM Multiply Result"
00008923 07                     EQUB 7
00008924 00                     EQUB 0
00008928                        ALIGN
00008928 E59F1010               LDR R1,answer+4
0000892C E59F2010               LDR R2,answer+8
00008930 E0000291               MUL R0,R1,R2
00008934 E58F0000               STR R0,answer
00008938 E1A0F00E               MOV R15,R14
0000893C            .answer
0000893C 00000000               EQUD 0
00008940 00000000               EQUD 0
00008944 00000000               EQUD 0
Z
ARM Multiply Result 411495885
CHECK 12345*33333 = 411495885
```

A PROFESSIONAL VIEW

Acorn's recently released View Professional is an intriguing product. An integrated word processor, spreadsheet and database software package, and yet quite different to any other such product for the Beeb. Mike Williams reports.

Product	View Professional
Supplier	Acorn Computers Ltd., Cambridge Technopark, 645 Newmarket Road, Cambridge CB5 8PD. Tel. (0223) 214411
Price	£99.95 inc. VAT.

View Professional is a new product in many ways, but also a confusing one. If you thought it was a new member of the View family then you would be wrong, but not entirely. If you thought it offered a word processor, spreadsheet and a database combined in a single package, then that would be the truth, but far from the whole truth. So what is View Professional?

For your money you get a substantial 'new-style' pack containing two ROMs, a 5.25" disc, a 3.5" disc, reference card, keystrip and a manual of just under 200 pages. Thus the one pack will suit all machines, and the software may be used from ROM, or loaded into sideways RAM from disc. There is also provision for its use with a 6502 second processor or Master Turbo.

Once loaded (*VP) you are presented with a spreadsheet style screen with the rows numbered from 1 downwards and the columns labelled with single letters A, B, etc. View Professional uses this same style of screen layout whatever task you are handling.

WORD PROCESSOR

If you want to use View Professional as a word processor just start typing. The text flows across all columns until the edge of the screen is reached, and then continues from the beginning of the next line. Standard 'View' controls are available for editing. The cursor keys, by themselves, with Shift and with Ctrl, move the cursor up, down, left and right. Function key f4 moves to the left margin, f5 to the end of the current line. Characters can be inserted and deleted, and most usefully there is a 'delete word' key as well. Blocks of text can be marked (marked sections are shown inverted black on white) and copied, moved or deleted. Lines of text can be split or joined together.

Escape takes you to an 'options' page. This is where the page layout, top and bottom margins, headers and footers and other parameters are specified (all with defaults) and changed, instead of using View's embedded commands. These are all global settings and apply throughout the current document.

SPREADSHEET

Alternatively, you may use the spreadsheet layout as it looks - as a spreadsheet. Rows and columns define slots. Text as headings or expressions may be entered into any slot. However, a slot containing an expression shows the result of that expression, not the expression itself. As you move around expression slots, the corresponding expression is shown at the top of the screen.

An expression may be a simple number, it may be an expression involving several numbers and operators, and it may include references to results in other slots. Expressions may include the four basic arithmetic operators, relational and logical operators, a range of mathematical functions, or any of the special functions:

CHOOSE COUNT
SUM MAX MIN

As well as numbers, the data used in a spreadsheet may include dates, and there are several functions relating specifically to these.

Expressions may specify a range of slots (in a row or column), or a complete array of slots. A single expression may be copied across a row or column, or a whole table. Re-calculation may be automatic or manual, and there are several other useful and practical features.

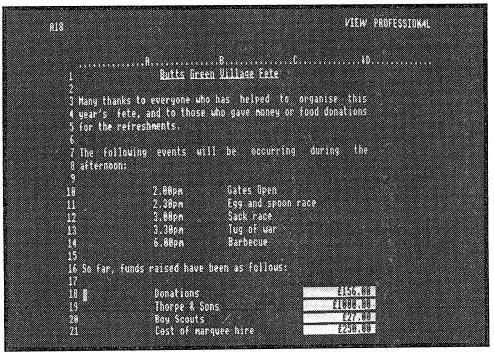
Sheets (and text files) may be saved and loaded. Several sheets saved individually may be linked together, parts of sheets may be saved, or loaded and added to an existing sheet. Indeed, the spreadsheet facilities appear both comprehensive and flexible.

DATABASE

At first sight this seems to be the most limited part of View Professional. Again the same basic spreadsheet style layout is used. Data is entered as already described in the rows and columns, either text, numeric or date format. Data may be sorted alphabetically or numerically into either ascending or descending order. You can also search through the database for information. In a sense that's it, as you'll find little else to remind you of a conventional database package.

INTEGRATION

The key to all that I have described, and the key to the whole concept of View Professional, is that it is a word processor, spreadsheet and database all at the same time. That's why all three tasks use the same basic spreadsheet layout. To think of it as three separate parts is to miss the whole point.



View Professional in use

Tab and Shift-Tab are used to move forwards and backwards from column to column within a row. Associated with every column there is a 'wrap-round' point, which by default is set to the right hand edge of the screen for all columns. Text may start in any column, and as it wraps round you continue from the start of the same column, not necessarily from the lefthand edge of the screen. As a consequence of this approach it is very easy to create multi-column text formats on the screen.

If your text file needs some spreadsheet data incorporated, then just build a spreadsheet at that point - you can also build the spreadsheet elsewhere and incorporate values from the spreadsheet wherever you wish in the text. If you want the data values ordered in some way just sort them - you don't have to transfer the data into a database, sort it and transfer the results back.

All of this is made even more flexible because you can choose your column widths to be individually whatever you need, and change them again later if you wish, and columns may also be added and deleted as and when required. In fact, this all takes some getting used to, and certainly more time than was available for this review would be needed to become fully conversant with all that View Professional has to offer.

There are some further points to note. View Professional includes two printer drivers and a printer driver editor, to control printed output. There are also three utilities supplied on disc which convert existing View, ViewSheet and ViewStore files, allowing them to be loaded into View Professional (perhaps unfortunately there are none to convert View Professional files back).

The manual, in my opinion, is excellent. It is well laid out, the explanations are good (and particularly in the earlier sections very much beginner oriented). The first half is more tutorial in style, the later sections providing a comprehensive reference. There is a glossary of useful terms, a list of error messages and even a reasonably detailed description of View Professional internal file formats for those who wish to write their own file processing programs.

CONCLUSIONS

I found the whole concept of View Professional intriguing, even exciting. It is not a word

continued on page 30

KEYSTRIP GENERATOR

One of the many excellent features of BBC micros is the function keys. But you need a keystrip to describe their functions. Chris Wilson's program provides one of the best utilities we have seen for both editing and printing all the keystrips you will ever need. Text by Mike Williams.

This utility, as well as being a highly useful and practical program, is a really smooth and sophisticated piece of software. Clearly, you will need to run the program to appreciate its quality, but you are unlikely to be disappointed if you do.The purpose of the program listed here is to create, edit and print (on an Epson or Epson compatible printer) function key strips which you may then use with your own software, or indeed with commercial software (useful if you have lost or damaged the original). This is not the first such utility which we have published (see BEEBUG Vol.2 No.9 and Vol.3 No.6), but the ease of use of Chris Wilson's program is outstanding.

Type the program in and save it away to tape or disc. Do take care with the assembler section. Mistakes here could cause your system to hang completely such that Ctrl-Break is the only form of rescue possible. When run the program should produce a screen display with title and authors name at top and bottom. In between there are three rows representing normal, Shift and Ctrl function key actions, with the corresponding function keys marked below.

To get an immediate feel for using the program press the right and left cursor keys, and the function key labels (blank at present) will scroll very smoothly left and right across the screen. To enter a function key label, scroll the key strip until the required key occupies a central position on the screen. Pressing f0 will then allow you to enter/edit the normal key label, pressing f2 will do the same for the Shift-function key label, and f4 the same again for the Ctrl-function key label.

In each case, a function key label may consist of up to two lines of text. Initially the upper line is active and the appropriate text may be entered. As you type away, the label text is continuously centred in the marked space. Pressing Return then allows you to do the same for the bottom line of the label. A further Return completes the editing of that function key label. You may return to the same label position as many times as you wish. When editing, pressing Return alone will leave any label definition as it is (blank if it is already blank). All the function labels may be entered and edited in the same way, scrolling the key strip left and right on the screen as necessary.

If you completely ruin a set of function key labels, pressing 'C' will allow you to clear all the positions ready for you to start again. The program does check your action before proceeding.Once you have completed a function key strip by editing on the screen, you may save it to disc or tape by pressing f6 and specifying a suitable file name. Similarly, pressing f8 will allow you to re-load any set of function key labels previously saved. Thus you may always re-edit any of your function key strips as required.

Pressing 'P' will produce a hard copy printout of your function key strip just the right size for placing under the perspex strip above the function keys. The print routine will work with Epson and Epson compatible printers. Lastly, Escape provides an exit from the program.

Note, should any errors arise, the error number and line number only will be displayed. Pressing any key allows the program to resume, or Escape to exit. Remember that 'errors' may result from incorrect filenames, or from a full disc or catalogue.That's all there is to it, but if you try this utility then we are sure you will agree that it is one of the slickest utilities we have ever published.

PROGRAM NOTES

We do not pretend that this is an easy program to follow, particularly the machine code handling of screen scrolling. However, most of the procedures have quite descriptive names so

that you should have no trouble following the general structure of the program.The load and save routines (lines 1540, 1550 and 1630, 1640 currently allow up to 11 characters for filenames. Editing these lines will allow that to be extended (for ADFS usage for example).

The machine code uses the EQUB and EQUS functions which are not available in Basic I. Those who still have Basic I will need to replace these by calls to corresponding procedures (see BEEBUG Vol.4 No.3 for a full coverage of this).

```
   10 REM Program FKstrip
   20 REM Version B1.4
   30 REM Author  Chris Wilson
   40 REM Beebug  Aug/Sept 87
   50 REM Program subject to copyright
   60 :
  100 ON ERROR GOTO3490
  110 MODE7:HIMEM=&7000
  120 PROCinitialise
  130 PROCfunkystrip
  140 END
  150 :
 1000 DEF PROCfunkystrip
 1010 REPEAT
 1020 IF INKEY(-21) PROCenterlegend(0)
 1030 IF INKEY(-115) PROCenterlegend(1)
 1040 IF INKEY(-33) PROCenterlegend(2)
 1050 IF INKEY(-56) PROCprintkeystrip
 1060 IF INKEY(-118) PROCsavekeystrip
 1070 IF INKEY(-119) PROCloadkeystrip
 1080 IF INKEY(-26) CALLscrollWEST
 1090 IF INKEY(-122) CALLscrollEAST
 1100 IF INKEY(-83) PROCclear
 1110 UNTILO
 1120 ENDPROC
 1130 :
 1140 DEF PROCenterlegend(alter%)
 1150 PROCprint(4,17,">> Enter legend fo
r"+CHR$(129+alter%)+alter$(alter%))
 1160 PROCprint(31,17," FnKey "+STR$(?ke
yno/13)+CHR$135+"<<")
 1170 keytext%=&700E+&138*alter%+?keyno
 1180 PROCinput(17,5+4*alter%,12):keytex
t%=keytext%+&9C
 1190 PROCinput(17,6+4*alter%,12):CALLpr
intlegends
 1200 ENDPROC
 1210 :
 1220 DEF PROCinput(tabX%,tabY%,maxlen%)
 1230 PROCprint(tabX%,tabY%,STRING$(maxl
en%,"-"))
 1240 botlim$=CHR$32:toplim$=CHR$126:inp
ut$=""
 1250 *FX21,0
 1260 key$=GET$:IFkey$=CHR$13THENENDPROC
 1270 REPEAT
 1280 len%=LEN(input$)
 1290 IFkey$=CHR$127THENkey$="":IFlen%>0
THENinput$=LEFT$(input$,len%-1)
 1300 IFkey$<botlim$ORkey$>toplim$THENke
y$=""
 1310 IF LEN(input$)<maxlen% input$=inpu
t$+key$
 1320 PROCcentre:key$=GET$
 1330 UNTIL key$=CHR$13
 1340 IF input$<>"" AND maxlen%=12 PROCi
nsertlegend
 1350 ENDPROC
 1360 :
 1370 DEF PROCprint(XY1,YX1,tew$)
 1380 ?textX=XY1:?textY=YX1:$text=tew$
 1390 CALLprintstring
 1400 ENDPROC
 1410 :
 1420 DEF PROCcentre
 1430 spaces%=maxlen%-LEN(input$)
 1440 spaceL%=spaces%/2
 1450 spaceR%=spaces%-spaceL%
 1460 PROCprint(tabX%,tabY%,STRING$(spac
eL%,"-")+input$+STRING$(spaceR%,"-"))
 1470 ENDPROC
 1480 :
 1490 DEF PROCinsertlegend
 1500 legend$=STRING$(spaceL%," ")+input
$+STRING$(spaceR%," ")
 1510 $keytext%=legend$:?(keytext%+&0C)=
&7C
 1520 ENDPROC
 1530 :
 1540 DEF PROCloadkeystrip
 1550 PROCprint(4,17,"> (LOAD) Enter Fil
ename: .......... <")
 1560 PROCinput(29,17,11)
 1570 IF input$="" GOTO1600
 1580 $&7500="LOAD "+input$+" 7000"
 1590 X%=&00:Y%=&75:CALL&FFF7
 1600 CALLprintlegends
 1610 ENDPROC
 1620 :
 1630 DEF PROCsavekeystrip
 1640 PROCprint(4,17,"> (SAVE) Enter Fil
ename: .......... <")
 1650 PROCinput(29,17,11)
 1660 IF input$="" GOTO1690
 1670 $&7500="SAVE "+input$+" 7000+3A9"
 1680 X%=&00:Y%=&75:CALL&FFF7
 1690 CALLprintlegends
 1700 ENDPROC
 1710 :
 1720 DEF PROCprintkeystrip
 1730 PROCcheckprinter
 1740 IF printeroff GOTO2000
 1750 PROCprint(4,17,">>> Printing Keyst
rip - Be Patient <<<")
```

```
 1760 *FX 3,10
 1770 *FX 201,1
 1780 VDU27,51,24:REM Set Line Feed (24/
256 of an inch)
 1790 PROChozizontbar(48,48):REM Cut Lin
e (first line!)
 1800 PROChozizontbar(15,12):REM Top Lin
e
 1810 PROCprintlegend(6):REM Function Ke
y Numbers
 1820 PROChozizontbar(240,48):REM Bot Li
ne
 1830 PRINT:REM Space
 1840 PROChozizontbar(15,12):REM Top Lin
e
 1850 PROCprintlegend(0):REM Control(1)
Function Key
 1860 PROCprintlegend(1):REM Control(2)
Function Key
 1870 PROChozizontbar(255,24):REM Mid Li
ne
 1880 PROCprintlegend(2):REM Shift(1)
Function Key
 1890 PROCprintlegend(3):REM Shift(2)
Function Key
 1900 PROChozizontbar(255,24):REM Mid Li
ne
 1910 PROCprintlegend(4):REM Normal(1)
Function Key
 1920 PROCprintlegend(5):REM Normal(2)
Function Key
 1930 PROChozizontbar(240,48):REM Bot Li
ne
 1940 PRINT'''''''''''':REM Space (inse
rt 13 "'")
 1950 PROChozizontbar(48,48):REM Cut Lin
e (last line!)
 1960 PRINT'"Cut paper at the first and
last lines, ";
 1970 PRINT"now simply fold the paper in
 half ..."
 1980 *FX 3,0
 1990 *FX 201,0
 2000 CALLprintlegends
 2010 ENDPROC
 2020 :
 2030 DEF PROCcheckprinter
 2040 printeroff=TRUE
 2050 VDU2,1,0,1,0,3
 2060 IF ADVAL(-4)=63 printeroff=FALSE:E
NDPROC
 2070 PROCprint(4,17,">>>>   PRINTER OFF
/UNATTACHED   <<<<"):VDU7
 2080 TIME=0:REPEAT UNTIL TIME>200
 2090 ENDPROC
 2100 :
 2110 DEF PROCprintlegend(line%)
 2120 ?(&709C+line%*&9C)=&0D
 2130 legend$=$(&7000+line%*&9C)
 2140 VDU15
 2150 FOR keylegend%=1 TO 10
 2160 PROCverticalbar:PRINTMID$(legend$,
13*keylegend%+2,12);
 2170 NEXTkeylegend%
 2180 PROCverticalbar:PRINT
 2190 ?(&709C+line%*&9C)=&7C
 2200 ENDPROC
 2210 :
 2220 DEF PROCverticalbar
 2230 PROCbitimage(0,2):PROCbitimage(255
,2):PROCbitimage(0,2)
 2240 ENDPROC
 2250 :
 2260 DEF PROChozizontbar(byte1%,byte2%)
 2270 PROCbitimage(0,2):PROCbitimage(byt
e1%,2)
 2280 FORdrawline%=1TO10
 2290 PROCbitimage(byte2%,88):PROCbitima
ge(byte1%,2)
 2300 NEXTdrawline%
 2310 PRINT
 2320 ENDPROC
 2330 :
 2340 DEFPROC bitimage(image%,bytes%)
 2350 VDU27,76,bytes%,0:PRINTSTRING$(byt
es%,CHR$(image%));
 2360 ENDPROC
 2370 :
 2380 DEF PROCasemblecode
 2390 osbyte=&FFF4
 2400 FOR pass%=0 TO 2 STEP 2
 2410 P%=&7700
 2420 [OPTpass%
 2430 .scrollWEST
 2440 LDAkeyno:CMP#&00
 2450 BEQstopWEST
 2460 LDX#&0D
 2470 .loopWEST:DECkeyno:JSRprintlegends
:DEX:BNEloopWEST
 2480 .stopWEST:RTS
 2490 .scrollEAST
 2500 LDAkeyno:CMP#&75
 2510 BEQstopEAST
 2520 LDX#&0D
 2530 .loopEAST:INCkeyno:JSRprintlegends
:DEX:BNEloopEAST
 2540 .stopEAST:RTS
 2550 .printlegends
 2560 TXA:PHA
 2570 LDA#&13:JSRosbyte
 2580 LDX#&00:LDYkeyno
 2590 .loopscroll
 2600 LDA&7001,Y:STA&7CE0,X:LDA&709D,Y:S
TA&7D0C,X
 2610 LDA&7139,Y:STA&7D90,X:LDA&71D5,Y:S
TA&7DBC,X
 2620 LDA&7271,Y:STA&7E40,X:LDA&730D,Y:S
TA&7E6C,X
 2630 LDA&73A9,Y:STA&7EF0,X
 2640 INY:INX:CPX#&26
```

```
 2650 BNEloopscroll
 2660 PLA:TAX
 2670 RTS
 2680 .printstring
 2690 LDAtextX:STA&70
 2700 LDA#&7C:STA&71
 2710 LDYtextY
 2720 .pokeaddr
 2730 CLC:LDA&70:ADC#&2C:STA&70
 2740 LDA&71:ADC#&00:STA&71:DEY
 2750 BNEpokeaddr
 2760 LDY#&00
 2770 .printloop
 2780 LDAtext,Y:CMP#&0D:BEQstop
 2790 STA(&70),Y:INY:BNEprintloop
 2800 .stop:RTS
 2810 .keyno EQUB&00
 2820 .textX EQUB&00
 2830 .textY EQUB&00
 2840 .text EQUS"Enough space"
 2850 ]:NEXTpass%
 2860 ENDPROC
 2870 :
 2880 DEF PROCinitialise
 2890 VDU23,1,0;0;0;0;
 2900 *FX 4,1
 2910 ?&FE00=&01:?&FE01=&2C:?&FE00=&02:?
&FE01=&38
 2920 ?&FE00=&06:?&FE01=&17:?&FE00=&07:?
&FE01=&1B
 2930 PROCasemblecode:PROCstringsetup
 2940 PROClegendsetup:PROCscreensetup
 2950 CALLprintlegends
 2960 ENDPROC
 2970 :
 2980 DEF PROClegendsetup
 2990 FOR blanktext%=&7000 TO &73A8 STEP
 &9C
 3000 $blanktext%=blank$
 3010 NEXTblanktext%
 3020 FOR alterkey%=0 TO 2
 3030 legaddr1%=&7002+alterkey%*&138
 3040 legaddr2%=legaddr1%+&8F
 3050 $legaddr1%=alter$(alterkey%):?(leg
addr1%+&07)=32
 3060 $legaddr2%=alter$(alterkey%):?(leg
addr2%+&07)=32
 3070 NEXTalterkey%
 3080 FOR keynumber%=0 TO 9
 3090 keyaddr%=&73B9+keynumber%*13
 3100 $keyaddr%="Key f"+STR$(keynumber%)
:?(keyaddr%+6)=32
 3110 NEXTkeynumber%
 3120 !&80=&BEEB
 3130 ENDPROC
 3140 :
 3150 DEF PROCstringsetup
 3160 fnkey$=CHR$154+"j"+CHR$135+STRING$
(38," ")
 3170 line1$=CHR$154+"h"+STRING$(40,",")
 3180 line2$=CHR$154+"*"+STRING$(40,",")
 3190 controls$=" "+CHR$129+CHR$157+CHR$
135+CHR$141
 3200 heading1$=controls$+"Function Keys
trip Editor And Printer   "
 3210 heading2$=controls$+" By Chris Wil
son  -  BEEBUG 1987(C)    "
 3220 dashs$="  "+STRING$(42,"=")
 3230 blank$=STRING$(13,"|"+STRING$(12,"
"))
 3240 DIMalter$(2)
 3250 FORI%=0TO2:READalter$(I%):NEXTI%
 3260 ENDPROC
 3270 :
 3280 DATA "CONTROL","SHIFT  ","NORMAL "
 3290 :
 3300 DEF PROCscreensetup
 3310 PRINTdashs$+heading1$+heading1$+da
shs$;
 3320 FORI%=0TO2
 3330 graphics$=CHR$(145+I%)
 3340 PRINTgraphics$+line1$+"4"+graphics
$+fnkey$+graphics$+"5";
 3350 PRINTgraphics$+fnkey$+graphics$+"5
"+graphics$+line2$+"%";
 3360 NEXT
 3370 PRINTCHR$148+line1$+"4"+CHR$148+fn
key$+CHR$148+"5"+CHR$148+line2$+"%";
 3380 PRINTdashs$+heading2$+heading2$;
 3390 PROCprint(0,22,dashs$)
 3400 ENDPROC
 3410 :
 3420 DEF PROCclear
 3430 PROCprint(4,17,"Clear Screen Y/N"+
STRING$(20,CHR$32))
 3440 a$=CHR$(GET AND 95)
 3450 IF a$="N" CALLprintlegends:ENDPROC
 3460 IF a$="Y" PROClegendsetup:CALLprin
tlegends:ENDPROC
 3470 GOTO3440
 3480 :
 3490 *FX3,0
 3500 *FX15,0
 3510 *FX124
 3520 *FX201,0
 3530 IF ERR<>17 THEN 3600
 3540 PROCprint(4,17,"Do you really want
 to exit Y/N ?"+STRING$(8,CHR$32))
 3550 a$=CHR$(GET AND 95)
 3560 IF a$="N" CALLprintlegends:GOTO 13
0
 3570 IF a$<>"Y" THEN 3550
 3580 *FX4
 3590 MODE7:END
 3600 PROCprint(4,17,">> ERROR "+STR$ERR
+" at line "+STR$ERL+" <<"+STRING$(10,CH
R$32)):a$=GET$:CALLprintlegends:GOTO 130
```

B

ANIMATED TITLING

Why not add some zip to your programs with these six procedures from Alexander Goh. They can pull in text from the left, right, top or bottom of the screen, or "shoot" characters onto the screen one at a time. There is even a Morse code option and the procedures will work in all modes.

Have you felt that a perfectly good piece of software was marred by poor presentation? If this has ever happened to you, you will know how frustrating it can be. Coloured menu screens are all very well, but they lack originality, and all look the same. This article presents six short procedures that will "pull-in" or display your text in a variety of different styles. Any or all of these procedures may be incorporated into your own programs to enhance your title screen displays, and you won't have to bother at all about how the effects are produced. So that you may see just what can be achieved, the procedures have been combined into a complete demonstration program. This also acts as a guide to how the procedures should be called when you use them in your own programs. To run the demo program, first type it in and save it away. Then stand back as you run the program, and wait for the show to begin.

THE PROCEDURES DESCRIBED

The full set of procedures, and their functions are listed in Table 1. In each case the parameters x, y, m$ and c have the same meaning. The text co-ordinates x and y specify the final position for a message on the screen, m$ is the message itself, and c is the colour of the message.The procedures will work in any screen mode (including mode 7), as the computer automatically calculates modes, and maximum x and y positions. This feature is illustrated in the demonstration program. The procedure PROCrmode (and the data at line 1640) is used by all the other procedures to determine the current screen mode, and should always be included with any of the other procedures. Note that the variables m, mx, my and m1$ are used by all the procedures, and should not be used for any other purpose in your programs.

USING THE PROCEDURES

To add the procedures to your own programs delete all but the procedures themselves from the demo program (the procedures run from line 1000 to line 1650 inclusive). Then create a spooled version of the procedures by typing:

```
*SPOOL <filename>
LIST
*SPOOL
```

To add the spooled procedures to any of your own programs, load in the program to be modified, and then type:

```
*EXEC<filename>
```

ignoring any subsequent occurrences of the "Mistake" error. When you next list your program, you will find that the procedures will have been appended to it. Add the procedure

1. **PROCin(x,y,m$,s,c)** - This will pull a message onto the screen one character at a time. The parameter 's' can be set to 0 or 1. If 1, sound effects are included, if set to 0 then they are not.

2. **PROCleft(x,y,m$,c)** - This will pull a message in from the left of the screen.

3. **PROCright(x,y,m$,c)** - As PROCleft above, except that the message is pushed in from the right of the screen.

4. **PROCup(x,y,m$,c)** - PROCup can be used to bring text from the bottom of the screen up to TAB position y - equivalent to winching it up.

5. **PROCdown(x,y,s,m$,c)** - PROCdown drops a message from the top of the screen to TAB position y. The parameter 's' specifies the TAB position from which to start dropping the message. This is included so that text can be dropped down to the bottom or the centre of the screen without corrupting the screen title or anything else displayed at the top of the screen.

6. **PROCmorse(x,y,m$,c)** - PROCmorse displays your message on the screen one character at a time, with bleeping noises. This is supposed to be an imitation of a "Morse Code" de-coder, but you will have to make up your own mind!

calls as necessary and save the program away ready for use.

```
   10 REM Program Scroll
   20 REM Version B1.1
   30 REM Author Alexander Goh
   40 REM BEEBUG Aug/Sept 1987
   50 REM Program subject to copyright
   60 :
  100 MODE7
  110 volume=-15
  120 VDU23,1,0;0;0;
  130 PROCdown(4,2,1,STRING$(27,"="),3)
  140 PROCin(4,1,"AN EXERCISE IN PRESENT
ATION",1,2)
  150 PROCup(2,5,"6 short routines that
will work",1)
  160 PROCleft(2,7,"wonders for the pres
entation of",3)
  170 PROCright(10,9,"your programs",2)
  180 PROCleft(8,13,"by Alexander Goh",6
)
  190 PROCdown(6,22,16,"Press SPACE to c
ontinue",5)
  200 *FX21
  210 REPEAT UNTIL GET=32
  220 CLS
  230 PROCin(10,1,"Starring:",0,2)
  240 PROCdown(10,5,3,"PROCdown",6)
  250 PROCin(10,7,"PROCin",0,1)
  260 PROCleft(10,9,"PROCleft",2)
  270 PROCright(10,11,"PROCright",3)
  280 PROCup(10,13,"PROCup",5)
  290 PROCmorse(6,15,"and PROCmorse",3)
  300 PROCleft(3,18,"and did you know th
at ....",7)
  310 TIME=0:REPEATUNTILTIME>500
  320 MODE2:VDU23;8202;0;0;0;
  330 PROCleft(0,2,"They will work in",9
)
  340 PROCup(8,4,"ANY",13)
  350 PROCin(4,6,"screen mode",1,12)
  360 END
  370 :
 1000 DEFPROCin(x,y,m$,s,c):LOCAL a,b
 1010 PROCrmode
 1020 FOR a=1 TO LENm$
 1030 FOR b=mx TO x+a-1 STEP-1
 1040 PRINTTAB(b,y)MID$(m$,a,1)" "
 1050 NEXTb
 1060 IF s THEN SOUND1,0,(255/a)*a,1:SOU
ND0,volume,1,1
 1070 NEXTa
 1080 ENDPROC
 1090 :
 1100 DEFPROCleft(x,y,m$,c):LOCALa,b
 1110 PROCrmode
 1120 FORa=1 TO LENm$
 1130 PRINTTAB(mx-a,y)MID$(m$,1,a);
 1140 NEXT
 1150 IF mx-a<=x ENDPROC
 1160 FOR b=mx-a TO x STEP-1
 1170 PRINTTAB(b,y)m$" "
 1180 TIME=0:REPEAT UNTIL TIME>2
 1190 NEXT
 1200 ENDPROC
 1210 :
 1220 DEFPROCmorse(x,y,m$,c):LOCAL a,b
 1230 PROCrmode
 1240 FOR a=1 TO LENm$
 1250 PRINTTAB(x-1+a,y)MID$(m$,a,1)
 1260 FOR b=1 TO 3:SOUND1,volume,200,RND
(4):SOUND1,0,0,1:NEXT
 1270 NEXT:ENDPROC
 1280 :
 1290 DEFPROCright(x,y,m$,c): LOCAL a,b
 1300 PROCrmode
 1310 FOR a=0 TO x-1
 1320 PRINTTAB(a,y)" "m$
 1330 TIME=0:REPEAT UNTIL TIME>2
 1340 NEXT
 1350 ENDPROC
 1360 :
 1370 DEFPROCdown(x,y,s,m$,c):LOCAL a
 1380 PROCrmode
 1390 FOR a=s TO y
 1400 PRINTTAB(x,a-1)m1$;
 1410 PRINTTAB(x,a)m$;
 1420 TIME=0:REPEAT UNTIL TIME>2
 1430 NEXT
 1440 ENDPROC
 1450 :
 1460 DEFPROCup(x,y,m$,c):LOCAL a
 1470 PROCrmode
 1480 FOR a=my-1 TO y STEP-1
 1490 PRINTTAB(x,a+1)m1$;
 1500 PRINTTAB(x,a)m$;
 1510 TIME=0:REPEAT UNTIL TIME>2
 1520 NEXT
 1530 ENDPROC
 1540 :
 1550 DEFPROCrmode:LOCAL A%,mode%
 1560 A%=135:mode%=((USR &FFF4) AND &FF0
000) DIV &10000
 1570 RESTORE
 1580 REPEAT
 1590 READm,mx,my
 1600 UNTILm=mode%
 1610 IF mode%<>7 COLOURc ELSE m$=CHR$(1
28+c)+m$
 1620 IF mode%<>7 m1$=STRING$(LEN(m$),CH
R$32) ELSE m1$=CHR$(128+c)+STRING$(LENm$
-1,CHR$32)
 1630 ENDPROC
 1640 DATA0,79,31,1,39,31,2,19,31,3,79,3
1,4,39,31,5,19,31,6,39,24,7,39,24
 1650 DATA0,79,31,1,39,31,2,19,31,3,79,3
1,4,39,31,5,19,31,6,39,24,7,39,24
```

B

6502 DEBUG

Debugging machine code can be a very slow process unless you have the right debugging aids. Lee Calcraft presents a simple 6502 debugging tool which displays the current address, the contents of the accumulator and X and Y registers, and the state of the processor's seven flags.

The essence of this debugging aid is brevity and simplicity. It assembles to a short piece of machine code (&D9 bytes in length), which may be called with a series of JSR instructions from the program which you are trying to debug. A sample output is given in table 1. The first field is the address, in hex, of the location in the host program immediately following the call to the debugger. Next come the contents of the accumulator and X and Y registers, again in hex; and finally the state of the processor's seven flags. As you can see, in the first line the zero and break flags are set, and all others unset. But more of this in a moment.

GETTING STARTED

We should start by getting the debug routine up and running. Type in listing 1 (at the end of the article), and save the program away before running it. When it is run it displays the predicted length of the code (stored in line 640) together with its actual assembled length. If there is a mismatch, you must assume that you have made transcription errors of some kind, which should be tracked down before executing the code. When all is well, you should save a copy of the machine code (under the name "debug") by copying the on-screen prompt.

TESTING THE DEBUGGER

The short program in listing 2 is given as a test for the debugger. It illustrates how the debug routine is called, and gives an idea of its power. To use the debugger on this program, first type in listing 2, and save it away. Then type:

```
*LOAD debug
```

to install the debugger.

Listing 2

```
 10 REM DEBUG TEST RUN
 20 REM Version B 0.7
 30 MODE7
 40 DIM code 100
 50 debug=&900
 60 FOR pass=0 TO 3 STEP 3
 70 P%=code
 80 [
 90 OPT pass
100 .start
110 JSR debug+3
120 LDA #&FF:LDX #&AA:LDY #0
130 JSR debug
140 JSR debug
150 ADC #5
160 JSR debug
170 LDY #3
180 .loop
190 DEY
200 JSR debug
210 BNE loop
220 RTS
230 ]
240 NEXT
250 CALL start
```

Now when you run listing 2, you should get an output similar to that given in table 1. The heading is given by the JSR to debug+3 (performed in line 110 of listing 2). Line 120 then loads the accumulator and X and Y registers with the values &FF, &AA and 0 respectively. Debug is called immediately after this (in line 130), and you can see the result by looking at the first line of output in table 1. This

Assembled version of Listing 2

```
0F4A
0F4A
0F4A                    OPT pass
0F4A           .start
0F4A 20 03 09           JSR debug+3
0F4D A9 FF              LDA #&FF
0F4F A2 AA              LDX #&AA
0F51 A0 00              LDY #0
0F53 20 00 09           JSR debug
0F56 20 00 09           JSR debug
0F59 69 05              ADC #5
0F5B 20 00 09           JSR debug
0F5E A0 03              LDY #3
0F60           .loop
0F60 88                 DEY
0F61 20 00 09           JSR debug
0F64 D0 FA              BNE loop
0F66 60                 RTS
```

gives an address of &0F56 (your address will probably be different). This is the address of the start of the instruction following the debug call.

The contents of the A, X and Y registers are then correctly given, and the flag display shows that the break flag and the zero flag are set. The former, which indicates the type of interrupt which last occurred, is of no interest here, but the latter is set because the last operation which we performed was to load Y with zero.Line 140 repeats the call to debug to demonstrate that calling the debugger does not alter ANY of the registers or flags. Line 150 then adds 5 to the accumulator. This causes the carry flag to be set (since the result of the addition exceeds &FF), and the zero flag to be unset (since the result of the addition is non-zero).

Lastly I have included an illustration of how the debug routine may be used to check the progress of a loop. The loop itself appears between lines 180 and 210, and is entered with Y equal to 3. This is decremented just before the JSR to debug, and the loop exits when Y reaches zero. The last 3 entries in table 1 confirm that the loop is indeed performed 3 times, with values of Y from 2 to 0 at the point of the test. Note also that, as expected, the three addresses displayed here by debug are identical.

Table 1: Output from running Listing 2

Addr	A	X	Y	NVBDIZC
0F56	FF	AA	00	0010010
0F59	FF	AA	00	0010010
0F5E	04	AA	00	0010001
0F64	04	AA	02	0010001
0F64	04	AA	01	0010001
0F64	04	AA	00	0010011

NOTES ON LISTING 1

1. The assembly address of listing 1 is given in line 110 as &900. This can be altered to any other convenient point (&DD00 is ideal on the Master and Compact). But you must remember to change the corresponding address in listing 2, or you will have problems!

2. As you might expect, the debugger preserves absolutely ALL registers - it would be quite useless if it did not.

3. For work space, debug uses only 4 locations at the bottom of the stack (&100 - &103).

4. The debug program contains two portable routines which may easily be incorporated into other software. The routine "write" listed from line 350 takes the contents of the accumulator and displays it as a pair of hex characters. The routine calls "hdloop" at line 400, which in turn calls OSWRCH. The complete "write" routine preserves all registers.

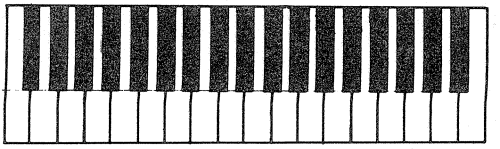
The second routine which may be of interest is called "heading". It will print any embedded message included at the location "table". See lines 590 onwards for the format of the text, which should be terminated by a zero byte. The routine "heading" does not preserve registers.

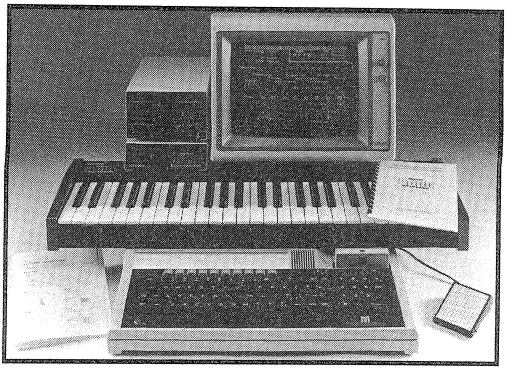
Listing 1

```
 10 REM Program 6502 Debug
 20 REM Version B 0.2C
 30 REM Author  Lee Calcraft
 40 REM Beebug  Aug/Sept 1987
 50 REM Program subject to copyright
 60 :
100 MODE7
110 code=&900
120 oswrch=&FFEE
130 osnewl=&FFE7
140 FOR pass=0 TO 3 STEP 3
150 P%=code
160 [
170 OPT pass
180 :
190 .start
200 JMP help
210 JMP heading
220 .help
230 PHP:CLD:STA &100:PLA
240 STA &101:PLA:STA &102
250 PLA:STA &103:PHA:LDA &102:PHA
260 LDA &101:PHA:INC &102:BNE noflo
270 INC &103:.noflo:LDA &103:JSR write
280 LDA &102:JSR write:JSR space
290 JSR space:LDA &100:JSR write
300 JSR space:TXA:JSR write
310 JSR space:TYA:JSR write:JSR space
320 JSR space:PLA:PHA:JSR bin
330 JSR osnewl:LDA &100:PLP:RTS
340 :
```

Continued on page 46

THE MUSIC 4000





It has been a long wait for the Music 4000, but this keyboard to accompany the Music 5000 is at last here. Ian Waugh, our music expert, has been playing with the new device.

Product	Music 4000
Supplier	Hybrid Technology Limited, Unit 3, Robert Davies Court, Nuffield Road, Cambridge CB4 1TP. Tel. (0223) 316910
Prices	£169 (inc. footswitch) Symphony Upgrade £47.00 Footswitch £11.50

Hybrid Technology's Music 5000 was reviewed last November in BEEBUG Vol.5 No.6. It is still the only currently affordable alternative to making music with MIDI. But whereas MIDI is basically performance orientated, the Music 5000 is more compositionally based. Many musicians find this the ideal way to work, but others prefer to compose at the keyboard. In fact, there is a case to be made for both methods, and I switch from one to the other depending upon the application.

The Music 4000 adds a 4-octave full-size keyboard to Hybrid's Music System. It's Italian, has a light springy touch and is nicely finished in slate grey plastic-coated steel. A generous length of ribbon cable plugs into the User Port and a footswitch plugs into the keyboard. It must be used only with the Music 5000, so do refer to November's review for compatibility. That review also describes Music 5000 modules which will be referred to in this article.The software on disc contains new keyboard modules which are fully integrated into the AMPLE system (the programming language for the Music 5000). A software upgrade kit is available for owners of ATPL's Symphony Keyboard. The disc also contains six music demos - you'll probably play these first - which show what Hybrid's Music System can do, at least in part. They certainly prove that the 5000 can produce full, heavy sounds.

The 4000 adds three main features to the system: real-time recording, step-time recording and the ability to play Music 5000 sounds from the keyboard.Two files contain over 80 new instrument sounds and special effects. These can be called up, played and edited directly from the Keyboard module and in this mode the footswitch acts as a sustain pedal. Some of the sounds verge on the brilliant: lush strings, honky-tonk piano and all manner of instruments and effects.

You can use the keyboard to enter music directly into Notepad or the Staff Editor in step-time, that is one note at a time. The pitch is determined by the key you play and chords are entered simply by pressing more than one note. In the Staff Editor they appear in the same order that you play them, so if you enter chords from the bottom up you can easily give each line of notes its own voice and process them

MIDI: The acronym MIDI stands for Musical Instrument Digital Interface. This has become something of a standard for connecting and linking various electronic instruments and a computer like the Beeb, and the MIDI interface is produced by Electromusic Research. For more information refer to the article "Music System Update" in Vol.4 No 10.

SPECIAL KEYBORD EFFECTS

Six special keyboard effects are shown on screen, and two affect pitch: 'Trans' transposes the keyboard range up or down in semitones and 'Scale' affects the pitch scaling in 16th of a semitone units. The default is 16 which sets adjacent notes a semitone apart - like a normal keyboard. Negative values invert the intervals: -16 turns the keyboard upside down with the low notes at the top and vice versa.

'Expand' can turn a single key-press into a short sequence of notes, a full chord or an echo effect. The pattern is programmed by pressing the footswitch and then the required notes. During play the pattern responds to the pitch of the note played and the time interval between successive key-presses. In instruments with more than one voice, 'Reduce' will reduce the volume of successive voices as they are played. In conjunction with Expand, Reduce can produce echoes and decaying arpeggios.

'Spread' is similar to Reduce except it alters the stereo position of each successive voice rather than its volume. Many stereo effects are possible including sounds which spread from one side to the other and which alternate between left and right positions.

Finally, 'Split' assigns each note in each of the four octaves to play a different voice rather than the corresponding pitch. It would be used with a mix, perhaps of special sound effects, rather than as a single instrument.

independently in the Mixing Desk.The duration of a note or chord in Notepad can be increased by pressing the foot-switch. In the Staff Editor the footswitch increases the duration and smaller values can be entered as usual with Shift and the cursor keys. In both editors you can use a combination of the keyboard and the full range of Music 5000 editing commands.

In the 'Recorder' module you enter parts in real-time, that is to say that they are recorded live as you play. You must decide how many voices each part will have, and assign the eight available voices accordingly. There is a sample mix on the disc to start you off, but it's very easy to create your own mixes.

In the Recorder you can alter the tempo, specify the number of beats per bar and their duration. There is a 'quantise' function which sets each note onto the smallest specified division of a bar. With this facility and by recording at a slow tempo you can produce quite exact recordings.You can get a two-bar count in, and alter the metronome volume. Any of the parts can be put in a 'Backing' line which means they will play as you record other parts. This is just like traditional multi-track recording but not always possible from MIDI. It's very helpful. You can also boost the volume of the keyboard against the backing while recording.

Just as you can construct parts in sections in the Notepad, you can record sections of parts in real-time, too. More than that, the system can convert your real-time performance into AMPLE notation in the Notepad or traditional notation in the Staff Editor. This is no mean programming feat and there are few pieces of software which can perform this task and none, as far as I know, for the BBC Micro. It just shows how powerful and fully integrated all the modules are.

The manual is similar in style to the 5000 manual and leads you fairly painlessly through the operation of the new modules. It helps if you have a working knowledge of the 5000 and you are referred to the User Guide occasionally to avoid duplication of information.

The 5000 and 4000 together form a complete, fully integrated music system. AMPLE gives you a most powerful and flexible composing environment, and the variety of modules means you can use those which most suit the music you are writing and the input method you prefer: AMPLE notation, traditional notation and now real- and step-time keyboard note entry.

The facilities the system offers still far outweigh any other system at the price. If you're a Beeb owner with a bent for music it really deserves serious consideration and I can see many uses for it in education. Hybrid are committed to developing a totally integrated music system and the Music 4000 is the next step up. B

Custom Clearance

Give your programs that professional touch by incorporating one of Barry Christie's customised screen clear routines.

I have selected three different clear routines for this month's column. They come from a set of 29 which I threw together one rainy evening (true!). I'll describe them very briefly to start with, then go on to look at each in some detail. The first clears the screen from both sides at once, like a closing theatre curtain. It works in modes 0-2. The second acts more like a closing venetian blind, and functions in all modes but 7; while the third is again limited to modes 0-2, and sucks the screen into a kind of whirlpool in the centre. Additionally, the third is accompanied by a routine to act in reverse, by exploding a new image onto the monitor.

All programs should be typed in and saved before running. They assemble to &900 (though this may be adjusted to suit your convenience), and the last line of each gives instructions for saving the machine code generated when the programs are run. Each program incorporates a demonstration, the first using mode 2, and the remainder offering a choice of modes. Note however that only the last of the three routines works in shadow screen modes.

CLOSING CURTAINS

When you run the first of the three routines, a screen will first be created in order to properly demonstrate the effect of the clear. The "curtains clear" will, as we said, operate in modes 0-2, though the example only demonstrates its use in mode 2. As you will see, it is clean and smoothly executed. As with each of the three routines, it automatically checks which mode it is being used in. This means that it is completely portable. You can either incorporate the assembler listing in your program, or save the machine code which it generates, and then just activate the clear directly from disc with:

```
*filename
```

The last line of each routine gives the parameters for saving the code in this way.

I said that this particular clear only works in modes 0-2. In fact it will also work in modes 3-6, but the effect is different. It clears from the left and right hand edges simultaneously, as it does in the 20K modes. But it clears both sections with an offset slatting effect. To try the routine in other modes than 2, just alter line 120 of the program, but remember that lines 130-150, which draw a design on the screen, will have no effect in modes 3 or 6, so you will be clearing an already blank screen, which is never so impressive!

```
  10 REM Program Curtains Clear
  20 REM Version B 0.9
  30 REM Author  Barry Christie
  40 REM Beebug  Aug/Sept 1987
  50 REM Program subject to copyright
  60 :
 100 MODE7
 110 PROCassemble
 120 MODE2
 130 GCOL0,135:CLG:GCOL0,4
 140 MOVE 280,200:MOVE 640,824
 150 PLOT 85,1000,200:GCOL0,0
 160 A=INKEY(100)
 170 CALL clearscreen
 180 END
 190 :
1000 DEF PROCassemble
1010 FOR pass%=0 TO 2 STEP 2
1020 P%=&900
1030 [:OPT pass%
1040 .clearscreen
1050 LDA #&A0:LDX #&50:JSR &FFF4
1060 STX &70:STY &71
1070 LDA #&78:STA &72:LDA #&32:STA &73
1080 LDA #&28:STA &74
1090 .clear
1100 LDA #&13:JSR &FFF4:LDX #&20
1110 .column
1120 LDA #&00:LDY #&07
1130 .block
1140 STA (&70),Y:STA (&72),Y:DEY
1150 BPL block:CLC:LDA &70:ADC #&80
1160 STA &70:LDA &71:ADC #&02:STA &71
1170 CLC:LDA &72:ADC #&80:STA &72
1180 LDA &73:ADC #&02:STA &73
1190 DEX:BNE column:SEC:LDA &70
1200 SBC #&F8:STA &70:LDA &71
1210 SBC #&4F:STA &71:SEC:LDA &72
1220 SBC #&08:STA &72:LDA &73:SBC #&50
```

```
1230 STA &73:DEC &74:BNE clear:RTS
1240 ]:NEXT pass%
1250 ENDPROC
1260 :
1270 REM To save machine code, run prog
1280 REM then use *SAVE mcclr1 900 966
```

VENETIAN BLINDS

The second program is the only true all-mode clear of the three presented. It creates a venetian blind effect, with many fine slats which become progressively wider until the whole screen is blanked out. When you run the program, you will be asked for a mode, before the demonstration screen is first created then destroyed. The routine will perform correctly in modes 0-6, though again the demonstration screen created in lines 130-150 will have no effect in modes 3 or 6.

This clear is also the most economical of the three, requiring just 56 bytes of code (decimal). It is also very fast, and I have incorporated a time-wasting loop to prolong the effect somewhat. It is currently set to clear the 20K modes on a model B in about one second - 10K modes take about half the time. If you wish to alter the time delay, simply adjust the value with which X is loaded at the start of line 1120. Currently the routine uses:

```
LDX #10
```

Increasing it (max is 255) slows down the clear, while reducing it (min is 1) gives the fastest. Just for purists, the slowest fill is not in fact given by LDX #255 but by LDX #0 (since 255+1=0 in the X register).

```
  10 REM Program Blinds Clear
  20 REM Version B 0.9
  30 REM Author  Barry Christie
  40 REM Beebug  Aug/Sept 1987
  50 REM Program subject to copyright
  60 :
 100 MODE7:PROCassemble
 110 REPEAT:MODE7
 120 INPUT'"Clear which mode (not 7) ",
M:MODE M
 130 GCOL0,135:CLG:GCOL0,4
 140 MOVE 280,200:MOVE 640,824
 150 PLOT 85,1000,200:GCOL0,0
 160 A=INKEY(100)
 170 CALL clearscreen
 180 A=INKEY(100):UNTIL FALSE
 190 :
1000 DEF PROCassemble
1010 FOR pass%=0 TO 2 STEP 2
1020 P%=&900
1030 [:OPT pass%
1040 .clearscreen
1050 LDA #&A0:LDX #&50:JSR &FFF4
1060 STX &72:STY &73:LDY #&07
1070 .loop1
1080 LDA &72:STA &70:LDA &73:STA &71
1090 LDA #&00
1100 .loop2
1110 STA (&70),Y
1120 LDX #10:.slow:DEX:BNE slow
1130 DEC &70:DEC &70:DEC &70:DEC &70
1140 DEC &70:DEC &70:DEC &70:DEC &70
1150 BNE loop2:INC &71:BPL loop2
1160 DEY:BPL loop1:RTS
1170 ]:NEXT pass%
1180 ENDPROC
1190 :
1200 REM To save machine code run
1210 REM program, press Escape, then
1220 REM use *SAVE mcclr2 900 938
```

WHIRLPOOL

Listing 3 is somewhat longer than the first two, but this is largely because it contains two separate routines: one to destroy the screen, and another to create a new one. When the program is run, the user is first prompted to enter a mode (0-2). The by now familiar demonstration screen is then generated, only to be simultaneously scrambled and sucked into a whirlpool at the centre of the monitor (gulp!).

This is all fairly familiar stuff, but now lines 200 and 210 generate a new screen consisting of some random fills. Note the CLS at the start of line 200. This is essential since the whirlpool method does not actually clear the screen area of RAM, it just writes to the 6845 video controller chip. The old screen therefore remains in RAM, and must be physically cleared with a CLS before proceeding.

Once the random shapes have been put onto the invisible screen, the routine PROCconstructscreen is called (line 220) to generate the whirlpool effect in reverse. This is very effective, as you can see from the demonstration - more effective than the scrambling on the screen clear, because the new screen contents show off the effect better than the original triangle pattern.

Instructions for saving the routines are again given at the end of the program. But this time there are two call addresses. &900 will effect the clear, and you should use &969 to regenerate the screen. If you are using the clear on its own, you should follow the call to "destructscreen" with a mode change, to reset the 6845 registers to their default values. The program outline below illustrates how you might use the two-part routine in your own programs. It assumes that you have saved the machine code as "mcclr3".

```
100 *LOAD mcclr3
.
.
1000 PROCcreatefirstscreen
1010 CALL &900: REM Destroy screen
1020 PROCcreatesecondscreen
1030 CALL &969: REM Display new screen
.
.
```

All pretty straightforward really! More from me next month.

```
  10 REM Program Whirlpool Clear
  20 REM Version B 0.8
  30 REM Author  Barry Christie
  40 REM Beebug  Aug/Sept 1987
  50 REM Program subject to copyright
  60 :
 100 MODE7
 110 osbyte=&FFF4
 120 PROCassemble
 130 REPEAT:MODE7
 140 INPUT'"Clear which mode (0, 1 or 2
) ",M:MODE M
 150 GCOL0,135:CLG:GCOL0,4
 160 MOVE 280,200:MOVE 640,824
 170 PLOT 85,1000,200:GCOL0,0
 180 A=INKEY(100)
 190 CALL destructscreen
 200 CLS:GCOL 3,7:FOR I%=1 TO 25
 210 PLOT 85,RND(1279),RND(1023):NEXT
 220 CALL constructscreen
 230 A=INKEY(500):UNTIL FALSE
 240 :
1000 DEF PROCassemble
1010 FOR pass%=0 TO 2 STEP 2
1020 P%=&900
1030 [ OPT pass%
1040 .destructscreen
1050 LDA #&10:STA &70
1060 LDA #&20:STA &71:LDA #&22:STA &72
1070 LDA #&50:STA &73:LDA #&62:STA &74
1080 .destruct
1090 LDA #&06:STA &FE00:LDA &71
1100 STA &FE01:LDA #&07:STA &FE00
1110 LDA &72:STA &FE01:LDA #&13
1120 JSR osbyte:LDA #&01:STA &FE00
1130 LDA &73:STA &FE01:LDA #&02
1140 STA &FE00:LDA &74:STA &FE01
1150 DEC &71:DEC &71:DEC &72:DEC &73
1160 DEC &73:DEC &73:DEC &73:DEC &74
1170 DEC &74:DEC &70:PHP:LDX #0
1180 .slow:DEX:BNE slow:PLP
1190 BPL destruct:LDA #&01:STA &FE00
1200 LDA #0:STA &FE01:RTS
1210 :
1220 .constructscreen
1230 LDA #&10:STA &70
1240 LDA #0:STA &71:LDA #&13:STA &72
1250 LDA #&10:STA &73:LDA #&42:STA &74
1260 .construct
1270 LDA #&06:STA &FE00:LDA &71
1280 STA &FE01:LDA #&07:STA &FE00
1290 LDA &72:STA &FE01:LDA #&13
1300 JSR osbyte:LDA #&13:JSR osbyte
1310 LDA #&01:STA &FE00:LDA &73
1320 STA &FE01:LDA #&02:STA &FE00
1330 LDA &74:STA &FE01:INC &71:INC &71
1340 INC &72:INC &73:INC &73:INC &73
1350 INC &73:INC &74:INC &74:DEC &70
1360 BPL construct
1370 RTS
1380 ]:NEXT pass%
1390 ENDPROC
1400 :
1410 REM To save machine code, run
1420 REM program, press Escape, then
1430 REM use *SAVE mcclr3 900 9C6
```

B

A PROFESSIONAL VIEW (continued)

processor, spreadsheet and database in the conventional sense; it is a professional's tool (and hence presumably the name). Certainly it does not offer all the power and flexibility of View, but it does most of what View does, it does several things better or more conveniently, and provides several useful features View doesn't. The main question is who is going to buy it?

If you're into word processing and spreadsheets, take a look at View Professional, if you have the opportunity. It has a lot going for it. If you are in the market for both word processing and spreadsheet applications then it is worth considering. If you already have View and ViewSheet, like all Master owners, then I doubt you have much to gain from using it. But maybe its full potential will be revealed when the promised Archimedes is released. B

THE COMMS SPOT

This month in the Comms Spot, Peter Rochford takes a look at some of the varied problems that have cropped up in the many letters and mailboxes received from members.

Do keep sending in your queries, either to BEEBUG's offices (the address is given inside the back cover of each issue), or contact me direct via Prestel, mailbox number 019996601.

When writing to me, do make sure you mention that the letter/mailbox is intended for the Comms Spot. I should add that I cannot possibly respond with individual answers to everyone, so please do not feel offended or ignored if you don't get a personal reply. I will endeavour to answer through this column by tackling some of the most common problems.

Apart from your own special queries, please feel free to send your general comments on any aspect of this column. If there is a particular area of comms you would like to see covered or, if there is any equipment you would like to see reviewed, do let me know. Comms means communication - so get writing!

MANUALS

A number of members have complained about poorly written manuals that do not cater well for the newcomer to the world of comms. Many of the manuals I have come across assume too much of the reader, or are not written in a clear manner. I can only help by mentioning it here, and asking manufacturers and software houses to take note. Also, I will take very careful consideration of this in any future reviews and report accordingly. You can help by complaining to the companies concerned if the product you buy is not sufficiently well documented.

OFFLINE MAILBOX EDITING

A very popular problem concerns the use of mailboxing on Prestel. A number of letters have asked about offline editing of mailbox messages. There are offline editors available as part of several comms packages. BEEBUG's own Command ROM has this facility as does Soft Mac's Commsoft. Both of these software packages will function with the majority of modems.

The way to achieve offline mailbox creation is by first going online to Prestel and saving to disc the particular (empty) mailbox frame, using the frame save facility in your comms software. Make sure that you mark the beginning of where the text starts on the frame before saving it. This is already done on some mailbox frames (with a hash). The end of the text field is also marked with a hash sign.

Now go offline, load the saved frame into the editor in your comms software and fill in the text from the marked point up to the hash sign. Do remember, that if you include any colour codes or other display attribute codes, these take up TWO character positions in the frame, although on your display they only show up as one. You must leave one character position unused within the frame for each code, to accomodate these, otherwise when you come to transmit the frame, not all of your text will be sent.

Once you have composed your frame, save it to disc. Now go online, select the same mailbox frame you saved before and used in the editor, fill in the recipients mailbox number and then press hash. The cursor should now be at the start of the text field. Using your software's load frame facility, load in your prepared mailbox. Now use the frame send function to send the frame.

It may be wise before pressing '1 TO SEND', to ask Prestel to send the frame back to you so that you can check that it has not been corrupted by line noise. *00 will do this. Some comms software may define a function key for this purpose (Copy key in Commsoft for example).

MAILBOXES WITH WORDWISE

The question of editing mailbox frames by using Wordwise has also arisen. I believe Peter Gaunt's 'Weirdbeard' viewdata software can cope with this by taking a standard Wordwise file and chopping it into appropriate lengths and formatting it ready for transmission as mailbox frames using the Weirdbeard software. I haven't used this package myself, so cannot comment on how good it is. You can contact the author, Peter Gaunt, via mailbox on Prestel, number 014511332.

USER-TO-USER FILE TRANSFER

This is a thorny subject to tackle. Many who have written in on this need to know from scratch how to go about sending data via the telephone line. I think that in the near future I will devote a whole Comms Spot to the subject of file transfer and the various protocols etc. Most of the practical problems with file transfer stem from people's fear of not being able to go from voice to data transmission and back, without losing the communications link. I have spent many hours in file transfer with few problems in that respect. As long as you get yourself well-organised before you start, and memorise or write down the various steps, you should have no problems. The main rule is: Don't panic!

Still on the subject of file transfer. One member has asked if he can practice file transfer offline with a friend by connecting their modems back-to-back. The answer is yes, you probably can do that. But, if you want to send data from one computer to another to help familiarise yourself with the software for file transfer, you don't need a modem. Just connect the RS423 on your Beeb to the serial port on his machine! You will need to make up a lead to do this, and the Beeb's User Guide has the pinout diagram for its RS423 port for your end of things. The cable should be wired up so that 'Transmit' at one end is wired to 'Receive'at the other end, and vice versa. Likewise connect CTS to RTS, though this is not essential. Then just use your comms software to communicate.

BULLETIN BOARDS

Finally, there have been many times when I have tried to log-on to a bulletin board and after getting the carrier, sat staring at a blank screen for some time. No amount of bashing the Return key would 'wake up' the host system. The immediate reaction was to suspect that I had configured my software wrong or my computer had crashed.

For those who have similar experiences, let me tell you that it is probably more likely that the system which you are trying to communicate with has a fault. Do re-check your comms software and modem is setup OK, however, but then log-on to another bulletin board. Chances are that this time you will be successful. Many BBs run unattended for days and can develop a fault without the sysop knowing. Don't be too quick to blame yourself!

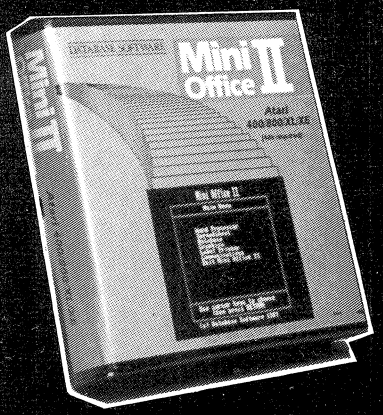
B

PAINT AND DRAW (continued)

```
 3430 :
 3440 DEF PROCfill1(x,y,inc)
 3450 newy=y
 3460 REPEAT
 3470 PLOT77,x,newy:newy=newy+inc
 3480 UNTIL POINT(x,newy)<>background
 3490 ENDPROC
 3500 :
 3510 VDU4:VDU28,0,4,39,0:CLS
 3520 IF ERR<>17 THEN REPORT:PRINT" at "
;ERL:PRINT"Press any key to continue";:a
%=GET ELSE PRINT"Do you really want to e
xit (Y/N)";:a%=GET AND 95
 3530 IF a%<>89 THEN CLS:VDU26:PROCmc:PR
OCscreen:GOTO150
 3540 VDU26:CLS:*FX4,0
 3550 *FX12,0
 3560 END
 3570 :
 3580 DEF PROCmc
 3590 MOVE935,posit%
 3600 GCOL0,0:PRINTCHR$246
 3610 IF colour%=0 THEN posit%=4 ELSE po
sit%=colour%
 3620 posit%=1020-(posit%*32)
 3630 MOVE935,posit%:GCOL0,3
 3640 PRINT CHR$246:GCOL0,colour%
 3650 ENDPROC
 3660 :
 3670 DEF PROCoscli($os)
 3680 X%=os:Y%=os DIV 256:CALL&FFF7
 3690 ENDPROC
```

B

Mini Office II



6 powerful home and business programs in just one package – at a price that simply can't be matched!

- WORD PROCESSOR: Compose a letter, set the print-out options using embedded commands or menus, use the mail merge facility to produce personalised circulars – and more!
- DATABASE: Build up a versatile card index, use the flexible print out routine, do powerful multi-field sorting, perform all arithmetic functions, link with the word processor – and more!
- LABEL PRINTER: Design the layout of a label with the easy-to-use editor, select label size and sheet format, read in database files, print out in any quantity – and more!
- SPREADSHEET: Prepare budgets or tables, total columns or rows with ease, copy formulae absolutely or relatively, view in either 40 or 80 column modes, recalculate automatically – and more!
- GRAPHICS: Enter data directly or load data from the spreadsheet, produce pie charts, display bar charts side by side or stacked, overlay line graphs – and more!
- COMMS MODULE: Using a modem you can access services such as MicroLink and book rail or theatre tickets, send electronic mail, telex and telemessages in a flash – and more!

Mini Office II is a comprehensive integrated suite of programs setting a new standard in home and business software.

Most of the wide range of features – many of which are usually only available on software costing hundreds of pounds – are easily accessed by using cursor keys to move up and down a list of options and pressing Return to select.

Mini Office II takes over where the original, highly successful Mini Office left off, with numerous extra features, two additional modules, a program to convert existing Mini Office files to Mini Office II format, and a 60 page, easy to follow manual.

And it's at a price everyone can afford!

B, B+	Tape £14.95
	5.25" disc £16.95
Master	5.25" disc £19.95
Master Compact	3.5" disc £21.95
B, B+, Master, Master Compact	4 by 32k Rom board £59.95

INSTANT Mini Office II

Now available on a power-packed rom board!

- Four 32k roms on one board packed with 128k of super fast machine code.
- Easy to install – no soldering needed.
- Split second application selection.
- Frees valuable ram space for much more data.
- Fully compatible with the whole BBC Micro range, Aries or Watford shadow ram board, and tape, DFS, ADFS and ECONET filing systems.

The full selection of Mini Office II is available through BEEBUG Retail at MEMBER'S DISCOUNT. Please refer to your BEEBUG Retail Catalogue

DATABASE SOFTWARE

Europa House, 68 Chester Road
Hazel Grove, Stockport SK7 5N

BEEBUG MAGAZINE OFFERS

BEEBUG FILER

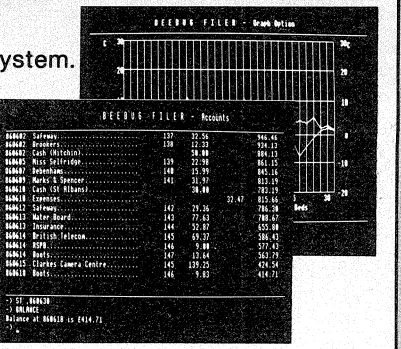
BEEBUG magazine's own database management system.

Suite of five programs including:

- ★ Top-level Menu Selector
- ★ Database Management including Mail-Merge
- ★ Graphics Option
- ★ Home Banking Option
- ★ Fast Sort Program

Supplied on dual 40/80 track disc — Instructions and program notes included.

Price £5.50 plus £1.00 post and packing.



BEEBUG Sideways RAM Module — see Vol.5 No.7

The cheapest and easiest way of adding sideways RAM to your Beeb

- ★ Plugs directly into any BBC ROM socket
- ★ Construction and software described in magazine

Available as a kit of parts for you to make up yourself, or ready made

Price £8.95 (kit) **£12.95** (ready made)

Post & packing £1.00 in each case

BEEBUG Master Alarm ROM

- ★ Automatic clock and date alarm for the Master 128
- ★ Available in ROM for immediate use

Supplied on EPROM

Price £6.45 plus £1.00 post & packing

ORDER FORM

Name.....

Address .....

.....

.....

Membership Number.....

Signature .....

I enclose cheque for £_____

Please debit my Access/Visa No.

Please send:	Price
Filer Disc	£.....
BEEBUG Sideways RAM Kit	£.....
BEEBUG Sideways RAM Module	£.....
BEEBUG Master Alarm ROM	£.....
Post & Packing	£.....
Total	£.....

Send to: BEEBUG Mail Order, Dolphin Place, Holywell Hill, St. Albans, Herts AL1 1EX
Telephone Orders welcome. Tel: (0727) 40303

UK BULLETIN BOARDS

All the above listed boards were believed to be active as of late July 1987. Circumstances may have changed since then and BEEBUG would appreciate members letting us know of any changes they have discovered, such as boards that have ceased operating or changed times and baud rates. Also, if you know of a board that is not listed here, please let us know for the benefit of other members.

Some new boards, particularly NBBS and OBBS systems, enable Beeb owners with Commstar or the Demon Zromm to have colour. With Commstar use mode 7 and switch off the filter mask at the command screen before entering 'Chat' mode. Zromm users should use Mode 7 and *Chat instead of *Terminal. Then when logged on answer 'Yes' to the question 'Are you using software on a BBC that allows colour'.

Aberdeen ITEC 24 Hours	0224 641585 (Aberdeen) 1200/75 Viewdata
Acorn 24 Hours	0223 243642 (Cambridge) 1200/75 Viewdata
Asylum 24 Hours	01 853 3965 (London) 300/300
BABBS I 24 Hours	0394 276306 (Felixstowe) 300/300
BABBS II 24 Hours	0268 778956 (Basildon) 300/300
Basildon ITEC 24 Hours	0268 22177 (Basildon) 1200/75 Viewdata
Betelgeuse 5 24 Hours	0463 231339 300/300 & 1200/75
Bloxham NOBB 22.00-01.00	0295 720812 (Banbury) 300/300
Bolton BBS 20.00-08.00	0204 43082 (Bolton) 300/300
Brixton ITEC 24 Hours	01 735 6153 1200/75 Viewdata
C-View Rochford 24 Hours	0702 546373 (Essex) 1200/75 Viewdata
Cardiff ITEC 24 Hours	0222 464725 (Cardiff) 1200/75
CBABBS 24 Hours (Ex.Thurs.)	021 430 3761 (Birmingham) 300/300
CBBS London NW 24 Hours	0895 420164 (London) 300/300
CBBS Surrey 24 Hours	04862 25174 (Surrey) 300/300
CBBS South West 24 Hours	0392 53116 (Exeter) 1200/75 & 300/300
CFC#9 24 Hours	0395 272611 (Exmouth) 300/300
CNOL 24 Hours	0524 60399 (Lancaster) 300/300

→ 38

EVENTS

PCW Show
Olympia, London
23-27th September 1987

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

Big Ben Club 5th Annual Show
Harderwijk, Netherlands.
10th October 1987

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

Business Ads

← 36

Communitel 24 Hours	01 968 7402 (London) 1200/75 Viewdata
Communitree 21.00-08.00	0874 711147 (Wales) 300/300
Compost Heap 24 Hours	0622 46036 (Maidstone) 300/300 1200/75
Cyberzone 24 Hours	01 638 2034 (London) 300/300
Cymrutel 24 Hours	0492 49194 (Colwyn Bay) 1200/75 Viewdata
Dark Crystal Fido 24 Hours	01 207 2989 (London) 1200/75 & 300/300
Distel 24 Hours	01 679 1888 (London) 300/300
Distel 24 Hours	01 679 6183 (London) 300/300 & 1200/75
Electronic Tourist Board 24 Hours	0874 730172 (Powys) 300/300 1200/75
FBBS Jersey 18.00-06.00	0534 39389 (Jersey) 300/300
FBBS Swansea 24 Hours	0792 203953 (Swansea) 300/300
Forum 80 Hull 19.30-23.30	0482 859169 (Hull) 300/300
Gnome at Home 24 Hours	01 888 8894 (London) 1200/75 Viewdata
Gosport Apricot 18.00-00.00	0705 524805 (Gosport) 300/300
Hackney BBS 24 Hours	01 985 3322 (London) 1200/75 Viewdata
Hamnet 24 Hours Sat-Sun 18.00-8.00 Mon-Fri	0482 465150 (Hull) 300/300
Haunting Thunder 21.00-08.00	0752 364059 (Plymouth) 300/300
Health Data 24 Hours	01 986 4360 (London) 1200/75 Viewdata
ITCU Exchange & Mart 24 Hours	01 960 4742 (London) 1200/75 Viewdata
Keyboard 20.00-00.00	0908 668398 (Milton Keynes) 300/300
LABBS 24 Hours	01 373 6337 (London) 300/300
Livingstone BBS 24 Hours	0506 38526 300/300
London W Tech Centre 24 Hours	0895 52685 300/300 & 1200/75
London Underground 24 Hours	01 863 0198 (London) 1200/75 & 300/300
Mailbox 80 24 Hours	051 428 8924 (Liverpool) 300/300
Mailbox 83 20.00-08.30 WE 24hr	0384 635336 (West Midlands) 300/300

To be continued

Personal Ads

Two **Olivetti 40T SS disc drives** twinned together, will work from BEEB or separate PACE mains PSU. Also two other identical disc drives for spares. All HALF PRICE + p&p. Tel: Newbury (0635)62391.

Aries B32 shadow RAM card with manual, as new, £45. Toolkit ROM, £12. Tel: Guernsey (0481)55039.

Disc drive Opus 5401SD, single 40T with own PSU in double case, manual and utilities disc, £50. Brother HR5 printer + mains adaptor, BBC cable, ribbon and paper, £40. Data recorder Sanyo DR101 - unused, £15. All in exellent condition and available following upgrade. Tel: 061-432-8368, Stockport area.

Acorn View Index and Printer Driver Generator cassettes for sale, £5 each. Tel: Mr.Spector, 01-952-7244.

Torch Z80 second processor with twin dual 80T drives and Perfect software, £300 ono. BBC Master ROM sockets, £5. BBC Master Peartree 32k RAM socket £17.50. 44"Epson printer lead, £5. Tel: Dr.Jowitt (0252)22521.

For BBC B: **Solidisk 128k sideways RAM board,** £25, Solidisc double density DFS with DFDC, £15 (both with utility discs and manuals). Hobbit cassette with book and instructions (boxed), £4. Tel: Basildon (0268)413688.

Acorn Z80 second processor, complete all software & manuals, some extras, best offer around £180. Tel: Ware (0920)66059.

PSION Organiser Datapak and Maths pack, never used, cost £12.95 and £29.95, will sell for £5 and £13. Tel: (0272)832983, eves.

Toolkit Plus ROM, £20 +p&p. Misc. software on tape & disc in original packaging. Assorted books on BBC Micro. Tel: 061-881-6111 after 6pm for list and prices.

Cumana 100k disc drive with PSU,£55. Datagem ROM & manual, £30. Spellcheck III ROM & manual, £20. Tel: (0707)50568.

Juki 6100 Daisywheel printer, mint condition, boxed with instructions and BBC connecting cable, £210. Wordprocessor F.O.C. to purchaser. Tel: Southwell (0636)813847.

Opus disc drive 40T single-sided (mains powered) with a 1770 DFS interface. Mint condition, boxed, unwanted gift, £75. Tel: Irvine (0294)52250.

Seikosha G.P.100, exellent condition, recent service, £59 for quick sale. Tel: (0843)32761.

Midwich 40/80 dual disc drive. APTL board with Wordwise Plus, Caretaker & Printmaster. Manual for each item, £100 the lot. Tel: (02404)2361.

WANTED Keyboard and tractor drive for Brother AIR15 printer and HCR external ROM box for BBC Micro B. Tel: J.Hawkins (0793)615636 daytime, (0242)242979 eves.

Accelerator and Help ROMs, boxed as new, £30 for both. Tel: (0903)755412.

Tandy 4-colour graphics plotter with interface cable and manual, Centronics I/F, £45. 35T double sided disc drive (works with 40T discs), £30. Tel: (09363)3300 eves. (Stoke-on-Trent area).

Wordwise Plus with Bruce Smith's handbook, £25. Beebugsoft Toolkit Plus, £10. Original Mini Office on 80 T disc, £5. Prices include p&p. Tel: Alan, Wrexam (0978)759732 eves./weekends.

Silver Reed EXP500 daisywheel, £100. Voltmace 14B joystick, £6. Printer sharer swithes £6 and £18. W.E. NLQ designer, £15. WYSIWYG Plus, £12. W.E.Fileplus, £8. ROMIT, £18. Printwise £12. Tel: Kings Langley (09277)66636 eves.

Author 40 and 80T plus manual, £15. Genealogy, 80T full tree printout, £10. Telebook 40T, £10. Summary writing 40T, £9. Tel: (0258)55495.

Watfford 32k RAM board unused, £40. As new linear graphics A4 plotter, £265. Linscan, £120. Novacad plotter driver, £10. Tel: (0662)3597 betw 7-9 pm.

Watford 128k ROM/RAM board inc. 16k battery backed RAM with read/write protect switches, manual and utils. disc, £75. Watford 32k shadow RAM board with ROM and manual, £45. Solidisc D.F.D.C. with latest DFS and ADFS ROMs inc. manual, £25. ACORN ADFS ROM with manual, £20. Contact Mr.Storey, 116 Eversley Ave, Barnehurst, Kent DA7 6RG.

WANTED ViewSheet ROM for BBC B. Tel: Cheltenham (0242)522030.

Dual 100k (40T/SS) disc drives (one Chinon & one Teac) in separate cases complete with dual interface and power leads, £80. Also Beebugsoft Toolkit Plus ROM for BBC B in origial packing, £20. Tel: Stephen Crump, Chorley (02572)78286.

Instant Mini Office ROM board, almost new, £35. Contact: John Crabtree, 10 Pathfields, Dartmouth, TQ6 9PH.

Double, dual sided 80T Tandon 5 1/4" disc drives (800k) boxed, £100. Will split at £55. Tel: C.M.Day (0249)782365.

HCR external ROM expansion box for BBC B, 32k of battery backed RAM fitted, as new, £60. Tel: (0704)42282, after 5 p.m.

Master/BBC ROM software inc. all manuals: ViewSheet £24, ViewStore £24, ADT £20, ACP £20, MAX £14, Logotron Logo £30. Tel: (0295)65262 or 01-979-4256.

BBC B Series 7, APTL, Watford 32k shadow RAM. Twin 40/80 DSDD drives in Technomatic plinth. ROMs: Wordwise Pus, all View family, ROM-Spell. TAXAN green monitor. Books etc. All mint condition. Offers. Tel: (0234)750050.

Acorn 40T SS 100k disc drive, £50. Cub monitor 1431, £120. APTL board £25. Acorn User barcode reader, £25. Commstar 2, £25. Digimouse, £25. ViewStore ROM, £25. Murom, Sleuth, Graphic Extn ROM, £10 ea. Drawer CAD, £25. Beebcalc, Masterfile, Aviator, Elite, JCB Digger, £5 each. Tel: Boston (0205)68276 after 6 p.m.

Solidisk Fourmeg 256K board with WW Plus and manager ROM, unused, £80. Tel. Cardington (02303)589.

Acorn ADFS, Western Digital 1770 Controller, Opus D.D.O.S, Sony micro cassettes. Tel: Chris Hall on (0721) 21479 with offers.

Disc Drive Shugart, twin 40T single sided drive (2x100k), £80 ono plus carriage. Tel: Peter Sotheran, Cleveland (0642)471662

Master Compact colour system, complete £470. EPSON LX86 printer £186. Both under guarantee. Tel: N'hants (0604)870465.

Master 128, 512k co-processor, TAXAN 620 monitor (all under guarantee). Twin Solidisk D/S 40/80T drives, mouse, data recorder, joystick, GEM and games software, manuals, £950 ono. Tel: (06286)5062 eves.

BBC Master 128 with 40/80T disc drive. Includes original games on tape and disc, joystic and interface, £450 for the lot ono. Tel: Lee (0679)21261 after 7pm.

Acorn Teletext adapter including latest BBC ROM, manuals and aerial lead, £50. Acorn-Seikosha AP100 printer including BBC cable and lots of paper, a good first cheap printer, £50. Tel: Russel, Nottm (0602)619414 betw. 9am and 5pm.

BBC B O.S.1.2. Series 7, Watford double density disc interface and ROM fitted. Complete with dust cover, immaculate condition, £230. Tel: Grays (0375)380369.

Master 128 twin 40/80 Cumana CD 800s disc drive. Zenith amber screen, Master ROM, Interword, Intersheet, System Delta database, Printwise, ADFS Masterfile II, Officemaster, Whiteknight chess, 20 DSDD discs and box. Master manuals and system manuals. Complete system at £800 ono. Tel: Berkhamsted (04427)4321.

BBC B Issue 7, Watford DFS, Solidisk 4 Meg 256k board. ROMs include MegaROM, Basic, Editor, Replay (8271), Iconmaster Manager ROM, all with full original documentation. Also lots of software including Impossible Mission, Repton 3, Airwolf, Scrabble and much more. Cost over £900, will sell for £550 (offered). Tel:(0462)50445 after 5pm.

BBC B with DFS & Watford 48k SW RAM. Acorn User / Micro User / Beebug from issue 1 - July '87. Pace SS 40T disc drive. BBC Microvitec Monitor. ADE Plus ROMs + manual. BBCsoft Monitor ROM + manual. Mini office II ROM board, Advanced User Guide, Advanced Disc User Guide, AMCON DFS ROM, over 50 discs inc. some games, £450. Tel: Tonbridge 351814.

Dual double-sided 40/80T switchable 5.25" disc drive with PSU, £110. Discs 80T: Paintbox II, Spellcheck (Wordwise) & Teletext editor, £5 each. ROMs: Beebugsoft Murom & Toolkit, Computer Concepts Disc Doctor, £8 each. Tel: (0303)42540 eves.

BBC B (OS 1.2), Acorn DFS, Wordwise, Toolkit, Disc Doctor & books and many games, £210. Viglen 40 S/S drive, £50. Watford 32k shadow RAM, £40. Watford ROM expansion board £20. CANON Typestar 5R printer/typeriter, 6 months old + lead, 4 ribbons, £120. Or everything for £420 + mono TV and either InterWord or Wordwise +. Tel: Leeds (0532)651614.

Penman 3-pen robot plotter/turtle, £150 ono. Grafpad digitiser tablet, £50 ono. Ultracalc II ROM £20 ono, all mint condition with original instruction manuals. Tel: 01-579 6646 eves. (answerfone during day).

For BBC B: Viglen PC console, £15. Aries B12 ROM board with 16k SWR, £15. 2 Viglen 40/80 double sided disc drives, £60 each. External PSU for disc drives £10. Mini Office II, 80T + manual, £8. Spellcheck II ROM + 80T disc + manual, £12. AMX Desk, disc + manual, £5. Plus more software. Tel: Sandiway, Cheshire (0606)888874.

BBC B hardware/software, all as new and in original packing. 6502 second processor + DNFS, Hi-basic and Hi-Wordwise + disc, £95. Separate keyboard and procesor case, 25. WE double density DFS kit, £40. Viewsheet ROM £25, Wordease ROM £12, Spellcheck II ROM + 80T disc £15. WE diagnostics disc £12. GDump screen dump ROM, Computer Concepts Graphics ROM, WE single density DFS, £10 each. Tel: Adrian 01-992-4935.

THE MASTER PAGES

Devoted to the Master Series Computers

We kick off this month's Master Pages with an interrupt driven clock by Mark Lock. This creates a continuously updated clock and calendar display at the top of the screen in all modes, while leaving the computer free to carry out other tasks.

If you have had problems using *COMPACT on the ADFS, then you will probably find C. Middlehurst's intelligent compaction routine of great benefit.

For all EXEC file users we also have a short article on using library directories - how to set them up, and how to minimise their volatility: plus of course a page of hints and tips for Master and Compact users.

And do keep the contributions trickling in!

MASTER SERIES

Interrupt Driven Clock

Want to keep track of the time? This utility by Mark Lock continuously displays the date and time while also allowing the computer simultaneously to pursue other tasks.

This interrupt driven digital clock works in all modes (including shadow modes), continuously displaying the time and date on the top line of the screen. In non-teletext modes the display is reversed out for clarity.

The clock, which may be used from within other programs, or in a stand-alone manner, is initiated by calling its start address (&DD00 as supplied), and switched off by calling start-address-plus-three (i.e. &DD03).

GETTING STARTED

To get the program up and running, first type in the listing, and save it away before running it. When the program is run, you will be prompted to save the assembled machine code. To run the program, either use:

```
CALL &DD00
```

if the assembled code is already in your machine, or run it directly from disc with:

```
*MCCLOCK  (*RUN MCCLOCK from cassette)
```

You should now see a time and date display on the top line of the screen (same format as *TIME), which should be undisturbed by scrolling or even clearing the screen. The routines used are all thoroughly legal, which means that the clock is not poked to the screen, but written using OSWRCH.

One consequence of this is that it will not generally function within proprietary software packages such as the Master Editor or Wordwise. There should however be no problem if you want to incorporate the routine into your own Basic programs.

NOTES

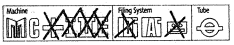
The routine currently uses the Master's Transient Program Area at &DD00, which is unused by the MOS. But you may relocate it if you wish by altering line 110.

If you want to run the clock from sideways RAM, you will need to relocate the temporary storage area set up in lines 680 to 700, as well as redirecting the vectors into RAM.

```
 10 REM Program Interrupt Clock
 20 REM Version B 0.7(B)
 30 REM Author  Mark Lock
 40 REM Beebug  Aug/Sept 1987
 50 REM Program subject to copyright
 60 :
100 MODE 135
110 code=&DD00
120 nvwrch=&FFCB:osword=&FFF1
130 osbyte=&FFF4:oscli=&FFF7
140 tab=31
150 rclock=&E:rcform=&0
160 mode=&355
170 :
180 FOR pass%=0 TO 3 STEP3
190 P%=code
200 [OPT pass%
210 .ton JMP set
220 .toff JMP setoff
230 .get LDA#rcform:sta parblk
240 LDX#parblk MOD 256
250 LDY#parblk DIV 256:LDA#rclock
260 JSR osword
270 RTS
280 :
290 .out:LDA#tab:JSR outch:LDA#0
300 JSR outch:JSR outch
310 LDA#17:JSRoutch:LDA#128+7
320 JSRoutch:LDA#17:JSRoutch:LDA#0
330 JSR outch:LDX#0:BRA outg
340 .outl INX:JSRoutch
350 .outg LDA parblk,x:cmp#&D
360 BNEoutl:LDA#17:JSRoutch:LDA#128
370 JSRoutch:LDA#17:JSRoutch:LDA#7
380 JSRoutch:LDY mode:LDA widthtab,Y
390 PHX:TSX:CLC:SBC &101,x:PLX
400 TAX:INX:LDA #&20
410 .spclp JSR outch:DEX:BNE spclp
420 RTS
430 :
440 .ptime PHA:PHX:PHY:JSRfx
450 JSR settime:CPY#0:BNEnim:INY
460 .nim PHY:PHX:JSR get:JSR out
470 LDA#tab:JSRoutch:PLA:JSRoutch
480 PLA:JSRoutch:JSRfxclose
490 PLY:PLX:PLA:RTS
500 :
510 .set JSRpvec:LDX#ptime MOD 256
520 LDY#ptime DIV 256:SEI:STX &220
530 STY &221:CLI:BRA set1
540 .clockpars EQUB &9C:EQUB &FF
550 EQUB &FF:EQUB &FF:EQUB &FF
560 .oldpvg JMP oldpvg
570 .set1 LDA#&FF:STA clockpars+4
580 LDA#14:LDX#5:JSR osbyte
590 JMP ptime
600 :
610 .settime LDX#clockpars AND 255
620 LDY#clockpars DIV 256
630 LDA#4:JSRosword:LDA&D0
640 AND#&40:BNEsep:LDX&318
650 LDY&319:RTS:.sep:LDX&364
660 LDY&365:RTS
670 :
680 .parblk EQUB 0
690 EQUS"ddd,nn mmm yyyy.hh:mm:ss"
700 EQUB &D
710 .setoff LDA#13
720 LDX#5:JSRosbyte:LDY#&FFCB DIV256
730 LDX#&FFCB MOD256
740 SEI:STX &20E:STY &20F:CLI
750 RTS
760 :
770 .pvec LDX#scrollp MOD 256
780 LDY#scrollp DIV 256:SEI:STX &20E
790 STY &20F:CLI:RTS
800 :
810 .scrollp
820 PHP:CMP#&A
830 BEQ timedo:CMP#&B
840 BEQ timedo1:CMP#&C
850 BEQ timedo3:PLP:JMP outch
860 .timedo PHA:PHX:PHY:LDA#&A
870 .timedo2:JSR outch
880 JSR ptime:PLY:PLX:PLA:PLP:RTS
890 :
900 .outch JMP nvwrch
910 .timedo1:PHA:PHX:PHY
920 LDA#tab:JSRoutch
930 LDA#0:JSRoutch:JSRoutch
940 LDA#ASC(" "):LDX#0:.timelp
950 JSRoutch:INX:CPX#26:BNE timelp
960 LDA#&B:BRA timedo2
970 :
980 .timedo3:PHA:PHX:PHY
990 LDA#&C:JSRoutch:LDA#31:JSRoutch
1000 LDA#0:JSRoutch:LDA#1:JSR outch
1010 JSR ptime
1020 PLY:PLX:PLA:PLP:RTS
1030 :
1040 .fx
1050 LDA#&EC:LDX#0:LDY#&FF:JSRosbyte
1060 STX&100:LDX#20:LDY#0:JSRosbyte:RTS
1070 .fxclose:LDA#&EC:LDX&100
1080 LDY#0:JSRosbyte:RTS
1090 :
1100 .widthtab EQUB 80:EQUB40
1110 EQUB25:EQUB80:EQUB40
1120 EQUB25:EQUB40:EQUB40
1130 :
1140 ]
1150 NEXT
1160 PRINT'"To save machine code, use:"
1170 PRINT'" *SAVE mcclock ";
1180 PRINT;~code;" ";~P%
1190 END
```

MASTER SERIES

Intelligent Disc Compactor



Use this compaction routine by C. Middlehurst to fully compact your ADFS discs.

If you are an ADFS user you will probably have come across the *COMPACT and *MAP commands. The latter tells you how much of the disc's free space is fragmented. If it is badly fragmented you can use *COMPACT to reorganise the disc space more economically. As some users will have discovered, you may need to issue the *COMPACT command as many as TEN times before compaction is complete. You will also have noticed that the routine writes garbage all over the screen unless you supply parameters with it.

The accompanying routine automates the process, and avoids screen garbage. When you run it, it will intelligently compact your disc, testing after each compaction whether there is any fragmented space still to be recovered. If so it will repeat the process. It tests its own progress by calling *MAP, then checking the screen display (using VPOS) to see how many lines of fragmented space *MAP has displayed. It also incorporates a routine to limit the number of retries when compacting, because sometimes *COMPACT fails to clear a particular space even after many repeated compactions. At present the program allows *COMPACT a maximum of 3 tries (specified by maxrepeats=3) at removing particularly awkward fragments before calling a halt. You can see how successful the compaction has been by the results displayed at the end. Ideally, there should be just one entry in the *MAP table at the centre of the screen.

NOTES

This routine assumes that PAGE is set to the default value of &E00, and makes use of as much RAM as possible for the compaction process. You should therefore be careful not to make the program significantly longer than currently listed here. Note also that compaction may take a number of minutes to complete, involving as it does the reorganisation of the disc storage space on a byte-by-byte basis.

WARNING

ADFS discs are capable of holding a vast amount of data, so make sure that a backup copy of your disc is made before attempting to compact it. During compaction your disc drive may make unfamiliar and repetitive noises. This is not a sign that something has gone wrong.

```
   10 REM Program Intelligent Compactor
   20 REM Version B 0.11(B)
   30 REM Author  C.Middlehurst
   40 REM Beebug  Aug/Sept 1987
   50 REM Program subject to copyright
   60 :
   70 ON ERROR GOTO 340
   80 MODE 135:HIMEM=&1200
   90 oldpos=0:repeats=0:maxrepeats=3
  100 FOR loop=1 TO 2
  110 VDU135,141:PRINTSPC(8)"COMPACTION
ROUTINE"
  120 NEXT
  130 VDU28,10,22,35,4
  140 PRINTTAB(0,16)"Insert disc and"'"p
ress space bar";
  150 REPEAT UNTIL GET=32
  160 CLS:flag=0
  170 *MOUNT
  180 *MAP
  190 IF VPOS=oldpos THEN repeats=repeat
s+1 ELSE repeats=0:oldpos=VPOS
  200 IF flag=1 AND VPOS>2 AND repeats<m
axrepeats THEN 280
  210 IF flag=1 AND VPOS=2 THEN 310
  220 IF flag=0 AND VPOS>2 THEN 250
  230 PRINT''"Fully compacted"'"Press an
y key";
  240 a=GET:GOTO 10
  250 INPUT'"Compact Y/N ",C$
  260 IF C$="Y" OR C$="y" THEN 280
  270 IF C$="N" OR C$="n" THEN 10 ELSE C
LS:GOTO170
  280 OSCLI("COMPACT 12 6D")
  290 flag=1
  300 CLS:GOTO180
  310 PRINT''''"Compaction complete"
  320 a=GET:GOTO 10
  330 :
  340 MODE135:REPORT:PRINT" at line ";ER
L
```

B

MASTER SERIES

Master Hints

Some more useful hints for Master and Compact users compiled by David Graham

HIDING EXEC FILES IN ALL MODES

Colin Pither

In EXEC Files Update (BEEBUG Vol.5 No.8), a method of hiding the contents of EXEC files was given; the point being that the contents of EXEC files unnecessarily clutter the screen as they are executed, confusing legitimate screen output. The method works extremely well in all modes but 7. To achieve blanking in ALL modes, simply place a CLS command followed by a colon at the very beginning of the first line in the EXEC file whose output you wish to see, thus:

```
PRINT"You won't see this"
CLS:INPUT"OK ",ok$
```

One snag with this of course is that in clearing the screen, you obviously lose all current screen contents.

MASTER ROM BULK ARCHIVE

Lee Calcraft

The new Master ROM allows easy copying between DFS and ADFS with its *FCOPY and *FCOPYNQ commands. These commands may be used with wildcards, allowing powerful bulk copy operations. Here are two examples to show what can be done:

```
*FCOPY D:1.A.* @
```

This copies all files from directory A of the DFS disc in drive one to the currently selected ADFS directory. A Y/N prompt is issued before each file is copied across.

```
*FCOPYNQ D:1.*.* @
```

This copies all files from the DFS disc in drive one to the currently selected ADFS directory. The "NQ" at the end of the command signifies that no queries will be issued during copying, so that all files are copied across in a single operation.

FORCING LOWER CASE IN !MENU FILES

David Graham

In the June issue we showed how the concept of !MENU files could be used with various ADFS menu programs including Peter Rochford's ADFS Menu, Alan Webster's Turbo menu and the menu front end in the Master ROM. Here is a line that you may like to add to all !MENU files used to boot into word processors. It forces the computer into lower case by disengaging Caps Lock:

```
*FX202,48
```

Using this method is more effective than setting the *CONFIGURE option on a Master to remove caps lock, since it is generally more useful to power up with caps lock on (both for making selections from menus, and for using Basic), and then disengage caps lock only when required.

DEFINE YOUR OWN SHADE

Dennis Weaver

Master and Compact users have a number of different preset pattern fills at their disposal. The following uses VDU23,2 to redefine ECF pattern 1 to give a credible grey effect in mode 0, and draw a filled rectangle as a demonstration.

```
100MODE 0
110VDU23,2,170,85,170,85,170,85,170,85
120GCOL16,1
130MOVE 400,500:PLOT 101,1000,1000
```

To compare the default fills, clear the VDU23,2 definition with VDU23,11|then repeat lines 120 and 130. To get the full range of the four default fills, use GCOL 16*n,1 where n is in the range 1-4. The following short program demonstrates the four default fills in mode 0 - press any key to change the fill:

```
100 MODE 0
110 VDU23,11|:REM Clear any user ECF
120 REPEAT
130 FOR n=1 TO 4
140 GCOL 16*n,1
150 MOVE 400,500:PLOT 101,1000,1000
160 PRINTTAB(5,5)"ECF Fill Number ";n
170 A=GET:NEXT
180 UNTIL FALSE
```

You can find much more on all this in the Master Reference Manual part 1 E.3-13, 16 and 17, and part 2 L.2-27.

MASTER SERIES

Using Libraries

David Graham looks into ADFS libraries in pursuit of a stable home for EXEC files and machine code utilities.

On a good number of occasions in these columns we have urged the use of EXEC files. This Beeb equivalent of the MS-DOS batch file is both powerful and easy to use, whether it be for complex functions, or the implementation of one-line commands. The obvious place to keep EXEC files is in the library directory of your current ADFS disc, and this is why we are taking a closer look at the use of libraries in the present article.

WHAT IS A LIBRARY DIRECTORY ?

Essentially a library directory is a directory known to the filing system which is searched for unknown star commands. If the command:

```
*NAME
```

is issued, the operating system first offers it to each of the machine's sideways ROMs in turn. If none responds to the call, the currently selected directory of the filing system is searched. If the file is still not found, the library directory of the current filing system is then searched. If at any stage the file is found it will be *EXECed if it is an EXEC file, or *RUN if it is machine code. As you will appreciate, this is a very powerful facility. It means that you can have a library full of EXEC files and machine code utilities, and can call these from anywhere on the disc, regardless of the currently selected directory with the simple command:

```
*filename
```

The Library directory is easily assigned using the command:

```
*LIB LIBNAME
```

where LIBNAME is the name designated by the user for the library directory. This may include a pathname, and even a drive number, thus:

```
*LIB :1.$.UTILITIES.LIBNAME
```

sets the library to the directory LIBNAME, which is a subdirectory of $.UTILITIES on drive one. If you now catalogue the disc you will see the library assignment as part of the header information. It should say:

```
Lib. LIBNAME
```

If the library is unassigned at any time, the display reads:

```
Lib. "Unset"
```

Unfortunately, the library designation is not stored on disc, and as a consequence it is highly volatile. If you change drives using:

```
*MOUNT 1
```

and then return to your original disc, you will find, on issuing a *CAT, that the library is again "Unset". But there are various ways around the problem of volatility.

AVOIDING LIBRARY VOLATILITY

First of all, if you enter the ADFS either on powering up the machine, or with a Ctrl-Break (or a Ctrl-A-Break), the ADFS makes an intelligent stab at trying to assign a library for you. It searches the root directory for a directory beginning with the letters "LIB", and designates this the library directory. If it finds none, it assigns the library to the root directory. If you enter the ADFS with:

```
*ADFS
```

or `*FADFS`

no such search is made, contrary to claims in the Reference Manual. But at least previous library assignments are not lost when these commands are issued. Indeed, contrary to expectation, the currently selected directory is also retained when *ADFS is issued.

By contrast, as we said earlier, the command:

```
*MOUNT
```

automatically clears the library assignment. Quite rightly Acorn regard this as an unfortunate "feature", and have prevented this from happening on the ARM version of the ADFS, though they still do not store the library assignment on the disc itself! Fortunately, you can get around the problem of *MOUNT on Master Series machines by using *DIR. Thus instead of issuing:

```
*MOUNT 1
```

to engage drive one, you can use:

```
*DIR :1  (or *DIR:1)
```

This has an identical effect in engaging drive one, but retains the previously set library directory. If you now issue an unrecognised star command, the previously assigned library directory (on drive zero) will be checked. Moreover, you can get back to drive zero with:

```
*DIR :0
```

again retaining the library setting.

REASSIGNING THE LIBRARY

If the library does become unset for any reason, it can of course always be reset with the command:

```
*LIB $.LIBRARY
```

or whatever. But you can reduce the work involved here by creating an EXEC file containing this string, and storing it in the ROOT directory of your disc under the filename "L". Then if you ever get the message:

```
Bad command
```

after issuing the name of one of the files in your library directory, you can reset the library with the command:

```
*$.L
```

from anywhere on the disc.

Colin Pither, a regular contributor to the Master Hints pages, suggests another solution. It again involves the setting up of an EXEC file, but he leaves the name of the library flexible, and this is prompted for by his EXEC file:

```
CLS:OS.("DIR$"):I.'"Assign which dir
ectory to the Library:",lib$:OS.("LIB
"+lib$)
```

He saves this file under the name "SET" in the root directory of his disc, and calls it with:

```
*$.SET
```

As with the *$.L call above, the specification of the root is necessary because the assumption is that the library is, as yet, unassigned. Another way, of course, is to make sure that every disc which you use contains a !BOOT file to set up the library directory. Then if the library ever becomes unset, pressing Shift-Break will reassign it.

Summary:

1. Keep all EXEC files and machine code utilities in a library directory.
2. Use a directory name beginning with the letters "LIB" for your library.
3. Ensure that the library directory is in the root directory.
4. Use *DIR:0 (or 1) to change discs - never *MOUNT.
5. Keep an EXEC file for reassigning lost libraries.
6. Include a library assignment on the !BOOT file of every disc.

B

6502 DEBUG (continued)

```
350 .write:PHP:PHA
360 ROR A:ROR A:ROR A:ROR A
370 JSR hdloop:PLA:PHA:JSR hdloop
380 PLA:PLP:RTS
390 :
400 .hdloop:AND #&F:CMP #&A
410 BCC aaa:CLC:ADC #7
420 .aaa:ADC #48
430 JSR oswrch:RTS
440 :
450 .bin:STA &102:TXA:PHA
460 LDA #&80:LDX #3
470 .more:DEX:BEQ miss
480 BIT &102:BNE one:PHA:LDA #48
490 .osw:JSR oswrch:JMP cont
500 .one:PHA:LDA #49:JMP osw
510 .cont:PLA:.miss:LSR A:BNE more
520 PLA:TAX
530 .space:LDA #&20:JSR oswrch:RTS
540 :
550 .heading
560 JSR osnewl:LDY #&FF:LDA #&D
570 .rept:JSR oswrch:INY:LDA table,Y
580 BNE rept:JSR osnewl:RTS
590 .table
600 EQUS"Addr   A  X  Y  NVBDIZC"
610 BRK
620 ]
630 NEXT
640 PRINT'"Length of code should be 21
7 bytes"
650 PRINT"Present length is ";P%-start
;" bytes"
660 PRINT'"Copy the following to save
machine code"
670 PRINT'" *SAVE debug ";STR$~start;"
";STR$~P%
```

B

EXPLORING ASSEMBLER

(Part 3)

A series for complete beginners to machine code by Lee Calcraft

This month: Multiple Branches Program Looping and Indexed Addressing

Last month I left you with a piece of unexplained code. The object was to work out its approximate equivalent in Basic.

```
.tester     LDA &70
            BNE notzero
.zero       LDA #ASC("Y")
            JMP write
.notzero    LDA #ASC("N")
.write      JSR oswrch
            RTS
```

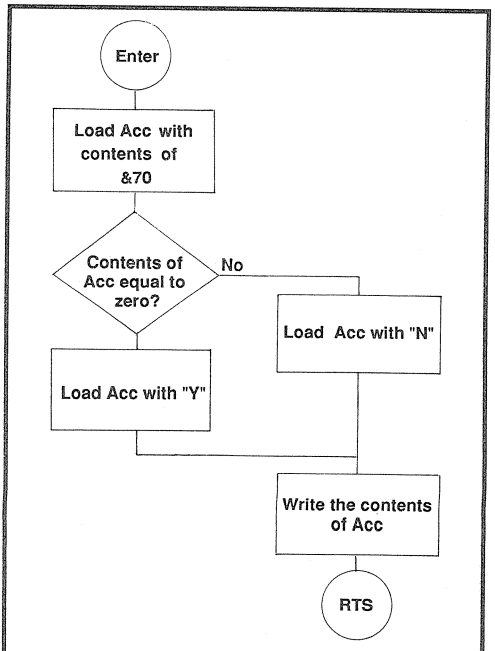


Table 1 Flowchart for zero test routine.

Table 1. Flowchart for zero test routine

Loosely translated, this becomes:

```
IF ?&70=0 THEN PRINT "Y" ELSE PRINT "N"
```

In other words the routine just checks location &70 to see if it contains a zero. If it does not, it branches to "notzero" (BNE notzero) then loads the accumulator with the ASCII code for "N", and jumps to the Beeb's operating system at oswrch which causes the "N" to be printed. If the test for zero is positive, the accumulator is loaded with a "Y" before oswrch is called. Table 1 gives a flow chart of the operation.

MULTIPLE BRANCHES

I want to give one further example of branching before moving on. It is concerned with checking keyboard input:

```
.read   JSR osrdch
        CMP #ASC("Y")
        BEQ yescode     \ Respond to Y
        CMP #ASC("N")
        BEQ nocode      \ Respond to N
        JMP read        \ Loop again
```

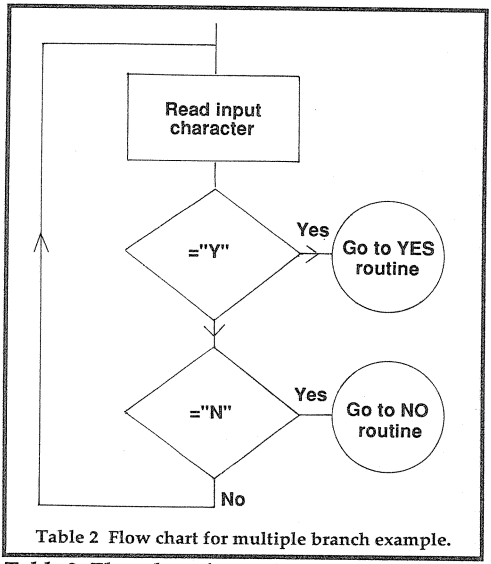


Table 2 Flow chart for multiple branch example.

Table 2. Flowchart for multiple branch example

This routine (see flowchart in Table 2) forms a continuous loop which repeatedly checks keyboard input for a "Y" or "N" response. It uses the operating system call, osrdch (address &FFE0), which reads individual characters

from the keyboard, and returns with the accumulator containing the ASCII value of the key pressed. Even though the operating system call is made just once in every loop, the data with which it returns is tested twice: once for a "Y" and once for an "N".

Both tests use the CMP instruction followed by a conditional branch, and rely on the important fact that the CMP instruction does NOT change the contents of the accumulator. It simply checks the contents against a particular value (ASCII "Y" or ASCII "N" in this case), and sets the flags on the result of the comparison. We can therefore test the accumulator as many times as we wish for different possible key presses without having to return to osrdch between each. If a "Y" is detected, the program branches to the label "yescode", and if an "N" is detected it branches to "nocode", otherwise it loops around and checks the keyboard again. Note the use of the backslash character (which appears as the 1/2 symbol in mode 7). This works just like REM in Basic, except that comments in assembler, once terminated by a colon, may be followed by executable code on the same line.

COUNTED LOOPS

In the above example, the program loop continues until a key is pressed. If we wish to create a loop which is executed a specific number of times, rather like Basic's FOR-NEXT loop, we can use the X or Y register as a loop counter. For example, the following loop will cycle just five times:

```
        LDY #0
.loop   JSR &FFE7
        INY
        CPY #5
        BNE loop
```

At the start of the loop, the Y register is loaded with zero. The JSR &FFE7 instruction is then performed, calling the operating system routine, osnewl, to generate a carriage return and line feed. The instruction INY (INcrease Y) is used to increase the value of the Y register by one. CPY #5 (ComPare Y) then compares the Y register to the value 5, and sets the flags accordingly. If it is not equal to 5 the program loops around (BNE loop), and produces another carriage return and line feed. This continues until Y finally reaches the value 5. The loop then exits, having produced a total of 5 carriage return line feed sequences. The equivalent of this routine in Basic would be:

```
FOR A=1 TO 5:PRINT:NEXT
```

There is a slightly more efficient way of performing this loop using the instruction DEY (DEcrement Y). Consider the following variant:

```
        LDY #5
.loop   JSR osnewl
        DEY
        BNE loop
```

By starting our count at 5, and counting DOWN instead of UP, we avoid the need for the CPY instruction because as soon as DEY causes the value of Y to reach zero, the zero flag is automatically set. This can result in a considerable time saving if the loop is to be performed a large number of times.

INDEXED ADDRESSING

In order to develop the idea of counting loops a little further, we will introduce the concept of INDEXED ADDRESSING. When we described the use of the two instructions LDA and STA at the start of the series, we said that they could be used respectively to load the accumulator with the contents of the memory location given in the operand, and to store the contents of the accumulator at the address given in the operand. Thus:

```
LDA &A00
STA &A01
```

would load the accumulator with the contents of memory location &A00, then store it at location &A01.

But it is not always convenient to specify the memory location in this way. Suppose you wanted to fill each location of a mode 7 screen with a particular character (as an example, character 32 could be used to clear the screen by filling it with spaces). Although you would need just one LDA instruction to load the accumulator with the required character, you would require 1000 individual STA operations to address the full mode 7 screen. Clearly this would be ridiculous, and this is where the 6502's very flexible addressing modes come into their own.The instruction:

```
STA &3000,X
```

employs so-called "indexed addressing", and has the following effect: "store the contents of the accumulator at the address found by adding the contents of the X register to the number &3000". Thus if X=0, the contents of the accumulator will be stored at location &3000, while if X=1, it will be stored at location &3001, and so on.

STRING CHECKING

Indexed addressing can be applied equally well to LDA and STA operations, as the following examples show. The first is a routine which checks the length of a string of characters starting at location &A00. It assumes that the string is terminated with a carriage return character (&0D).

```
        LDY #&FF
.more   INY
        LDA &A00,Y
        CMP #&0D
        BNE more
```

You may be a bit surprised by the first instruction of the routine. This loads the Y register with the value &FF, rather than zero, as one might have expected. The reason for this is that we need to perform the associated INY instruction at the START of the loop rather than at the end. This is because we want to branch on the contents of A, and not on the latest value of Y. The Y register is therefore initially loaded with the value &FF, but this is immediately increased by one to give the value zero. This takes advantage of the fact that all three 6502 user registers A, X and Y, wrap around from &FF to zero, and vice versa.

The rest of the routine is straightforward. The accumulator is loaded with the contents of &A00 (since Y is zero). This is tested for an &0D character, and if one is found, the loop terminates. If none is found, the loop returns to the point "more", Y is incremented, the accumulator is now loaded with the contents of &A01 (&A00+1), and the test performed again.

Listing 1 incorporates this code into the assembler header developed earlier in the series to produce a stand-alone program. When it is run, the user is asked to enter a string, which may be of any length allowed by the operating system. This is placed in memory at location &A00 (using the $ operator in line 320), and the length checking routine is called. Before the routine terminates it stores the contents of the Y register (containing the length of the string) at location &70. This is then read from Basic with PRINT ?&70 at line 340, before the program repeats.

Listing 1

```
100 REM MEASURE LENGTH OF STRING
110 REM Version B 0.4
120 MODE 7
130 FOR pass=0 TO 1
140 P%=&900
150 [
160 OPT pass*3
170 :
180 .start
190 LDY #&FF
200 .more
210 INY
220 LDA &A00,Y
230 CMP #&0D
240 BNE more
250 STY &70
260 RTS
270 ]
280 NEXT
290 :
300 REPEAT
310 INPUT'"Enter String "A$
320 $&A00=A$
330 CALL &900
340 PRINT"Length of string = ";?&70
350 UNTIL FALSE
```

SCREEN FILLING

We can use an essentially similar loop to create a simple screen filling routine capable of filling up to 256 screen locations in sequence. The unscrolled mode 7 screen is located from &7C00 to &7FE7, and the following routine will fill the top 6 lines of it (a 240 byte segment) with whatever character is contained in location &70.

```
        LDA &70
        LDY #240
.loop   STA &7BFF,Y
        DEY
        BNE loop
```

The routine is nice and short, and will execute extremely quickly. As you can see, we have moved the DEY instruction to the end. This is

continued on page 54

BEEBUG WORKSHOP - If....

One of the most flexible and useful of statements in Basic is the IF-THEN-ELSE construction. However, this versatility can pose problems when determining the precise outcome of such a statement. This workshop is aimed at clarifying some misconceptions, and showing how to make the best use of this construction.

The art of programming could be said to be the art of writing programs which are readily understandable, and yet in some instances you will find programmers using such constructions as:

```
IF A>=0 THEN IF
   B(A)>5 THEN
```

when:

```
IF A>=0 AND B(A)>5
   THEN
```

is clearer. We are left wondering if the two statements produce identical effects; if so why use the former construction at all?

In the case above, the two statements are logically equivalent, with the latter being the easier to follow. However, in BBC Basic the latter statement would give an execution error if the value of A were less than 0 (say -1) because B(A) would be B(-1), and negative subscripts are not allowed. In addition, the former construction executes considerably faster, because the second expression is not evaluated unless the first is true. In general when using AND, OR and EOR, each and every expression is evaluated before testing the overall truth of the statement. There are, therefore, occasions when the former statement is preferable, even if it is less readable.

MULTIPLE "ANDS" AND "ORS"

Suppose we wished to design an IF statement to perform the following piece of logic: "A pensioner is a male aged 65 or over, or a female aged 60 or over." Below I present three ways of achieving this, but do they all evaluate correctly?

```
IF sex$="M" AND age>=65 OR sex$=
"F" AND age>=60 THEN PROCpension

IF (sex$="M" AND age>=65) OR
(sex$ ="F" AND age>=60) THEN
PROCpension

IF age>=65 OR sex$="F" AND
age>=60 THEN PROCpension
```

When mixing AND, OR, NOT and EOR in the same statement you should be aware that, just like ^, *, /, + and - there are priorities. The order for the logical operators is highest to lowest: NOT, AND, OR, EOR. Should you wish to alter the precedence use brackets, just as you would in any mathematical expression.

NEGATING IF STATEMENTS

One often sees program segments such as:

```
100 IF A=1 OR A>5 THEN 110 ELSE
    PROCedure
110 rest of program
```

Notice how the program performs PROCedure only if "A=1 OR A>5" is NOT true. When this is encountered in a program listing I suspect that the programmer can't work out how to negate the statement, which could be rewritten as:

```
110 IF A<>1 AND A<=5 THEN
    PROCedure
110 rest of program
```

Negating expressions where no logical operator is present is straightforward. However, I think that most of us find it quite tricky to reverse logical statements where expressions are linked with one or more logical operators. Try reversing this plain English text, for example:

"I will go to the theatre if I have the money and I have nothing else to do, or I am invited by a

friend as a treat".
I won't go to the theatre if ....." (Oh dear!)

Interestingly, the first form of lines 100/110 above could be written as:

```
100 IF A=1 OR A>5 THEN ELSE
    PROCedure
110 rest of program
```

omitting the line number after the THEN, or even missing out the THEN altogether:

```
100 IF A=1 OR A>5 ELSE PROCedure
110 rest of program
```

PROCEDURE TO REVERSE LOGIC

Negating logical expressions with no operators is straightforward:

EXPRESSION	NEGATED EXPRESSION
A=5	A<>5 or NOT(A=5)
A<7	A>=7 or NOT(A<7)
A>=8	A<8 or NOT(A>=8)

It is when a logical operator is involved that it becomes more difficult. The simplest way is to bracket the expression and put NOT in front of it, for example:

```
negating:   A=1 OR A>5
gives:      NOT(A=1 OR A>5)
```

Should this not be to your liking you can apply the rule: negate each operation but change AND to OR, and OR to AND throughout.

```
Thus:       NOT(A=1 OR A>5)
becomes:    NOT(A=1) AND NOT(A>5)
or:         A<>1 AND A<=5
```

Negating expressions with more than one operator should be done in stages.

In BEEBUG Volume 1 we published a three part series on logic, and in Vol.1 No.6 gave a program to evaluate logical expressions. This program will produce a truth table (all true and false combinations) for the supplied expression, and is included on this month's magazine cassette/disc. By using it, you will be able to test the equivalence of say:

```
          A AND (B OR C)
and:  (A AND B) OR (A AND C)
```

Similarly by comparing truth tables, it will allow you to test if an expression is the negative of another.

ELSE

One occasionally finds quite experienced programmers assuming a form of syntax without actually testing it. Maybe some versions of Basic on other machines work differently, but in BBC Basic the following program contains an error (can you spot it?):

```
1000 DEF PROCtime
1010 IF hrs=1200 m$="noon":
     ENDPROC
1020 IF hrs=0 m$="midnight":
     ENDPROC
1030 IF hrs<1200 m$="am" ELSE
     m$="pm":ENDPROC
1040 :
2000 DEFPROCdate
```

The mistake made here is that the programmer probably assumed in line 1030 that m$ would be set to "am" or "pm" accordingly, and WHATEVER the outcome ENDPROC would be performed. This is INCORRECT; ENDPROC is never executed if hrs<1200, and the program 'falls' into the next procedure without ending correctly. In this example ENDPROC is only executed if hrs>=1200.

The rules are that if the expression evaluates to TRUE, all statements before the ELSE are performed and the program then continues to the next line; similarly if the expression is FALSE then all statements after the ELSE are evaluated. The corrected version of line 1030 should be:

```
1030 IF hrs<1200 m$="am" ELSE
     m$="pm"
1035 ENDPROC
```

MULTIPLE IFS, THENS AND ELSES

Using a single statement, the previous "time" example could have been coded thus:

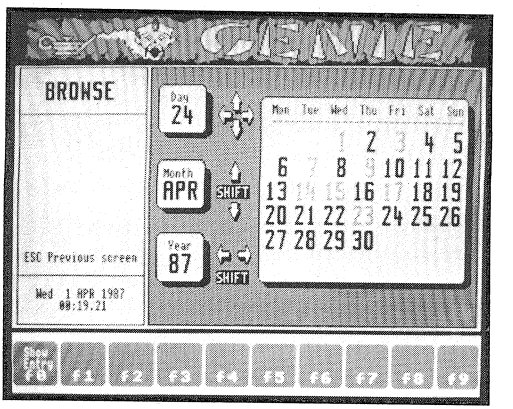
```
1000 DEF PROCtime
1010 IFhrs=0 THEN m$="midnight"
     ELSE IF hrs=1200 THEN
        m$="noon"
     ELSE IF hrs<1200 THEN
        m$="am" ELSE m$="pm"
1020 ENDPROC
```

I personally prefer the statements coded on separate lines, as in the first procedure, because it aids clarity, but that is just personal preference.

Should you wish to use more complex constructions, then you should encounter fewer problems by using:

continued on page 59

YOUR WISH IS MY COMMAND



An integrated desktop package for all BBC micros - that's Genie from Permanent Memory Systems. Dave Somers re-arranges his office to give Genie a workout.

Product	**Genie**
Supplier	**Permanent Memory Systems, 38 Mount Cameron Drive, East Kilbride G74 2ES. Tel. (03552) 32796 (24 hours)**
Price	**£79.35 inc. VAT and p&p.**

Imagine, if you can, that you are sitting at your computer using your wordprocessor and entering your latest masterpiece. The telephone rings, and a friend asks if you can supply him with the telephone number of a mutual friend. Like a good computer user, your telephone list is of course stored on a database. You then have to save your current text to disc, find your database disc, load up the software and read in your telephone database, then find the desired entry. The wordprocessor disc then has to be re-installed, the wordprocessor entered, and the text re-loaded off disc.

The above operation can take many minutes, especially if you can't find your database disc! Now, imagine that at a press of a key (or three) you could instantly enter a database containing all your telephone numbers (and much more besides). A quick search could be carried out, and the required telephone number retrieved and relayed to your friend. At the press of a few more keys you could be back in your wordprocessor exactly where you were before.

SUMMON UP GENIE

Permanent Memory Systems have conjured up Genie to offer just such a facility. Genie consists of a small module which contains the necessary firmware (64K of RAM, 32K of which is battery-backed, and 32K of machine code in ROM), and a length of ribbon cable with a 28 pin header.

Installation is straightforward, with the 28 pin header being plugged into a vacant sideways ROM socket on the main circuit board (for technical reasons, Genie should not be inserted into a ROM socket on an expansion board).The Genie circuit board is contained within a small plastic case measuring 3.5" square by 1" deep, and this can either be left inside the computer, or placed outside.

YOUR WISH IS MY COMMAND

The Genie environment can either be entered by typing *GENIE or by pressing Shift-Ctrl-G. Once this is done, there is a slight pause while the whole of the computer's memory is copied into Genie's 32K on-board RAM. Once you have finished using Genie, the computer's memory is restored to its former state, and the previous application continues running as if nothing had happened.

From the main menu it is possible to enter any one of Genie's six applications, namely the address book, calendar, calculator, desk diary, note pad, and phone book. Throughout, the screen display is divided into four main areas - the Genie logo, the menu area, the work desk,

and the function key strip. The work area conveniently contains details of the cursor control keys.

The menu area shows the title of the current utility, with details of which options are available. The current date and time are constantly displayed at the bottom of this area. When used on a Master 128 this is initialised from the in-built real-time clock, otherwise it will have to be entered manually when Genie is first selected after power-up. The work desk occupies most of the screen, and the utilities run in this area.

The function key area at the bottom of the screen is used to display the options (if any) that the function keys evoke in the current application. This saves the need for those annoying function key strips that are all too easily misplaced.

THE ADDRESS AND PHONE BOOK

The most used facility on Genie will probably be the address book, which is accessed from the main menu. The work area is filled with a 'card' type display divided up into 5 fields: Name (24 characters), First Name (14), Address (6 lines of 30), Phone Number (15),and any other details (2 lines of 30 characters).

There are options to browse, work (i.e. edit), find an entry (with the use of wildcards for ease of operation), and to print out an entry (to produce an address label, for example).

Editing throughout Genie is consistent. The cursor keys are used to move the cursor around the screen allowing text to be entered at the desired place. Text entry is always in 'insert' mode. The function keys are used for ancillary functions, Delete-character-right uses the familiar Copy or Ctrl-A keys (as with Wordwise/ Interword). Once an address has been entered, it will be stored alphabetically in Genie's 32K of battery-backed memory.

The phone book option causes a list of the addresses, names and phone numbers to be displayed, and these can be scrolled to find the desired entry.

THE DESK DIARY AND CALENDAR

Whenever the diary is selected, any entry for that day will automatically be displayed on the screen. Like the address book, you may browse, search, and edit entries in the diary. An entry for a specific date may be chosen in browse mode, and this can then be edited by selecting the work option. Entries are in a free format display of 10 lines of 34 characters, and word-wrap is automatic.

The calendar can be entered either from the main menu, or from within the desk diary. When entered from the diary, any dates with a corresponding diary entry are highlighted.

THE NOTEPAD

The notepad facility offers the ability to store up to 99 free format entries of 16 lines of 40 characters. There are, in common with the other modules, facilities for browsing, editing, and searching through the entries.

THE CALCULATOR

The calculator operates like a normal desktop model, and offers simple arith- metic operations. It has 10 digit entry, and the usual memory and percentage facilities. From the calculator menu it is also possible to 'pull up' a units conversion table detailing metric to imperial conversion factors, and an ASCII table (should you ever have the desire for one).

THE OPTIONS MENU

The options menu is accessed from the main menu and is used to alter various Genie parameters. The system date and time can be entered from here (unless using a Master 128), and the date display is entered in a similar fashion as for the calendar. The paper and ink colours for the main display can be altered to suit your personal preferences, although there is no way to alter the colours of the Genie logo or function key area.

GENIE STATUS

At any time you can discover how memory is being used. The status option will show how many entries there are in each of the notebook, desk diary, and address books, and how much memory each of these occupies.

SECURITY

Like all databases, there may be information of a sensitive or confidential nature that you would not wish to be divulged to anyone else. For this reason, Genie can be protected by a password of up to 10 characters in length. Once 'locked' in this way the correct password has to be entered before any of the modules, and hence your data can be accessed. If you should forget your password, then there is a means by which PMS can decode this for you if you can establish yourself as the true owner!

The entire contents of the battery-backed RAM contained within Genie can also be saved to disc as a backup to safeguard against failure. The files are of course in an encoded format.

CONCLUSIONS

Genie is both extremely versatile and very useful, and should prove helpful to most users, in a manner comparable to the popular 'SideKick' utility for the IBM PC and its clones.

As a rule I am very wary of most battery-backed devices. After all, what's the use of having battery-backup if it only lasts for a few months and then fails? However, Genie is guaranteed for five years use, so it should not cause any problems in that area.

PMS are planing to bring out a series of disc based utilities for use with Genie. These will include the ability to import and export data with other databases such as ViewStore, Masterfile, and other such popular databases.

PMS are also planning to produce an Electron compatible version. Apart from the split mode operation it will in all essence be just like the BBC/Master version.

Genie is indeed a most useful piece of firmware, albeit at quite a high price, that will readily become an indispensable aid for many BBC users.

B

EXPLORING ASSEMBLER. Part 3 (continued)

necessary because we are once again looping on the value of Y, whereas in the previous example we were looping on the value held in the accumulator. Decrementing Y at the end, in that case, would have interfered with the test on the accumulator contents.

This has a further consequence. We cannot use &7C00 as the base address, since if you look at the loop, you will see that it is never executed when Y is zero. So to make sure that screen location &7C00 is covered, we need to lower the base screen address in the STA instruction by 1 to &7BFF.

The routine has been incorporated into the assembler header, and is given in Listing 2. When you run it, it will ask for the character number to be used, and you should enter any number between 32 and 255. You should be impressed by the speed of the fill - the 240 character block appears instantaneously, and far faster than any Basic routine could achieve.

In the next issue we will implement a block move routine, and will extend the present routine to fill areas of any size.

Listing 2

```
  100 REM 240 BYTE MODE 7 FILL
  110 REM Version B 0.4
  120 MODE 7
  130 FOR pass=0 TO 1
  140 P%=&900
  150 [
  160 OPT pass*3
  170 :
  180 .start
  190 LDA &70
  200 LDY #240
  210 .loop
  220 STA &7BFF,Y
  230 DEY
  240 BNE loop
  250 RTS
  260 ]
  270 NEXT
  280 :
  290 REPEAT
  300 INPUTTAB(24,15)SPC16;TAB(5,15)"Ent
er Character no "A
  310 ?&70=A
  320 CALL &900
  330 UNTIL FALSE
```

B

TIME SERIES ANALYSIS

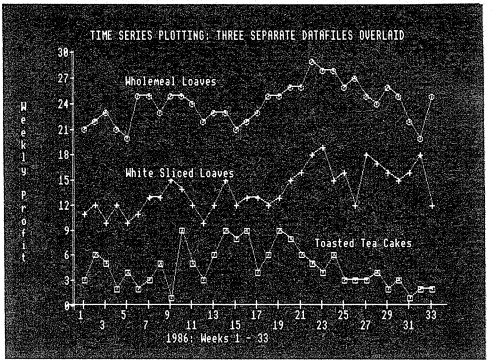
We all know that computers are useful tools for analysing and displaying data, but of course you need the right software. In the first of two related articles C.R.Woodings describes a program to analyse and display time series.

The program accompanying this article is the first of two that we are publishing for the statistical analysis of numeric data. This month we deal with the analysis of time series, that is data measured at regular intervals of time. In a future article we will be dealing with regression analysis and the correlation between two sets of data.

One problem with any program for processing data is the means by which that data may be entered and edited in the first instance. BEEBUG Vol.5 No.10 described how Acorn's ViewSheet could be used as a general purpose data entry facility through the use of a decoder program. This enables you to transfer data from ViewSheet into data files on disc so that these can be accessed by other programs. This month's program, Series, is such a program. It uses these data files and allows you to draw three types of time-series graph both on the screen and, if you have a suitable screen dump, to a printer as well.

For those who do not have access to ViewSheet, or wish to write their own data entry program, details of the data file format used by the Series program are given later.

The first type of graph produced by the Series program is a straightforward sequence plot, where the actual results are plotted on the Y-axis, and the time intervals are plotted along the X-axis. This routine includes the option to overlay other graphs on the same axes, so that you could, for instance, plot several different series on the same screen or piece of paper.



The second type is a series plot which uses an exponential smoothing equation to "iron out" the noise in a graph which might otherwise be very jagged. The routine enables you to select a smoothing factor to give just enough cleaning up of the data to enable the main trends to be easily seen. As with the sequence plot, it allows the drawing of several lines on the same axes.

The final type, the cumulative sum plot ('cusum'), is one of the most powerful ways of highlighting the points in a series when slight changes occur in the data. If you are not used to using this technique, the interpretation of the graph can initially present difficulties.

When the graph is running downhill, it indicates that there is a group of results below the average value for the data. Similarly, an uphill graph indicates a run of results above the grand average. Slight changes from one side of the average to the other cause sharp points which highlight the change in level much better than in a straightforward series. An upward sloping straight line on a sequence plot will give a 'U' shaped parabolic curve on the cusum, indicating a smooth transfer from below to above average data. The line always starts and finishes on the X-axis because the procedure is plotting only the accumulated deviations of each data point from the grand average.

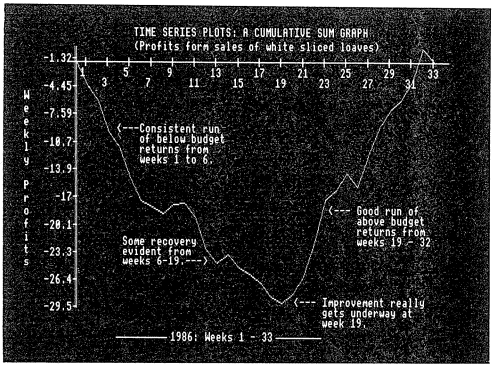
The X-axis itself is drawn at zero on the Y-axis, and the Y-axis units are accumulated deviations, and not the units of your original data. The zero point on the Y-axis will be

chosen to ensure that the curve is fully displayed, and will vary in position between top and bottom of the screen according to the trends in the data.

For a full discussion of the cusum technique, and for those interested in going more deeply into the subject, consult Statistical Methods in Research and Production by Davies and Goldsmith, Oliver and Boyd, 1972.

USING THE PROGRAM

First type in the program and save away to disc (or tape). Before loading or chaining the program, PAGE must be set to &1300 or below to provide sufficient memory. When you run the program, the menu screen appears, the "No data" heading (PROCtop) indicating that you need to load a data file. Select option 1, and the screen shows whatever files you have in the 'D' directory (the program expects any data files to be in directory 'D', and ADFS users will need to create such a directory first within the '$' directory). Enter the name of the file required. You need not enter the directory letter, but the program will not object if you choose to do so. The menu then reappears (the data having been processed by PROCanalysis), and you should now select one of the plotting options (2 would be a good one to start with) to get a quick preview of the graph layout.



Scaling is automatic (PROCaxes and PROClabel) and the screen immediately fills with the plot (PROCdrawline). There is however no title, the axis labels are the default values set in lines 140 and 150, and the Y-axis scaling leaves the lowest data point on the X-axis. Before sorting this out, return to the menu by pressing Escape and try the other two plot types (options 3 and 4). If you have entered particularly "noisy" data, try various smoothing constants (PROCsmooth) to see how the higher values progressively dampen the oscillations and just leave the real trends visible.

When you have seen enough of the automatically plotted graphs, select option 5 (PROCparameters) from the menu and enter the title and axis labels you really want. You can also choose a new Y-axis scale to tidy up the axis labels, and position your plot more attractively on the graph. For instance, if your smallest value (and therefore the Y-axis origin) is 7.6 and your largest 41.3, you could reset the Y-axis start to zero and the Y-axis top to 50. The last set-up question asks whether or not you want the points joined up. If you have fewer than about 30 data points it is generally best to have them joined.

After entering your choice the menu reappears, and you can now re-select the plot option you want and see the reformatted graph.
This time, instead of pressing Escape to get back to the menu, use the cursor keys (PROCnotes). The cursor moves over the screen, and any letters which you now type are added to the display at the cursor position (see illustrations). Be careful not to overtype the graph lines or labels. Whilst the Delete key will erase as normal, the graphics data will not be replaced. When you have finished, do not press Escape unless you want to lose all the on-screen typing. Press the Copy key instead, and the whole screen will be saved to disc under the name SCREEN (line 2320).

If you wish to plot several series graphs on the same screen (PROCoverlay), you can do this by saving one plot first. This can then be reloaded, and a new graph plotted on the same screen. It is advisable to do the overlaying after entering the titles and labels, but before writing any on-screen information, because the additional graph(s) could overwrite your typing. It is obviously important to have selected a Y-axis

scale which will cope with the lowest and highest data points to be plotted in all the series you wish to have on the screen. The overlay option is available immediately after saving a graph. Overlaying can be done indefinitely, but in practice the screen will get untidy with more than about 4 sets of data. The points are plotted with characters starting with ASCII 42 (p%=42 in line 110), and they get less ideal the more overlays you do.

You can improve the program by re-defining some characters to give more conventional graph points if you wish. Remember that you can only overlay series on series and smoothed on smoothed. You cannot mix graphs on the same axes or overlay cusums.

After the overlay option, the program gives you the opportunity to output the current graph to your printer. The print routine (PROCprint at line 2580) expects to find a machine code screen dump resident at memory location &A00, (CALL &A00 in line 2600). Alternatively you can substitute a call to a printer dump ROM such as BEEBUG's own Dumpmaster.

PROGRAM NOTES

For those not using ViewSheet to create and edit data files, the format required is a simple list of numbers, the first indicating how many data items follow. The data is loaded from a file into the array data(N%), which has been dimensioned to 99 in line 120. The accumulated deviations for the cusum are worked out by line 1310 and are loaded into the similarly sized cusum(N%) array. The size of these arrays can be increased or decreased to suit your available memory and data file requirements. As they stand they will hold a Viewsheet column (datafile) with 100 entries.

Note that the user-defined character buffer from &B00 is used to store the title, x-axis label and y-axis label (see line 120). You may need to change this (to &900 for example) depending on your system and how you use it.

```
10 REM Program SERIES
20 REM Version B1.5
30 REM Author C.R.Woodings
40 REM BEEBUG Aug/Sept 1987
50 REM Program subject to copyright
60 :
100 MODE 7:ON ERROR GOTO 280
110 oscli=&FFF7:DIM command 40:p%=42:S
%=0
120 DIMdata(99),cusum(99):flag=0:filen
ame$="NO DATA":title=&B01:xlabel=&B51:yl
abel=&B97:param=FALSE:lines=1
130 ?(ylabel+21)=&0D:data%=FALSE
140 $title="Series Graph"
150 $xlabel="result number"
160 REPEAT
170 smooth=FALSE:MODE7:REPEAT:M%=FNmen
u:UNTIL M%>0 AND M%<8
180 IF M%=1 PROCtop:PROCdataload("Ente
r a data file name "):IF data% PROCanaly
sis
190 IF NOT data% THEN 240
200 IF M%=2 MODE0:PROCplot("series")
210 IF M%=3 PROCsmooth:MODE0:PROCplot(
"series")
220 IF M%=4 MODE0:PROCplot("cusum")
230 IF M%=5 PROCparameters:PROCanalysi
s
240 IF M%=6 CLS:INPUT"*" $command:PROC
oscli:PROCcontinue
250 UNTIL M%=7:VDU26,12:@%=&90A:*FX4
260 END
270 :
280 IF ERR=214 OR ERR=15 PROCerror("No
 valid data") ELSE IF ERR=222 THEN PROCe
rror("No such file") ELSE IF ERR<>17 CLS
:REPORT:PRINT" at line ";ERL ELSE GOTO 1
60
290 PROCcontinue:*DIR $
300 GOTO 160
310 :
1000 DEF FNyesno(x,y,A$):LOCAL ans
1010 PRINTTAB(x,y)STRING$(39-x,CHR$32)
1020 PRINTTAB(x,y)A$;" (Y/N) ? ";
1030 REPEAT:ans=(GETAND&DF):UNTILans=&5
9OR ans=&4E:PRINTCHR$ans
1040 =(CHR$ans="Y")
1050 :
1060 DEF PROCcontinue
1070 PRINTTAB(7,23)"Press any key to co
ntinue ";
1080 REPEAT UNTIL GET
1090 ENDPROC
1100 :
1110 DEF FNinput(len,loASC,hiASC)
1120 LOCALK%,Z%:K%=0:In$=""
1130 PRINT STRING$(len,".");STRING$(len
,CHR$8);:*FX15,1
1140 REPEAT:Z%=GET
1150 IFZ%=127 AND K%>0 K%=K%-1:In$=LEFT
$(In$,K%):VDUZ%,46,8:GOTO1180
1160 IFZ%>loASC AND K%<len AND Z%<hiASC
K%=K%+1:In$=In$+CHR$Z%:VDUZ%:GOTO1180
```

```
 1170 IF NOT (Z%=13 AND K%>0) VDU7
 1180 UNTILZ%=13
 1190 =In$
 1200 :
 1210 DEF PROCanalysis
 1220 sum=0:max=-1E37:min=1E37:maxval=-1
E37:minval=1E37
 1230 FORN%=1TOK%
 1240 IFdata(N%)>max max=data(N%)
 1250 IFdata(N%)<min min=data(N%)
 1260 sum=sum+data(N%)
 1270 NEXT
 1280 average=sum/K%
 1290 IF NOT param loval=min:top=max
 1300 FOR N%=1 TO K%
 1310 cusum(N%)=cusum(N%-1)+data(N%)-ave
rage
 1320 IFcusum(N%)>maxval maxval=cusum(N%
)
 1330 IFcusum(N%)<minval minval=cusum(N%
)
 1340 NEXT
 1350 Yser=800/(top-loval)
 1360 Ycus=800/(maxval-minval)
 1370 Xscale=1000/K%
 1380 ENDPROC
 1390 :
 1400 DEF PROCplot(B$)
 1410 PROCaxes(B$):PROCdrawline(B$)
 1420 PROClabel
 1430 PROCnotes:VDU22,7:PROCtop
 1440 IF B$="series" AND FNyesno(1,7,"Ov
erlay another series graph") PROCoverlay
:GOTO1430
 1450 IF FNyesno(1,7,"Copy the graph to
printer") PROCprint:GOTO1430
 1460 PROCreload:GOTO1430
 1470 ENDPROC
 1480 :
 1490 DEF FNmenu
 1500 PROCtop:VDU28,9,20,38,9
 1510 PRINT"1 Load Datafile""2 Sequence
 Plot""3 Exponentially Smoothed Plot""
4 Cumulative Sum Plot""5 Labels/Printou
t etc.""6 Star Command""7 Quit":VDU26
 1520 PRINTTAB(7,20)"Enter Number of Pro
cedure    ";:VDU8,8,8
 1530 temp$=GET$:PRINTtemp$
 1540 VDU28,0,24,39,4
 1550 =VALtemp$
 1560 :
 1570 DEF PROCdouble(row,words$,colour)
 1580 column=INT((40-LEN(words$))/2)-2
 1590 FORI=row TO row+1:PRINTTAB(column,
I);CHR$141;CHR$colour;words$:NEXT
 1600 ENDPROC
 1610 :
 1620 DEF PROClabel
 1630 X%=0:pos=16:FORN%=1 TO K% STEP K%
DIV20+1:X%=Xscale*N%
 1640 MOVEX%-12,12:PRINT"|"
 1650 MOVEX%-20,-36+pos:PRINT;N%:pos=-po
s:NEXT
 1660 FOR N%=0 TO (10+(B$="cusum")):Y%=N
%*80+24
 1670 IF B$="series" MOVE-124,Y%-14:PRIN
Tloval+N%*(top-loval)/10;"-" ELSE MOVE-1
24,minval*Ycus+N%*86.4:PRINTminval+N%*(m
axval-minval)/10;"-"
 1680 NEXT:VDU29,160;140;
 1690 D=LEN($xlabel):E=INT(51-D)/2:MOVE
E*16,-90:PRINT$xlabel;
 1700 IF (B$="series" AND smooth) PRINT"
 (Smoothed:f=";Ksmooth;")"
 1710 D=LEN($ylabel):E=INT(21-D)/2
 1720 FORN%=1 TO D
 1730 MOVE-148,(840-E*40)-N%*40:PRINTMID
$($ylabel,N%,1)
 1740 NEXT
 1750 VDU26:D=LEN($title):E=INT(81-D)/2:
MOVE E*16,1000:PRINT$title;
 1760 ENDPROC
 1770 :
 1780 DEF PROCparameters param=TRUE
 1790 ret$="<Return> accepts, or type in
 a new one:"
 1800 CLS:PRINTTAB(11,8)"Graph Title is:
-"''$title''ret$
 1810 temp$=FNinput(79,31,127):IF temp$<
>"" $title=temp$
 1820 CLS:PRINTTAB(11,8)"X-axis label is
 now:"''$xlabel''ret$
 1830 temp$=FNinput(50,31,127):IF temp$<
>"" $xlabel=temp$
 1840 CLS:PRINTTAB(11,8)"Y-axis label is
 now:"''$ylabel''ret$
 1850 temp$=FNinput(20,31,127):IF temp$<
>"" $ylabel=temp$
 1860 CLS:PRINTTAB(11,8)"Y-axis origin i
s at ";loval''ret$
 1870 temp$=FNinput(10,39,58):IF temp$<>
"" loval=VALtemp$
 1880 CLS:PRINTTAB(11,8)"Y-axis top is a
t ";top''ret$
 1890 temp$=FNinput(10,39,58):IF temp$<>
"" top=VALtemp$
 1900 CLS:IF NOT FNyesno(5,10,"Join up t
he Series Points") lines=0 ELSE lines=1
 1910 yser=800/(top-loval)
 1920 ENDPROC
 1930 :
 1940 DEF PROCdataload(message$)
 1950 data%=FALSE:*DIR D
 1960 PRINT'"Data files on this disc:"
 1970 *INFO D.*
 1980 PRINT'message$;
 1990 filename$=FNinput(12,32,126)
 2000 IF LEFT$(filename$,1)="*" THEN $co
mmand=filename$:filename$="NO DATA":PROC
oscli:GOTO1960
```

```
 2010 D=OPENUP(filename$)
 2020 INPUT#D,K%
 2030 FOR N%=1 TO K%:INPUT#D,data(N%):NE
XT
 2040 CLOSE#D:$ylabel=filename$:data%=TR
UE:*DIR $
 2050 ENDPROC
 2060 :
 2070 DEF PROCoscli
 2080 X%=command MOD 256
 2090 Y%=command DIV 256
 2100 CALL oscli
 2110 ENDPROC
 2120 :
 2130 DEF PROCerror(m$)
 2140 CLS:PRINT'm$
 2150 ENDPROC
 2160 :
 2170 DEF PROCsmooth:smooth=TRUE
 2180 CLS:PRINTTAB(0,9)"Enter Smoothing
Const. 1(min) to 9(max)"
 2190 PRINTTAB(19,11);:Ksmooth=VALFNinpu
t(1,48,58)
 2200 Ksmooth=(10-Ksmooth)/10
 2210 ENDPROC
 2220 :
 2230 DEF PROCnotes
 2240 *FX4,1
 2250 VDU4:REPEAT:A%=GET
 2260 IF(A%=138 AND VPOS<31) VDU10
 2270 IF(A%=139 AND VPOS>0) VDU11
 2280 IF(A%=136 AND POS>0) VDU8
 2290 IF(A%=137 AND POS<79) VDU9
 2300 IF A%>31 AND A%<128 PRINTCHR$(A%);
 2310 UNTIL A%=135
 2320 *SAVE SCREEN 3000 7FFF
 2330 ENDPROC
 2340 :
 2350 DEF PROCaxes(B$)
 2360 VDU26,19,1,0;0;19,0,7;0;:CLS:@%=&3
07
 2370 MOVE4,4:DRAW4,1019:DRAW1275,1019:D
RAW1275,4:DRAW4,4
 2380 VDU29,160;140;5:MOVE-4,-16:DRAW-4,
800:MOVE0,800:DRAW0,-16
 2390 IFB$="cusum" VDU29,160;940-maxval*
Ycus;
 2400 MOVE-20,0:DRAW1040,0
 2410 ENDPROC
 2420 :
 2430 DEF PROCdrawline(B$)
 2440 IFB$="cusum" MOVE0,0 ELSE MOVEXsca
le,(data(1)-loval)*Yser
 2450 X%=0:FORN%=1TOK%
 2460 X%=Xscale*N%
 2470 IF (smooth AND N%>1) Y%=((data(N%)
*Ksmooth+(Y%/Yser+loval)*(1-Ksmooth))-lo
val)*Yser ELSE Y%=(data(N%)-loval)*Yser
 2480 IFB$="cusum" DRAWX%,cusum(N%)*Ycus
 ELSE PLOTlines+4,X%,Y%:MOVEX%-8,Y%+8:PR
INTCHR$p%:MOVEX%,Y%
 2490 NEXT
 2500 ENDPROC
 2510 :
 2520 DEF PROCoverlay
 2530 CLS:PROCdataload("Enter overlay da
tafilename ")
 2540 PROCreload:VDU29,160;140;5
 2550 p%=p%+1:PROCdrawline(B$)
 2560 ENDPROC
 2570 :
 2580 DEF PROCprint
 2590 PROCreload
 2600 VDU26,2:CALL&A00:VDU3:REM expects
a screendump in Page &A
 2610 ENDPROC
 2620 :
 2630 DEF PROCtop
 2640 VDU26:CLS:CLOSE#0
 2650 PROCdouble(1,"SERIES GRAPHS: "+fil
ename$,132)
 2660 PRINTTAB(0,3)CHR$145STRING$(39,"£"
)
 2670 PRINTTAB(0,22)CHR$145STRING$(39,"£
")
 2680 VDU28,0,21,39,4
 2690 ENDPROC
 2700 :
 2710 DEF PROCreload
 2720 VDU22,0:VDU19,1,0;0:VDU19,0,7;0;
 2730 *LOAD SCREEN
 2740 ENDPROC
```

Ⓑ

BEEBUG WORKSHOP (continued)

```
   IF-THEN-ELSE IF-THEN-ELSE...
```

than:

```
   IF-THEN-IF...
```

In other words, complete one IF-THEN-ELSE before tagging on another IF. If you do not follow this suggestion you will never be 100% sure which ELSE goes with which IF.

If you have kept up with me, it may have come as some surprise that such a common and apparently simple statement as the IF-THEN-ELSE should be so involved. By following the good practices outlined here you should have no trouble at all.

Ⓑ

Colouring Archie

Lee Calcraft's second foray into Archie's workings concentrates on colour. As well as introducing the colour commands on the new machine, a program is presented to allow easy experimentation with the 4096 different colours using the keypad as a fader control.

As you will probably already know, the Archimedes is capable of displaying up to 256 colours on screen at once, chosen (in groups of 16) from a palette of 4096. This represents a most welcome leap forward from the very limited colour repertoire of its 6502-bound ancestor. Just how impressive the Archimedes displays can be, may be seen from watching the Desktop in action. But how does the humble programmer conjure up these colours for himself?

It is all done with a mixture of GCOL, VDU19, a set of variants of the COLOUR command, plus the new command TINT. And if you are happy with the machine's pre-defined colours for any particular mode, then it is all very straightforward. In all but mode 7, you just use the command:

```
COLOUR n
```

and all subsequent text will appear in the predefined logical colour number n, providing that physical and logical colours are aligned. Similarly for graphics. Just use the new single-parameter version of GCOL:

```
GCOL n
```

where n defines the logical colour to be used for the next plot. In the 2, 4, and 16 colour modes, the colour numbers are exactly the same as those of the model B, including the use of flashing colours in the 16 colour modes.

256 COLOUR MODES

The 256 colour modes however (i.e. modes 10, 13 and 15), behave differently from the remainder in many respects. In the present case, they differ in that there are 64 predefined colours available directly by stating a colour number. This is brought up to 256 by use of the TINT command to specify up to four different hues of each of the 64 preset colours. The way in which the 64 colours are arranged is particularly useful. The bottom two bits of the COLOUR number determine the degree of red in the colour, the next two bits the green, and the next two the quantity of blue. If you are in mode 15, and enter:

```
COLOUR %110000
```

all subsequent text will appear in deep blue.

We have made use here of the Archimedes' extremely useful binary operator (%) to enter a colour number 110000 binary (=48 decimal), giving maximum blue, and zero red and green. If we change this to:

```
COLOUR %110011
```

we will get deep magenta (maximum blue and red, with no green). To draw a deep magenta rectangle, use:

```
GCOL %110011        (i.e. GCOL 51)
RECTANGLEFILL 500,500,200,200
```

Note the new user friendly rectangle fill command, and there are counterparts for triangles, circles and ellipses.

MIXING YOUR OWN

It is important to remember that at this stage we are not defining our own colours, only choosing one of the 256-colour modes' 64 default colours. Mixing your own colours is accomplished with relative ease providing you are not in one of the 256 colour modes; and the following does not apply to these.

To compose a specific colour in modes providing up to 16 colours, you can use a special variant of the COLOUR command:

```
COLOUR n,r,g,b
```

where n is the logical number of the colour being defined, and r, g and b are the red green and blue components of that colour respectively. The values of r, g and b must be given in multiples of 16 (since only the top 4 bits of each byte are used), with a maximum of 240. In practice it is easier to think of the command as:

```
COLOUR n, r*16,g*16,b*16
```

where r, g and b now take values between 0 and 15. Once a new logical colour has been

created in this way, you must still issue a COLOUR n or a GCOL n command to put that colour to use of course.

COLOUR BLENDER

The program is based on the use of COLOUR n,r,g,b and allows the user to mix his own colours from the full palette of 4096. When it is run, it produces a central rectangle showing the currently defined colour (white by default), and below this are three smaller squares showing the red, green and blue components of the central area.Keys on the numeric keypad are then used to raise or lower the three component colours. A fourth pair are used to increase or decrease all three components together, permitting selected hues to be lightened and darkened, and allowing the easy display of the grey scale. The latter consists of colours made up with equal proportions of the three primaries. The screen also displays the parameters of the COLOUR command required to produce the exact blend currently selected.

There are a couple of points worth noting in the program. The first is the use of the time wasting loop in line 180 The Archimedes responds far too quickly without this. Secondly, I have made use of two of the machine's new assignment operators, += and -=. They work as follows:

```
FRED += 4
```

has the same effect as:

```
FRED = FRED + 4
```

BORDERING

The Archimedes VIDC chip supports the use of coloured borders surrounding the normal screen area. These may be used in all modes but 7, and are called up with a variant of the VDU19 command. It takes the following form:

```
VDU19,0,24,r*16,g*16,b*16
```

where r, g and b can take values from 0 to 15, specifying the red green and blue components of the border. Interestingly enough, this command works just as well in the 256 colour modes, even though the four parameter COLOUR command discussed above, with its similar syntax, does not.

```
  10 REM Program Blender
  20 REM Version B1.3
  30 REM Author  Lee Calcraft
  40 REM BEEBUG  Aug/Sept 1987
  50 REM Program subject to copyright
  60 :
 100 ON ERROR GOTO 1340
 110 MODE12
 120 VDU23,1|
 130 red=15:green=15:blue=15
 140 PROCinstr
 150 REPEAT
 160 PROCkeys
 170 PROCboxes
 180 TIME=0:REPEAT UNTIL TIME=10
 190 UNTIL FALSE
 200 END
 210 :
1000 DEFPROCboxes
1010 GCOL 1
1020 RECTANGLEFILL 345,420,590,500
1030 FORA=2TO4
1040 PROCrect
1050 NEXT
1060 ENDPROC
1070 :
1080 DEFPROCrect
1090 GCOL A
1100 RECTANGLEFILL (200*A-55),200,1
90,200
1110 PRINTTAB(22,27)"COLOUR,logical col
our, ";16*red;", ";16*green;", ";16*blue
;STRING$(8,CHR$32)
1120 ENDPROC
1125 :
1130 DEFPROCkeys
1140 IF INKEY(-28) red-=red<15
1150 IF INKEY(-123) red+=red>0
1160 IF INKEY(-43) green-=green<15
1170 IF INKEY(-124) green+=green>0
1180 IF INKEY(-44) blue-=blue<15
1190 IF INKEY(-27) blue+=blue>0
1200 IF INKEY(-60) red-=red<15:green-=g
reen<15:blue-=blue<15
1210 IF INKEY(-59) red+=red>0:green+=gr
een>0:blue+=blue>0
1220 COLOUR 1,red*16,green*16,blue*15
1230 COLOUR 2,red*16,0,0
1240 COLOUR 3,0,green*16,0
1250 COLOUR 4,0,0,blue*16
1260 ENDPROC
1265 :
1270 DEFPROCinstr
1280 PRINTTAB(26,1)"C O L O U R    B L
E N D E R"
1290 PRINTTAB(6,13)"K E Y P A D"
1300 PRINTTAB(4,15)"RED GRN BLU GRY"
1310 PRINTTAB(0,17)"INC  7   8   9   -"
1320 PRINTTAB(0,19)"DEC  4   5   6   +"
1330 ENDPROC
1335 :
1340 MODE12:REPORT:PRINT" at line ";ERL
```

Ever so Spritely

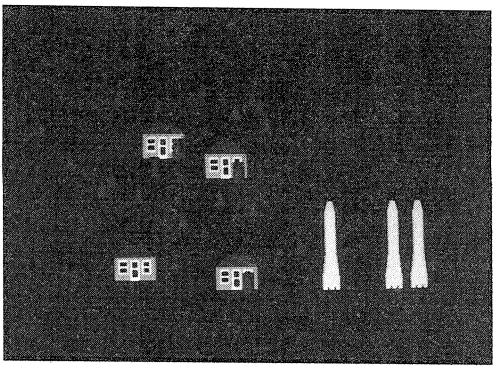
Generating and manipulating multi-coloured sprites becomes child's play with Archimedes, as Mike Williams shows.

Sprites are essentially multi-coloured user-defined characters, but in the past, programming these objects was fraught with difficulty unless one had access to suitable commercial software designed for the purpose. Archimedes changes all this, as sprites are an intrinsic feature of its operating system, with a host of star commands for their manipulation. In addition, Acorn have thoughtfully provided an excellent sprite editor (called SEDIT) on the Welcome disc supplied with the machine.

If you want to incorporate sprites in your programs, and they have many applications other than games, then the starting point is the sprite editor. Initially this displays a table of sprite definitions currently known to the system. If you have only just switched on then this table will be empty, otherwise you will see listed the sprites last used. It is probably best to start by clearing this table.

To create a sprite you must first enter a name for your sprite in the table, and specify the screen mode in which it will be used. The editor also allocates an initial size to a sprite, dependent upon the mode selected. As many sprites as you wish may be entered into this table, and the mouse (what else?) may be used to select any sprite by moving the pointer to its entry in the table and clicking. A further click on a selected sprite takes you into the edit screen.

Here, the sprite is shown enlarged to the left of the screen, and in its real size to the right. At the top of the screen there is a colour palette, which will clearly be mode dependent. In mode 15 for example you have 256 colours available in four multi-coloured bands across the screen. As well as the solid colours, four colour patterns are available. Use the mouse to select a colour, move back to the enlarged version of your sprite and click to fill in pixels. In addition, the middle button selects the logical colour for text or graphics (white by default) and the right hand button selects the background colour (default black).



Sprites generated with the sprite editor

You are not limited to the initial number of rows and columns for your sprite. Function keys f3 and f4 may be used to enlarge the sprite by a row or column at a time respectively. As you increase the sprite's size in this way, so both the enlarged and actual sprite displays increase. The enlarged version is more quickly limited by screen size, and then becomes a window showing just part of a much larger sprite. Moving the pointer to the position on the actual size representation allows the window to be moved around. Rows and columns may also be deleted using Shift-f3 and Shift-f4.

The size of any sprite is limited overall by the amount of memory allocated for sprite definitions. Archimedes can be configured to allocate any desired amount of memory on start-up, for example:

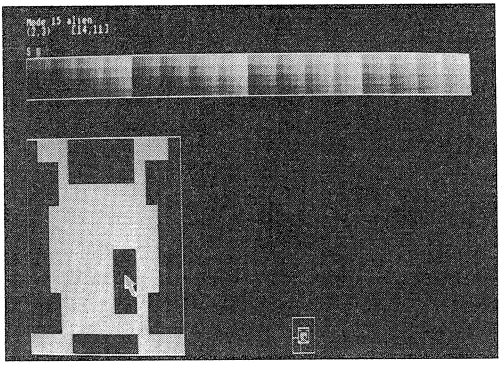
```
*CONFIGURE Spritesize 3
```

but this may also be changed dynamically using the equivalent *SSPACE 3. Memory for sprites is allocated in chunks of 32K bytes at a time so the 3 above reserves 96K for sprites. In use, the lack of sufficient memory is indicated by the error message "sprite cache".

As well as adding (or deleting) rows and columns at the top and righthand edges of your sprite definition, you can also add or remove rows and columns at any point, by using the

mouse to select a position and pressing f5 or f6 (Shift-f5 and Shift-f6 to delete). Other useful features include the use of f1 and f2 to colour whole rows and columns. Further, f7 and f8 allow you to reverse a sprite either left/right or up/down - useful for moving sprites which need different versions facing different ways.

Using these features I was able to define four multi-coloured sprites in no time at all. Press Escape to get back to the sprite table, and you have an option to save all your sprite definitions away to a file. The file can always be reloaded if more editing is required or if you want to add to the number of sprites.



Sprite editor in use

At this stage you can start writing your sprite program. The short program listed here demonstrates the use of some of the star commands available for this purpose. The *SLOAD command at line 110 loads in the specified sprite definitions (called Alien, Tree, Rocket and House in my example). There then follow three similar sections which use further star commands. *SCOPY (used with OSCLI at line 140) duplicates a sprite definition with a new name. This loop creates three more rockets called Rocket1, Rocket2 and Rocket3. Similarly, in the following sections, House spawns a further four houses, and Tree propogates no less than 20 trees. None of this is strictly necessary here (as we can plot any sprite in many different positions), but it does illustrate the power of the sprite commands.

The other star command used is *SCHOOSE (see line 150 for example) which makes the chosen sprite the currently active one. This can then be plotted (using a new version of PLOT with parameters in the range &E8 to &EF) as in line 160.

A final touch is the use of a 'mask'. As normally defined a sprite occupies a rectangular shape so that those pixels not defined by the user remain in black (or in the background colour used in the original definition). Move such a sprite across another coloured area on the screen and this frame becomes all to obvious. A mask allows these parts to be transparent so that the background shows through. This is achieved using the editor, specifying that a mask is to be used (Shift-f9) and selecting 'transparent' as the colour (f9). To use a mask, a sprite must be plotted with 8 added to the first GCOL parameter (as in line 280). The loop from line 310 uses the mouse to move the alien around the random screen display of houses, trees and rockets, no doubt an everyday scene for any self respecting alien!

```
  10 REM Sprites v2
 100 MODE 15
 110 *SLOAD SpriteDef
 120 GCOL 8,0
 130 FOR I%=1 TO 3
 140 OSCLI("SCOPY ROCKET ROCKET"+STR$I%
)
 150 OSCLI("SCHOOSE ROCKET"+STR$I%)
 160 PLOT &ED,RND(300)+800,200
 170 NEXT I%
 180 FOR I%=1 TO 4
 190 OSCLI("SCOPY HOUSE HOUSE"+STR$I%)
 200 OSCLI("SCHOOSE HOUSE"+STR$I%)
 210 PLOT &ED,RND(400)+200,RND(500)+150
 220 NEXT I%
 230 FOR I%=1 TO 20
 240 OSCLI("SCOPY TREE TREE"+STR$I%)
 250 OSCLI("SCHOOSE TREE"+STR$I%)
 260 PLOT &ED,RND(600)+200,RND(600)+150
 270 NEXT I%
 280 GCOL 11,0
 290 *SCHOOSE ALIEN
 300 oldx=640:oldy=512:PLOT &ED,oldx,ol
dy
 310 REPEAT
 320 *FX19
 330 MOUSE x,y,b
 340 PLOT &ED,oldx,oldy:PLOT &ED,x,y:ol
dx=x:oldy=y
 350 UNTIL FALSE
 360 END                                B
```

1st course

Plotting for Effect (Part 3)

In this final part in our mini-series on the PLOT instruction, Mike Williams takes a look at some of the many variations that are possible in the use of that most versatile of graphics commands.

So far, we have looked at two very specific uses of the PLOT instruction in Basic, PLOT85 to display solid-looking triangles, and PLOT77 which we can use to fill or colour any shape (though Master and Compact users have many other alternatives available). At this stage it is useful to realise that PLOT is a general purpose graphics instruction unlike the MOVE and DRAW instructions that we referred to previously.

RELATIVE PLOTTING

In the past, whenever we wanted to refer to a point on the screen, we have specified a position (X,Y) on the screen, where X and Y are the horizontal and vertical distances from the origin (0,0). This is called an absolute reference or position. The point (0,0) is normally at the bottom left hand corner of the screen, though you can change this, as we did for circle plotting, using the VDU29 instruction. For example, here is a simple procedure for drawing a square:

```
1000 DEF PROCsquare(size,x,y)
1010 MOVE x,y
1020 DRAW x,y+size
1030 DRAW x+size,y+size
1040 DRAW x+size,y
1050 DRAW x,y
1060 ENDPROC
```

There is a second way of specifying a point on the screen, by giving the distance from the last point visited by the graphics cursor, rather than the absolute or fixed position. This uses PLOT1 instead of DRAW (or PLOT5). If we rewrite the procedure above, it will read as shown below.

If you type in either of these procedures, you can test them out in immediate mode, provided you select a graphics mode first, by just typing PROCsquare followed

```
1000 DEF PROCsquare(size,x,y)
1010 PLOT 0,x,y
1020 PLOT 1,0,size
1030 PLOT 1,size,0
1040 PLOT 1,0,-size
1050 PLOT 1,-size,0
1060 ENDPROC
```

by suitable parameters in brackets. For example:

```
MODE 4
PROCsquare(200,640,512)
```

This will draw a square of side 200 with its bottom left hand corner in position (640,512).

In the second version of the procedure, I have used PLOT instructions throughout, although I could just as easily have used a MOVE instruction at line 1010 instead. The instructions used here to draw the square could be written as follows: "from the starting point, draw a distance 'size' vertically upwards, then a line the same distance horizontally, then another line the same distance but now downwards (hence the minus sign), and lastly a fourth line the same distance again back to the starting point (to the left, and thus again minus)". At no time do we use the actual position of any corner of the square as everything is specified relatively.

Whether you prefer the first or the second method is, I think, largely a matter of personal choice, though some might argue that the second method involves less arithmetic, and is thus both simpler and faster. When you are deciding which method to use, be guided by the circumstances, but always try to choose the simplest method if a choice is available.

DOTTED OR SOLID LINES

Another option available with the PLOT instruction is the drawing of dotted instead of the more usual solid lines. Adding 16 to the

'PLOT' number will produce this effect. Thus PLOT17 is the same as PLOT1 (relative plotting) but with a dotted line, and PLOT21 is likewise similar to PLOT5 (absolute plotting).

Although a PLOT instruction is normally listed with a specific 'PLOT' number, this can instead be replaced by a variable. For example, we will rewrite the last procedure so that it will draw a square with either a solid or a dotted line. We will do this by introducing an extra parameter, which I'll call 'type'. This will have the value 0 for a square of solid lines, and 1 for a square with dotted lines. The revised version of our procedure is as follows:

```
1000 DEFPROCsquare(type,size,x,y)
1010 t=16*type+1
1020 PLOT 0,x,y
1030 PLOT t,0,size
1040 PLOT t,size,0
1050 PLOT t,-size
1060 PLOT t,-size,0
1070 ENDPROC
```

If you type this in, you can again test it out in immediate mode. Thus:

```
PROCsquare(0,400,200,200)
```

will give a square of size 400 and in position (200,200) drawn with solid lines while:

```
PROCsquare(1,400,600,600)
```

will give a square of the same size in position (600,600) but drawn with dotted lines.

The procedure first calculates the correct 'PLOT' number for the type of line required, and assigns this value to the variable 't'. This is then used with the PLOT instruction to give dotted or solid lines. The facility to replace the 'PLOT' number with a variable is a very useful and flexible feature of the PLOT instruction. Some surprising results may often be had when a little ingenuity is used.

PLOTTING POINTS

Another useful variation on the PLOT instruction provides for the plotting of a point, rather than a line. Again the position of the point can either be specified as an absolute position on the screen (using PLOT69) or relative to the last position (using PLOT65). This enables single points to be positioned anywhere you want on the screen.

SUMMARY OF PLOT INSTRUCTIONS

All PLOT instructions have the same basic format of PLOT k,x,y where x,y is a relative or an absolute position on the screen, and k determines the plotting function carried out. All PLOT instructions come in groups of eight, while in each group the offset from the base value has the following meaning:

0	Move cursor relative (to last point)
1	Draw relative in current foreground
2	Draw relative in logical inverse
3	Draw relative in current background
4	Move cursor absolute
5	Draw absolute in current foreground
6	Draw absolute in logical inverse
7	Draw absolute in current background

This offset determines whether absolute or relative co-ordinates are to be used and the way in which colour is used. For a detailed list of PLOT numbers refer to the appropriate User Guide for your machine. The principal features are, however, summarised in Table 1.

All Systems	
0-63	Solid and dotted lines
64-71	Point plot
72-79	Fill left and right
80-87	Triangle fill
88-95	Fill right only
Master, Compact & Archimedes	
96-103	Rectangle fill
104-111	Fill left and right
112-119	Parallelogram fill
120-127	Fill right only
128-143	Flood fill
144-159	Outline & solid circle
160-167	Circular arc
168-175	Segment of a circle
176-183	Sector of a circle
184-191	Block copy/move
192-207	Outline & solid ellipse
Archimedes only	
208-215	Graphics characters
216-231	Reserved
232-239	Sprite plot
240-255	Reserved

Table 1. Summary of PLOT Commands

INTELLIGENT MODEMS
(Part 2)

Peter Rochford concludes the reviews of the latest in intelligent modems.

TANDATA TM722
Price 573.85 inc VAT

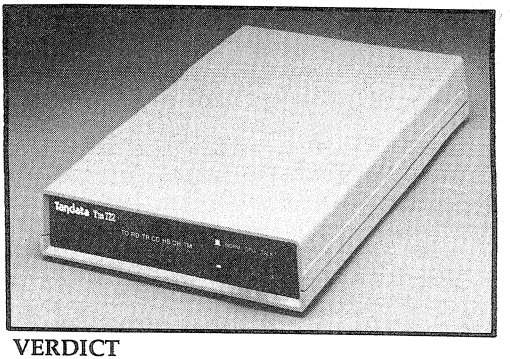
This is a high speed V22/V22bis modem without V21/V23 which I find rather surprising considering the cost. The review sample, which I am told by Tandata's PR company is a prototype, came housed in a grey case with black front panel.

The front panel sports some seven LEDs for monitoring modem status along with three push button switches. The switches are for self test, and to enable the modem to be switched into originate/answer modes without using the terminal it is connected to. The rear of the modem includes a connection to the external PSU, and a set of DIP switches for configuring the modem in a specific state on power-up.

Documentation supplied with the review sample was a computer printout of the manual which at the time of the review was not back from the printer's. The manual appears to be well-written and the final version should be fine.

In use the TM722 worked perfectly and cost me a fortune in phone calls as the only service I could access at 2400/2400 to test it out fully was the Prestel local access number in Reading (I live in London). I hope that the BT VASSCOM high speed network soon comes on line!

There is not much more I can say about this modem. The software follows the standard Hayes protocols, and the only extra feature of note is the error correction which can be specified as one of several including that for the BT VASSCOM network.



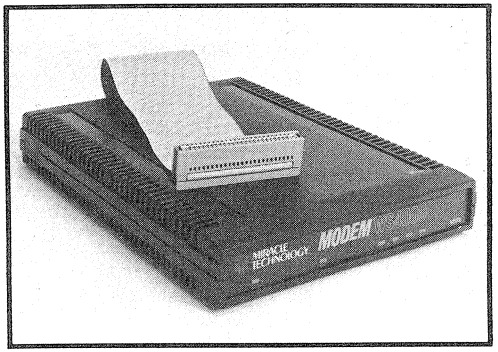
VERDICT
The TM722 is a rather unassuming looking piece of equipment considering the cost and I find the exclusion of V21/V23 operation rather naughty. If I had this kind of money to spend on a modem there is no doubt in my mind that I would rather settle for the Pace Series Four. It has so much more to offer.

MIRACLE WS3000		
Price	339.25	(V21/V23)
	569.25	(V21/V23/V22)
	747.50	(V21/V23/V22/V22bis)
MIRACLE WS4000		
Price	195.44	(V21/V23)
	482.94	(V21/V23/V22)
	661.19	(V21/V23/V22/V22bis)
Note	Prices are inclusive of VAT	

I have grouped these two modems together as they are so similar in design and construction. In fact, the WS4000 is just a stripped down version of the WS3000. The WS3000 comes housed in a black plastic case with grey front panel, whilst the WS4000 has a reddish brown casing with dark blue front panel. There are no controls on the front of either modem, and both are graced with a bank of 10 LEDs for indicating modem status.

At the rear of the modems there are the standard connections with a lead for the external free-standing PSU. There are also software reset switches and, in the case of the WS3000, a control port socket. There is very little mention in the manual as to the use of the WS3000 control port, except a brief word saying that a printer may be connected to it for

printing out the contents of the number store. The standard of construction of the WS modems is very good indeed and the styling to me, is quite pleasing. Documentation consists of an 80 page manual which is well-written and presented. Plenty of information here for those who need it in-depth, whilst the newcomer will be gently led by the hand.



In use, the modems performed without any problems. They both use the standard Hayes AT set for control along with the usual set of S registers. Like the Tandata modems, the Miracles cannot detect dialtone or engaged signals which the Pace can, and neither can they detect automatically whether the line needs to be pulse or tone dialled.

A point to note regarding the WS4000 is that it does not have battery-backup for the number store. Odd!

VERDICT

I find it rather strange that Miracle should have two modems so similar on the market at the same time. It is true that the WS3000 has more facilities than the WS4000 but for what is being offered as extra, the WS3000 is very pricy. In fact, I feel that although these are both excellent modems, they are rather overpriced at the moment, especially with regard to the V22bis versions.

Supplier: Pace Micro Technology
Allerton Road,
Bradford,
West Yorkshire,
BD15 7AG
Tel. (0274) 488211

Supplier: Miracle Technology
St. Peters Street,
Ipswich,
IP1 1XB
Tel. (0473) 216141

Supplier: Tandata
Albert Road North,
Malvern,
Worsc.,
WR14 2TL
Tel. (0684) 892421

A list of the features of all the modems is listed below and provides an at-a-glance guide to what they all offer.

FEATURES AND FACILITIES

MODEM	V2	V23	V22	V22bis	AD	AA	LS	HY	V25	EC	NS	B	TP
Pace Linnet	Y	Y	N	N	Y	Y	Y	Y	N	N	30	N	P
Tandata TM512	Y	Y	N	N	Y	Y	Y	Y	Y	Y	8	N	TP
Miracle WS4000	Y	Y	OPT	OPT	Y	Y	OPT	Y	N	N	OPT	OPT	P
Pace Series 4	Y	Y	Y	Y	Y	Y	Y	Y	N	N	64	N	TP
Tandata TM722	N	N	Y	Y	Y	Y	Y	Y	N	Y	16	N	TP
Miracle WS3000	Y	Y	OPT	OPT	Y	Y	Y	Y	N	N	60	Y	TP

Key to facilities:

V21	300/300	V23	1200/7	V22	1200/1200
V22bis	2400/2400	AD	Auto Dial	AA	Auto Answer
LS	Loudspeaker	HY	Hayes compatible	V25	V25 command set
EC	Error correction	NS	Number Store	B	Bell (USA) standards
TP	Tone and Pulse Dialling			OPT	Option (see text)

CALLING BASIC FROM MACHINE CODE

Claus Alsted

Should you wish to call Basic directly from machine code, this can be done with one of the three statements below, depending on which operating system you have:

```
JMP &E018 \ on OS 1.2
JMP &E11B \ on OS 2.0
JMP &E4B0 \ on OS 3.2
```

This is equivalent to *BASIC but is less convoluted than an OSCLI call. A Basic program may be CHAINed from machine code in the following manner as shown below:

```
....        \ somewhere
JMP chn     \ in the
....        \ program

.cmd EQUS "K.0 CH."+CHR$34+"<prog>"+CHR$34+"|M"

.chn LDX#cmd AND &FF  \ LSB of command string
     LDY#cmd DIV &100 \ MSB ditto
     JSR oscli        \ Define fn key 0
     LDA#138          \ Enter fn key 0
     LDX#0            \ ASCII 128 into -
     LDY#128          \ keyboard buffer
     JSR osbyte       \ "Press" fn key 0
     JMP &E018        \ Enter Basic cmd mode
```

The point is that the character inserted in the keyboard buffer is not released until Basic command mode is reached. As the single ASCII code 128 is the same as function key 0, the following command:

```
CH."<prog>"<return>
```

is given and the program is CHAINed.

ADFS COMPACT

S.Williams.

In the ADFS it is sometimes necessary to issue the *COMPACT command more than once if you wish to reduce the disc to its most compact form. By issuing the *MAP command after each compact, you can check if the next compact has achieved any extra reduction (see also Intelligent Disc Compactor).

SAVING BASIC PROGRAMS

Frank Mallett

This seems rather an obvious hint, but I've found it very useful. Include a line say:

```
10 REM SAVE"name"
```

in all your programs. This will prevent you forgetting the name, and will facilitate saving by simply tracing over SAVE"name" using the cursor keys. Interested readers should refer to Beebug Vol.6 No.2, which contains an Auto-Save utility, which goes further than the above hint. You will also find that our Master ROM (usable on BBC Model Bs too) has a *S command, which looks for the embedded name and saves the program, automatically updating the name in the program if it has been changed.

WW+ CAN'T EXECUTE

Rob Barnes

From time to time Wordwise Plus programs which have previously worked well throw up "Can't execute" errors. This is probably the consequence of pressing Escape or Break during printing. If this happens, "Preview" your text or the segment program, and try again.

INTERWORD AND DOUBLE SPACED LINES

Dr M.A.Healey

There is often a need to print a document with double spaced lines. Unfortunately when line spacing is set to 2, this causes Interword to print one page across two sheets of paper.

To overcome this, use sub menu 4 to set lines per page to 33, top space to 3, bottom space to 3, header position to 1, and footer position to 32. Now use menu 5 to set line spacing to 2. Print, and double spacing is achieved, with correct page breaks.

MASTERING DYNAMIC OVERLAYS

In BEEBUG Vol.3 No.2 you published an excellent program to allow procedures and functions to be dynamically loaded from disc and overlaid in memory. This allows quite large programs to be used with ease. Is it possible to tell me how to modify the original program so that it will run with Basic IV on the Master?

Dr Francis Greaves

The original program contained routines to cater for Basic I and II. To modify the program for other versions of Basic, delete lines 370 to 490 inclusive, and replace the values in lines 500 to 580:

Line No.	Tube Hi-Basic	Master Basic IV	Compact Basic VI
500	&1F	&E4	&C5
520	&C4	&96	&96
540	&E5	&60	&BA
560	&E9	&B0	&AF

Our thanks to Roger Cullis for this information.

VIEWSHEET AND THE ADFS

*With reference to Mr Reid's letter on using ViewSheet on the Master (Postbag Vol.5 No.10), I would suggest that the best solution is not a patch to ViewSheet, but to set up a dedicated ViewSheet disc for use with the ADFS. In the root directory ($), set up sub-directories with names like BUDGET, SALES, TAXES etc. Then with *MOVE, or even using ViewSheet itself, copy the data and window files, preferably with somewhat more expressive names, to the appropriate new directories. In each case an appropriate *TITLE line may also be entered. In this way one may make the best of the ADFS.*

Acorn themselves are to blame for the lack of understanding with regard to the ADFS. The 'best' description to date is nowhere in the Master 'info', but in the dedicated manual provided with the Electron's Plus 3!

J.Duis

Many users find problems in converting from the DFS to the ADFS, but properly used the ADFS has many advantages, allowing much more information to be stored, and for better organised.

HELP ACROSS THE TUBE

I was very interested in the Help program (Vol.5 No.10), but I discovered two problems. The program as written will not work across the Tube, nor will it function correctly with the ADFS.

When using the Tube, PAGE is set to &800 so the Help program overwrites itself when assembling. The way to overcome this is to assemble higher in memory in the I/O processor by changing the following lines:

```
 140 PROCassemble(&2A00)
1980 .noway:JMP &2C00
2060 A$="*SAVE Query "+STR$
~Q%+" "+STR$~P%+" FFFF"+STR
$~execute+" FFFF"+STR$~Q%
```

*Note that it is necessary to switch the Tube off when running the Help program to create the Query file. Once created, switch on the Tube and call *RUN Query.*

*When operating the Help system using the ADFS, typing *HLP <Return> results in the whole H.DATA file being displayed. This can be cured by adding:*

```
1785 STA &79:LDA &79
```

Alan Mothersole

Thanks to Mr Mothersole for more useful information. The ADFS problem arises because the zero bit is set differently using the ADFS compared with the DFS in the OSBGET call at line 1780. This is taken into account in the program except in line 1460 where an alternative solution is to add CMP#0 immediately following JSR getbyte.

MORE FOR THE COMPACT

I found the following Basic VI (Compact) entry points for the Comparator utility in BEEBUG (Vol.5 No.5). I hope this is of interest.

Name	*Basic IV*	*Basic VI*
pchar	*&BD98*	*&BCF8*
ptoken	*&BD37*	*&BC97*
plineno	*&A081*	*&A004*
chkprg	*&BDE5*	*&BD45*
chkgoto	*&9B26*	*&9B06*
exit	*&8F86*	*&8F67*

D.J.Pilling

Refer to the DATA statements in the program (Basic IV given above for reference). Ignore any resulting checksum error.

BEEBUG MEMBERSHIP

Send applications for membership, membership renewals, membership queries and orders for back issues to the address shown. All membership fees, including overseas, should be in pounds sterling drawn (for cheques) on a UK bank.

MEMBERSHIP SUBSCRIPTION RATES

£6.90 - 6 months (5 issues) UK ONLY

£12.90 - 1 year (10 issues) UK, B.F.P.O., Channel Islands

£19.00 - Rest of Europe

£26.00 - Americas & Africa

£23.50 - Middle East

£28.00 - Elsewhere

BACK ISSUES (Members only)

VOLUME	SINGLE ISSUES	VOLUME SETS (10 ISSUES)
1	40p	single issues 7, 9 & 10 only
2	50p	£3.50
3	70p	£5.50
4	90p	£7.50
5	£1.20	£10.50
6	£1.30	–

DISCOUNT: Any 10 or more issues, deduct £1.50 from the total single issue price. 20 or more deduct £3.00, 30 or more deduct £4.50, etc.

Please add the cost of post and packing as shown:

DESTINATION	FIRST ISSUE	EACH SUBSEQUENT ISSUE
UK, BFPO + CH. IS.	40p	20p
Europe + Eire	75p	45p
Elsewhere	£2	85p

All overseas items are sent airmail (please send a sterling cheque). 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 that require an invoice. Note that there is no VAT on magazines.

Back issues are for members only, so it is ESSENTIAL to quote your membership number with your order.

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

St. Albans (0727) 40303
Manned Mon-Fri 9am-5.00pm

(24hr Answerphone Service for Access/Visa orders and subscriptions)

If you require members discount it is essential to quote your membership number and claim the discount when ordering.

BEEBUG MAGAZINE is produced by BEEBUG Ltd.

Editor: Mike Williams
Editorial Assistant: Kristina Lucas
Production Assistant: Yolanda Turuelo
Membership Secretary: Sara Anketell-Jones
Editorial Consultant: Lee Calcraft

Additional thanks are due to Sheridan Williams, Adrian Calcraft, John Yale, Tim Powys-Lybbe, and Dave Somers.

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 of Guidance for Contributors', is available on receipt of an A5 (or larger) SAE.

In the case of material longer than a page, we would prefer this to be submitted on cassette or disc in machine readable form using "Wordwise", "View", or other means, but please ensure an adequate written description of your contribution is also included. If you use cassette, please include a backup copy at 300 baud.

In all communications, please quote your membership number, and enclose a SAE if you require a reply.

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

Produced by Head Office Design (0782) 286375 ISSN – 0263 – 7561

Magazine Cassette/Disc

AUG/SEPT 1987 CASSETTE/DISC CONTENTS

KEYSTRIP GENERATOR — impressive utility with smooth scrolling display for editing and printing all your function key strips.

PAINTING AND DRAWING — for amusement and education, a versatile painting and drawing program for all those rainy days.

THE MASTER SERIES

INTERRUPT DRIVEN CLOCK — get your Master to display a continuously updated time and date on the screen while you work.

INTELLIGENT DISC COMPACTOR — a short utility to keep your disc files in order.

BARRY CHRISTIE VISUALS

CUSTOM CLEARANCE — three classy routines for bringing a graphics display to an exciting close.

ARCHIMEDES

EVER SO SPRITELY — short program demonstrating the use of Archimedes' sprite commands.

COLOURING ARCHIE — program demonstrating colour mixing in 4096 shades.

USING THE ARM ASSEMBLER — demonstration program using the ARM assembler.

6502 DEBUG — a utility for machine code programmers that can be used for a dynamic display of register contents.

BEEBUG WORKSHOP — useful program for evaluating logical relationships.

EXPLORING ASSEMBLER (PART 2) — two complete example programs in our introduction to machine code programming.

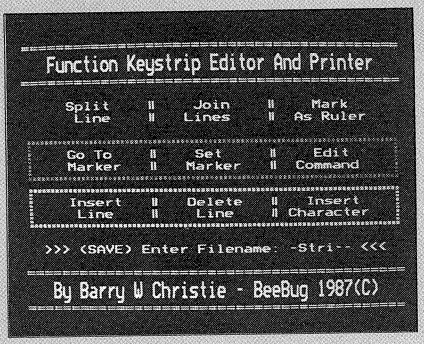
TIME SERIES ANALYSIS — this application program will display and analyse sets of data collected at regular time intervals.

ANIMATED TITLING — six procedures for 'shooting' text in all directions across the screen, all packaged up in a scintillating demonstration.

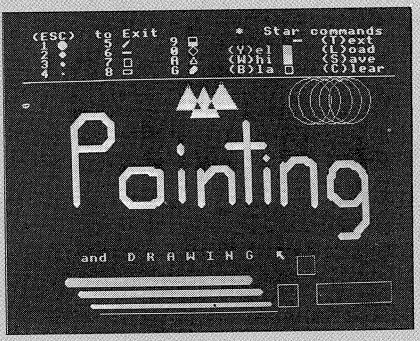
EXTRA FEATURES THIS MONTH

MAGSCAN — data for this issue of BEEBUG (Vol.6 No.4).

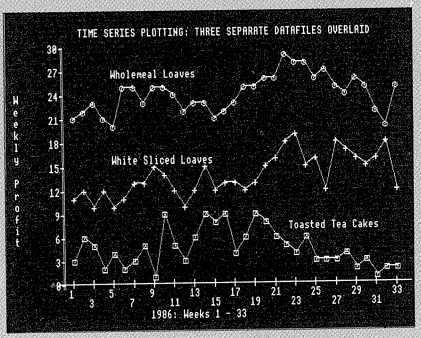
Keystrip Generator



BEEBUG Paint and Draw



Time Series Analysis



All this for £3.00 (cass) £4.75 ($5\frac{1}{4}$" disc) £5.75 ($3\frac{1}{2}$" disc) + 50p p&p.
Back issues ($5\frac{1}{4}$" disc since Vol.3 No.1, cass since Vol.1 No.10) available at the same prices.

Subscription rates (UK only)	**$5\frac{1}{4}$" Disc**	**$3\frac{1}{2}$" Disc**	**Cass**	**Subscription rates** (Overseas)	**$5\frac{1}{4}$" Disc**	**$3\frac{1}{2}$" Disc**	**Cass**
6 Months (5 issues)	£25.50	£29.50	£17.00	6 months (5 issues)	£30.00	£34.00	£20.00
12 Months (10 issues)	£50.00	£58.00	£33.00	12 months (10 issues)	£56.00	£64.00	£39.00

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

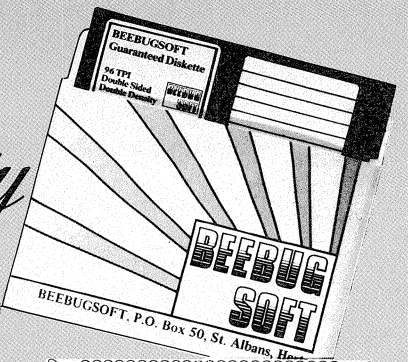
Cassette subscriptions can be commuted to $5\frac{1}{4}$" disc subscription on receipt of £1.70 per issue of the subscription left to run. ($3\frac{1}{2}$" disc £2 per issue)

All subscriptions and individual orders to:

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

The Ultimate in Quality & Reliability

BEEBUG DISCS



Our blank discs are produced for Beebug by one of the worlds leading disc manufacturers in a special dust free environment. Every single disc is then individually tested for data reliability and any sub-standard units are instantly rejected.

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	£9.90	£9.40	0657
25	£23.00	£21.85	0661
50	£48.00	£45.60	0665

80 Track Double Sided Quad Density

	Price	Members Price	Order Code
10	£10.90	£10.35	0660
25	£29.00	£27.55	0664
50	£53.00	£50.35	0668

Can you afford to trust your data to anything less reliable? Many unbranded discs on the market are produced from low quality media rejected by the major manufacturers.

FREE

Memorex Plastic Library Case

Supplied with each purchase of 10 discs.

FREE

New LOCKABLE Storage Box

(Normal Price £13.80)

With special carrying handle with each purchase of 25 or 50.

All Prices Include VAT.

UK Carriage 10 £1.00, 25/50 £3.75.
Overseas send same price including UK carriage & VAT (To cover extra postage).

Official Orders Welcome

Hotline for Access/Visa telephone orders 0727 40303 (24 hours).

Name_____

Address_____

Please send me ____ (qty) ____ (stock code) at £____ (unit price) ____ plus £____ (postage).

I enclose a cheque value £____

Please debit my Access/Visa with £____

Card No.

Expiry ____/____

Membership No. (To claim members price) ____

BEEBUG

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

Telephone: 0727 40303