INSTALLING A
HELP SYSTEM
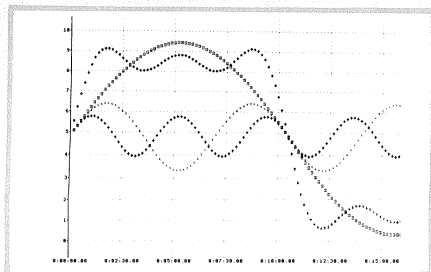
BEEBUG

FOR THE BBC MICRO

TURN YOUR BEEB INTO A CHART RECORDER

Chart Recorder

Psion Organiser

Fancy Lettering

FANCY LETTERING

by Thomas Williamson

Different *styles*

and *SIZE*s

for the BBC micro

Master and Compact

Real-Time Graphics

Flitty Flea

Spell Master

BEEBUG ON MICRONET

The BEEBUG pages on Micronet are now fully operational and we invite all members with access to this system to take a look. BEEBUG is fully signposted from within Micronet, but if you want to go direct to the BEEBUG front page, just key *800909# (as stated last month). Remember, of course, that these pages are not restricted to BEEBUG members only — all users of Micronet have access. Thus we have included a full description of what BEEBUG is and what it has to offer members, with the opportunity of joining BEEBUG through a response frame.

You will also find that a large part of our mail order catalogue is on-line, and you can order any goods you wish using your credit card. Do remember to quote your BEEBUG membership number to ensure that you get the discount to which you are entitled. You may also renew your membership on-line (*80090971#).

We shall also be providing information about the current and forthcoming magazines, and this medium will allow us to provide up-to-date information on publication and distribution dates. You will also find extracts from past and present issues of the magazine, including hints and tips, and programs for downloading to your micro.

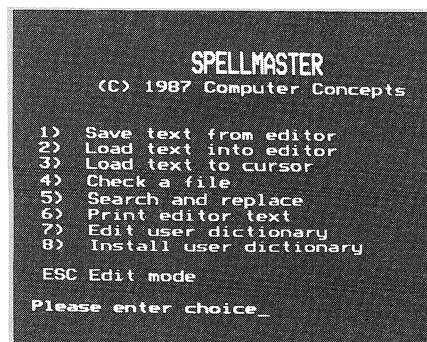We are naturally interested in your reactions and opinions regarding this new venture for BEEBUG. You can send mailboxes if you wish direct to me, Mike Williams, on my own Micronet number 819991213, though I cannot guarantee to reply to every single mailbox I receive. Any particularly interesting comments may be published on the Postbag page of a future issue of BEEBUG, unless you indicate otherwise.

MAGAZINE CASSETTE/DISC

From the start of Volume 6 (which starts with the May issue) all the magazine programs will also be available on 3.5" disc as well as 5.25" disc and cassette. To coincide with this we are offering a special deal on all new disc and cassette subscriptions taken out for the start of Volume 6. Full details are in this issue.

PROGRAM/REVIEW CLASSIFICATION

We hope that the new classification symbols for programs and reviews clarify matters with regard to the variety of Acorn systems. The complete set of icons is shown below. These show clearly the valid combinations of machine (version of Basic) and filing system for each item, and Tube compatibility. A single line through a symbol indicates partial working (normally just a few changes will be needed); a cross shows total incompatibility. Reviews do not distinguish between Basic I and II.

Computer System		Filing System	
Master (Basic IV)		ADFS	
Compact (Basic IV)		DFS	
Compact (Basic VI)		Cassette	
Model B (Basic II)		**Tube Compatibility**	
Model B (Basic I)		Tube	
Electron			

Down in the Dumps

Latest release from Design Dynamics, publishers of CAD package Interactive 3D (see BEEBUG Vol.4 No.10 & Vol.5 No.8) is a "Mode-00 Dump". Described as more of a plotter driver than a screen dump, Mode-00 Dump offers an amazing 640 x 512 resolution on most dot-matrix printers. The Mode-00 Dump costs £12.95 inclusive direct from Design Dynamics, and this together with Interactive 3D is currently available on special offer for £20.00 inclusive. Design Dynamics are on 0525-402447.
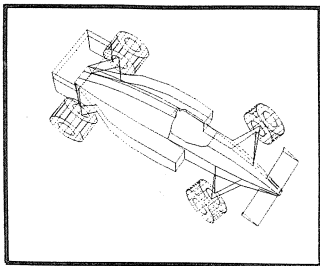
Software for the Compact

The availability of educational software on the Compact takes a big step forward with the release of no less than 98 educational programs by Ludinski Computer-Assisted Learning. Micro Maths, Micro English and Micro French offer complete courses of 24 programs taking children from the age of eight eventually up to GCSE level. Mega Maths is a step-by-step A-level course while Animated Arithmetic is designed for infants.

Both Micro English and Micro French include audio tapes, and the English course can be used for EFL teaching as well. Prices are around £24 and LCL are on 0491-579345.
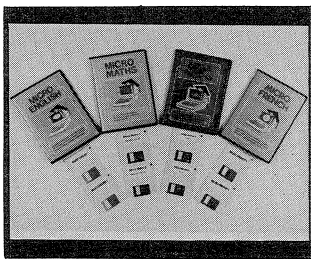
Art on your Heart

If you want to see your latest computer graphics masterpiece emblazoned across your chest, then Banbury Business Systems can help. A special printer ribbon allows the image to be output on paper which can then be ironed onto a T-shirt. Up to 100 images are possible with each ribbon which costs £15.00 for black and £23.00 for colour direct from Banbury. Contact Mrs Susan Crawford on 0295-720812.

Virtual Instruments

Perhaps taking a leaf out of the BEEBUG magazine (see Oscilloscope Vol.5 No.5 & 6 and Chart Recorder Vol.5 No.10), the Design Consultancy are offering the OASIS Virtual Instrument System. This includes a precision Data Converter and software to turn your Beeb into a sophisticated multi-meter, oscilloscope, chart recorder, data logger and plotter. Further information on the OASIS VIS (priced at £399 plus VAT) can be obtained on 0603-747877.

Play it again Sam

Vine Micros have announced the long awaited Master Replay version of their popular tape-to-disc utility. Even better than the original BBC 'B' Replay, the new version can transfer any memory from 32K right up to 128K shadow RAM. Master Replay costs £41.95 complete, or £36.95 for use with ROMBOARD'3'. Vine Micros can be contacted for further information on 0304-812276.

Long Live the Electron

Acorn may have lost interest in the Electron, but the good news for all Electron users is that Advanced Computer Products has obtained an agreement from Acorn to continue manufacture of the Electron Plus 1 interface under licence. This uses all the original Acorn moulds and tooling. ACP are also handling all servicing arrangements for the Plus 1 as well as looking after other Electron peripherals. For further information contact ACP on 0276-76545.

Pascal for Beeb and Master

The popular Oxford Pascal Compiler is still available, and now there is a separate version for the Master series. Oxford Pascal is a full Pascal compiler with BBC graphics and sound extensions. Compiled programs can be run independently of the Pascal system. Oxford Pascal for the BBC 'B' costs £49.95 on disc, £39.95 on cassette; the Master version costs £59.95 on disc only. For more information contact Systems Software (Oxford) Ltd on 0865-54195.

Teletext Across the Net

Now available from BBC Telesoftware is an Econet Teletext Server system. Once installed on an Acorn Econet network system, any station on the network can access the full range of Teletext facilities including display of information pages, downloading of software and interactive access to Teletext data pages. The Teletext server can even be accessed across a 'bridge' connecting two or more network systems together.

The software is available free through the BBC's Telesoftware service and will be broadcast at frequent intervals. For more information on this and all aspects of the BBC's Telesoftware service contact Jeremy Brayshaw on 01-576 0563.

Tree Searches

Anyone interested in genealogy and family trees should consider the "Family History System" now published by Micro Aid at £20.52 inc. VAT and p & p. This is a revised version of a program previously called Easytree, and existing users can upgrade by returning the original disc to Micro Aid at 25 Fore Street, Praze, Camborne, Cornwall TR14 0JX together with £5.00, or contact Micro Aid on 0209-831274.

Good Viewing

Now that View is supplied with the Master and Compact there seems to be growing support for this word processor and the other members of the View family (ViewSheet, View-Store, ViewSpell etc). "Mastering View, ViewSheet and ViewStore" is the title of a new book by Clive Williamson published by Sigma Press at £12.95. The book is distributed by John Wiley & Sons who can be contacted on 0243-784531. We expect to review this book in the near future.

MASTERING **View, ViewSheet and ViewStore**
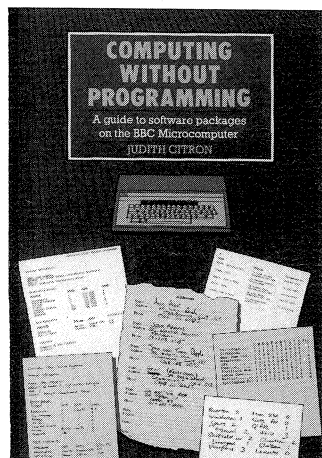
Clive Williamson

SIGMA
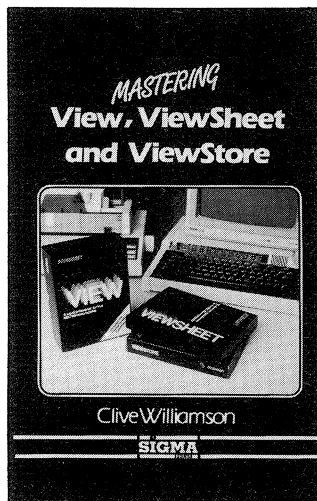
Increase your Wordpower

Users of the Wordpower word processor by Ian Copestake will be interested in a new product from the same source. Power Font gives you a choice of 219 different characters, including accented and scientific characters. Even the Russian alphabet is claimed to be easy. Power Fonts is supplied on disc, and works with any BBC micro, and in any mode except mode 7. The cost is £24.50 and Ian Copestake is on 04867-4755.

Computing without Programming

This is the title of a new book by Judith Citron published by Chapman and Hall at £9.50. The book is a guide to choosing and using three widely used applications, word processing, spreadsheets, and databases. In each category the book describes two particular packages in some detail, including BEEBUG's own Masterfile II and Quickcalc. Other packages described are Wordwise Plus, Edword 2, Quest and Ultracalc 2. There are obvious omissions here (no Acorn products such as View or ViewSheet for example), and the book is biased towards the educational user. However, there is much useful information for everybody on the particular packages covered. For more information contact Associated Book Publishers on 0264-62141. This book is available from BEEBUG at 5% member's discount.

COMPUTING WITHOUT PROGRAMMING
A guide to software packages on the BBC Microcomputer
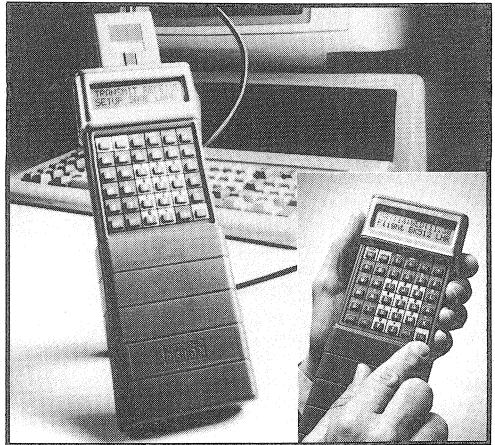JUDITH CITRON

Let PSION organise your BBC Micro

David Graham wires his Beeb up to a Psion Organiser II, and reports on the ensuing dialogue.

Product : Psion Organiser II
Supplier : Psion Ltd, Psion House,
Harcourt St, London W1H 1DT.
Tel. 01-723 9408
Prices : Organiser II from £99.50
Datapacks from £12.95
RS232 Link £47.95
BBC micro connector £12.95
All prices include VAT.

First of all, I must try to answer the question which you are probably now asking. Why put a review of the Psion Organiser in BEEBUG magazine? The answer is simple. The Organiser can be used to put some of the BBC Micro's power in your pocket. Programs (written in OPL) and standard datafiles from software such as Wordwise, View, etc, can be downloaded from the BBC micro into the Organiser for instant access. As a practical test, we downloaded the whole of the BEEBUG Retail price list into the Organiser from a modified Wordwise file. Pressing "F" (for Find) from the main menu of the Organiser, followed by a few characters of text was enough to locate the price of any item in a fraction of a second - no more fumbling around in the BEEBUG price list. And, of course, the file can be updated on the Beeb, and reloaded as often as you wish.

This gives an idea of just one area of application of the Organiser. Before taking a closer look at the way in which the Organiser can be linked to the Beeb via its RS232 port, I will give a run down of the general features of the device itself.

The Organiser is a small hand-held computer using conventional technology. It has an alphanumeric keyboard (arranged alphabetically), and a two line by sixteen character LCD display which is automatically scrolled to display longer lines. The machine is based on the CMOS 6303X microprocessor, which is a member of the 6800 family. It runs at 1MHz and uses 24 or 32K of internal ROM and up to 16K of RAM. Plug-in Datapacks, containing EPROMs of 8K and upwards are used for permanent storage. Being EPROMs, they need to be wiped clean from time to time by exposure to ultraviolet light to get rid of unwanted data.

The Organiser comes with a considerable amount of built-in applications software, and in addition there are plug-in applications packs (Finance, Maths and Spelling Checker currently), together with a fully supported high level language called OPL. The major on-board applications include a database with a fast Search option, and a diary which can take text entries for appointments at half-hourly intervals up to the year 2000, and can sound an alarm up to one hour before each appointment is due. In addition there is a set of eight alarms, which can be set to repeat at hourly, daily or weekly intervals, but which unfortunately cannot be set more than one week ahead. The device also provides an accurate clock and calendar, and a calculator which can call a series of inbuilt functions, plus any which the user has set up in OPL.

The OPL programming language with its 112 commands and functions is both powerful and fast, offering many of the

facilities of BBC Basic. BBC users will particularly appreciate its structured nature, and its heavy reliance on procedures and functions. The most noticeable differences from BBC Basic are that there are no line numbers – GOTO freaks should not go into a state of shock, labels are provided – and the language is compiled rather than interpreted. You type in your program, and if it is syntactically correct, it is compiled, and once this version is saved, it can be run at the press of a key.

COMMUNICATIONS

The Organiser has an add-on RS232 link which contains its own comms software. This is downloaded into the Organiser on switch-on, and allows you to transmit or receive complete program or data files from another computer. In its Terminal mode, the Organiser can operate as a stand-alone terminal in which characters typed on its keyboard appear at the remote computer, and vice versa. This makes it possible to access Telecom Gold on the Organiser, given a suitable modem.

The Organiser's well written comms software gives complete control over communications protocol, baud rate and so on. This is all adjusted using cursor keys, from the SETUP option on the Comms menu. Once you have set the protocol, you may save it to a file which can be loaded in again at any time. A total of 13 parameters may be set in this way, including the handshaking protocol. XON/XOFF, XMODEM and RTS/CTS are all supported, the latter being used for direct communication with the Beeb.

Transferring files between Organiser and Beeb is child's play. It just involves writing a short program on the Beeb to issue a couple of FX calls to set up the RS423 port for send or receive, and then VDU statements can be used to send characters to the Organiser, or the GET function to receive them. In the very simplest case, the following short program will receive data from the Psion and display it more or less legibly on the Beeb's screen:

```
10 *FX2,1        40 A=GET
20 *FX7,7        50 VDUA
30 REPEAT        60 UNTIL A=26
```

For the purposes of this test, the data may be sent either in the Organiser's Transmit or Terminal modes, but in either case, the handshaking protocol should be set to RTS/CTS. If you are transferring files between the Beeb and the Organiser, rather than just testing out the principle, you will need to introduce some minor sophistications. In particular, control codes sent and received by the Organiser will need processing. In fact the Organiser gives you full control over the codes used as End of Field and End of File markers, the defaults being VDU9 and VDU26; and you may also determine whether line feeds are to accompany carriage returns in both Send and Receive modes. It is therefore very easy to customise the Organiser to suit your own requirements, though if you are communicating with the Beeb, you can just as easily write your send and receive software to fit in with the defaults on the Organiser.

The Receive option on the Organiser can be used to receive program files from the Beeb just as easily as data files. But to be of any use, the program to be transferred must be written in ASCII format in the OPL language. A wordprocessor such as Wordwise, or a text editor like the Master Editor are ideal for this purpose. Once entered at the Beeb's keyboard, the program can then be printed out, and checked over before transferring to the Organiser for testing. This gives great benefit since it is much easier to type in a program on the Beeb than it is using the Organiser's small keyboard and two line display. I tested this out, by typing a small OPL program into Wordwise, and saving it away to disc. It transferred without any fuss, and ran first time.

CONCLUSION

The Organiser is a fascinating device, and the Beeb and the Organiser work well together. The Beeb can support the Organiser in providing a medium for program development, and an easy way of backing up Organiser files. The Organiser on the other hand can be seen as extending the computing power of the Beeb into the field. Specific applications programs developed on the Beeb in OPL can be instantly accessed in locations remote from the home or office, and large databases can be downloaded from the Beeb for instant access. If you need kilobytes in the pocket, this seems a good route to travel, though your pocket must not be insubstantial – in more ways than one.

FANCY LETTERING

Multi-coloured, shaded, 3D lettering and more are all possible with the routines described by Thomas Williamson.

Have you ever wanted to create professional looking title screens for your programs? This utility, with its choice of multi-coloured lettering styles and sizes, could be just what you need.

Characters can be displayed on the screen in any one of three different styles, '3D', 'outline' and 'normal'. Letters in each style may be of any size and in any position, and either upright or italic. The three styles are produced by three separate procedures, PROC3D, PROCOL and PROCN (3D, outline and normal), and these all call one further procedure PROCP which does all the hard work.

To see what can be achieved, type in the demonstration program listed, but don't initially enter all the DATA statements, just lines 10320, 10330 and 10340 will suffice. If you select mode 1 and set C1=1, C2=2 and C3=3 (all in immediate mode), you can test out the program by typing:

PROC3D(100,100,"ABC",3,0)

This will display the letters "ABC" in fancy lettering on the screen. You can also try entering the other two procedures, also in immediate mode, to test their effect, provided you stick to the letters "ABC". Once all is working, the rest of the data may be typed in. When run, the program will then provide a complete demonstration of the various styles and sizes possible.

USING THE LETTERING PROCEDURES

The three main lettering procedures PROC3D (3D), PROCN (normal) and PROCOL (outline) all use the same parameters:

Q% - Left graphics co-ordinate (X)
R% - Bottom graphics co-ordinate (Y)
A$ - Text to print

S% - Size of text (1 to n)
I% - Italic flag (0-no, 1-yes)

To incorporate such lettering effects in your own programs you will need the three lettering procedures, the procedure PROCP, and the DATA statements which define the characters you are using. These definitions are the values needed for PLOT commands to draw each character. You can also select the colours for the text, by changing the values of C1, C2 and C3 (see the procedure PROCdemo).

The demo program listing includes the data for all the displayable characters (ASCII 33 to 126), but in practice you should include only those that you need in any one program. To assist in this process, the characters are arranged in the DATA statements as follows:

Characters				Lines	
33 to 47	(! to /)		10000 to 10140	
48 to 57	(0 to 9)		10150 to 10240	
58 to 64	(: to @)		10250 to 10310	
65 to 90	(A to Z)		10320 to 10570	
91 to 96	([to £)		10580 to 10630	
97 to 122	(a to z)		10640 to 10890	
123 to 126	({ to ~)		10900 to 10930	

If you do omit any of the data lines, do NOT renumber the remainder as errors will result.

```
10 REM Program Fancy Letters
20 REM Version B0.3
30 REM Author  T.Williamson
40 REM BEEBUG  April 1987
50 REM Program subject to copyright
60 :
100 MODE1
110 ON ERROR GOTO 1450
120 PROCdemo
```

```
  130 END
  140 :
 1000 DEFPROCdemo
 1010 VDU19,2,4;0;
 1020 C1=1:C2=2:C3=3
 1030 PROC3D(32,800,"FANCY LETTERING",3,
0)
 1040 PROC3D(32,780,"_____",3,
0)
 1050 PROC3D(100,612,"by Thomas Williams
on",2,0)
 1060 PROCOL(0,400,"Different",3,0)
 1070 PROCN(732,400,"styles",3,1)
 1080 PROCN(200,200,"and",3,0)
 1090 FORA%=5TO1STEP-1
 1100 PROC3D(1000-A%*112,200-A%*12,MID$(
"SEZIS",A%,1),A%,1)
 1110 NEXT
 1120 ENDPROC
 1130 :
```



```
 1140 DEFPROC3D(Q%,R%,A$,S%,I%)LOCALA%,B
%,W%
 1150 Q%=Q%+(LENA$-1)*24*S%
 1160 FORB%=LENA$TO1STEP-1:A%=ASCMID$(A$
,B%,1)
 1170 IFA%=32GOTO1190
 1180 FORW%=-S%*8TO-4STEP4:GCOL0,C1:PROC
P(W%,W%):GCOL0,C2:PROCP(W%+4,W%):NEXT:GC
OL0,C3:PROCP(0,0)
 1190 Q%=Q%-S%*24:NEXT
 1200 ENDPROC
 1210 :
 1220 DEFPROCOL(Q%,R%,A$,S%,I%)LOCALA%,B
%
 1230 FORB%=1TOLENA$:A%=ASCMID$(A$,B%,1)
 1240 IFA%=32GOTO1260
 1250 GCOL0,C3:PROCP(0,-4):PROCP(-4,0):P
ROCP(0,4):PROCP(4,0):GCOL0,C1:PROCP(0,0)
 1260 Q%=Q%+24*S%:NEXT
 1270 ENDPROC
 1280 :
```

```
 1290 DEFPROCN(Q%,R%,A$,S%,I%)LOCALA%,B%
 1300 GCOL0,C3
 1310 FORB%=1TOLENA$:A%=ASCMID$(A$,B%,1)
:IFA%=32GOTO1330
 1320 PROCP(0,0)
 1330 Q%=Q%+24*S%:NEXT:ENDPROC
 1340 :
 1350 DEFPROCP(T%,U%)LOCALC%,P%,X%,Y%
 1360 MOVEQ%+T%,R%+U%:RESTORE(A%*10+9670
):READB$
 1370 FORC%=1TOLENB$-3 STEP4:X%=EVAL("&"
+MID$(B$,C%,2)):Y%=EVAL("&"+MID$(B$,C%+2
,2))
 1380 IFX%>128 X%=X%-256
 1390 IFY%>128 Y%=Y%-256
 1400 O%=X%:X%=X%MOD100:IFX%<>O% P%=0:EL
SEP%=1
 1410 IFI%=1 PLOTP%,(X%+Y%/2)*S%,Y%*S%:E
LSEPLOTP%,X%*S%,Y%*S%
 1420 NEXT
 1430 ENDPROC
 1440 :
 1450 ON ERROR OFF
 1460 MODE 7:IF ERR=17 END
 1470 REPORT:PRINT" at line ";ERL
 1480 END
 1490 :
10000 DATA 6D0000016405001E
10010 DATA 6B24FDFA6A000306
10020 DATA 660006246C00FADC6A0CF000660C1
000
10030 DATA 6C0000002466DC00246BF8FE04FC02F
C00FCFEFEFC00FC02FC04FE040004FE02FC00FCF
EFCFCFEFC00FC02FE04
10040 DATA 12248CFE00FF72E20001
10050 DATA 7600F218FE060204040202000004FE0
2FC00FEFEFECF6F6FEFEC00FC02FC04FE060004020
202
10060 DATA 6B1E0306
10070 DATA 7000FC02FE0400180204040
2
10080 DATA 6A0004020204000018FE04FC02
10090 DATA 6D06001896FA0CF4640CF4F4
10100 DATA 6D06001896F40C00
10110 DATA 6AFC040202040004
10120 DATA 68120E00
10130 DATA 6D0000001
10140 DATA 1224
10150 DATA 6A00060004020204000018FE04FC02F
A00FCFEFEFEC00E802FC04FE
10160 DATA 6A00006000990000024FFFEFEFF
10170 DATA 641E0204040206000FE02FC00FAF
EFCF2F2FEFEC00FE1200
10180 DATA 641E0204040206000FE02FC00FAF
EFCFCFE04FE02FC00FAFEFECFCFEFEFA00FC02FE04
10190 DATA 700000024F4E81200
10200 DATA 7624E000EE020404020600004FE0
2FC00F4FEFCFCFEFA00FC02FE04
10210 DATA 761EFE04FC02FA00FCFEFEFEFC00E80
2FC04FE06000402020400CFE04FC02FA00FCFEF
EFC
```

```
10220 DATA 64241200EEDC
10230 DATA 6A0006000402020400006FE04FC02F
A00FC02FE0400006020404020600004FE02FC00FAF
EFCFCFE9600FCFEFEFC00FA02FC04FE
10240 DATA 640602FC04FE0600040202040018F
E04FC02FA00FCFEFEFC00F402FC04FE0600040202
204
10250 DATA 7008000264040002
10260 DATA 6AFC0402020400004640400002
10270 DATA 7600EE121212
10280 DATA 660F10006406F000
10290 DATA 1212EE12
10300 DATA 641E02040402060004FE02FC00FAF
EFCFCFEFCFEFEFC00FA64FA0001
10310 DATA 7000FA00FC02FE040018020404020
60004FE02FCFDFDFE02001464FEFE02FC00F
EFE00EE02FE04000202
10320 DATA 001E02040402060004FE02FC00E26
412EE00
10330 DATA 0C00040202040006FE04FC02F4006
E000402020400006FE04FC02F60000DC
10340 DATA 7606FEFCFCFEFA00FC02FE0400180
2040402060004FE02FC
10350 DATA 0C00040202040018FE04FC02F4000
0DC
10360 DATA 12008A120C006A12EE0000DC
10370 DATA 64120C006A12EE0000DC
10380 DATA 76000012FA006AF4FEFCFCFEFA00F
C02FE0400180204040206004FE02FC
10390 DATA 002464EE1200641200DC
10400 DATA 6A000600990000246700FA00
10410 DATA 640602FC04FE06000402020400001E
10420 DATA 002464E81218F4F00CEC
10430 DATA 12008A000024
10440 DATA 002409DC092400DC
10450 DATA 092412DC0024
10460 DATA 640602FC04FE0600040202040018F
E04FC02FA00FCFEFEFC00E8
10470 DATA 00240C0004FE02FC00FAFEFCFCFEF
400
10480 DATA 640602FC04FE0600040202040018F
E04FC02FA00FCFEFEFC00E8700006FA
10490 DATA 00240C0004FE02FC00FAFEFCFCFEF
4006D0009EE
10500 DATA 640602FC04FE0600040202040006F
E04FC02FA00FC02FE040006020404020600004FE0
2FC
10510 DATA 64241200930000DC
10520 DATA 642400E202FC04FE0600040202040
01E
10530 DATA 642409DC0924
10540 DATA 642405DC042405DC0524
10550 DATA 12248A0012DC
10560 DATA 6D00001209128A0009EE
10570 DATA 64241200EEDC1200
10580 DATA 7000FA0000240600
10590 DATA 642412DC
10600 DATA 6A0006000024FA00
10610 DATA 671E060606FA

10620 DATA 99001800
10630 DATA 7602FCFEF900FF01FF02FE01FEFFF
FFE01FF020002010102006FE09000602060602040
40204FE02FC8BF70F008CFC0F00
10640 DATA 641202040402040004FE02FC00F00
2FE9A06FEFCFCFEFC00FC02FE04000402040402040
40004FE02FC
10650 DATA 642400E202FC04FE0600040202040
00CFE04FC02FA00FCFEFEFC
10660 DATA 7612FE04FC02FA00FCFEFEFC00F40
2FC04FE060004020204
10670 DATA 762400DC6406FEFCFCFEFA00FC02F
E04000C02040402060004FE02FC
10680 DATA 640A0C00040202040002FE04FC02F
A00FCFEFEFC00F402FC04FE0600040202040
10690 DATA 640000001E02040402060004FE02FC8
AF40C00
10700 DATA 6A04FCFEFEFC00FC02FC04FE06000
40202040004FE04FC02FA00FE0200020202FC02F
E040002020404020400004FE02FC00FEFEFCFCFEF
C006E0A0202
10710 DATA 002464EE02040402060004FE02FC0
0EE
10720 DATA 6B0004009A000012FE0066050001
10730 DATA 6AF404020204001864050001
10740 DATA 002464E2121293F709F1
10750 DATA 7000FC02FE04001E
10760 DATA 001501020201003FE02FD00FC64040
203030202FF01FE00EB
10770 DATA 001864FA02040402060004FE02FC0
0EE
10780 DATA 640602FC04FE06000402020400CF
E04FC02FA00FCFEFEFC00F4
10790 DATA 64F4002464FA02040402060004FE0
2FC00F4FEFCFCFEFA00FC02FE04
10800 DATA 78F4FE00002464FAFE04FC02FA00F
CFEFEFC00F402FC04FE060004020204
10810 DATA 001864FA02040402060004FE02FC
10820 DATA 640602FC04FE06000402020 4FE04F
C02FA00FC02FE040204040206004FE02FC
10830 DATA 6A2400E202FC04FE040202048D120
F00
10840 DATA 641800EE02FC04FE06000402020 46
41200E8
10850 DATA 641809E80918
10860 DATA 641800EB01FD03FF0302020300046
4FC02FD03FE0301010103 0015
10870 DATA 12188A0012E8
10880 DATA 641800EE02FC04FE06000402020 46
41200E2FEFCFCFEFA00FC02FE04
10890 DATA 64181200EEE81200
10900 DATA 7624FCFEFEFC00FAFEFCFCFE04FE0
2FC00FA02FC04FE
10910 DATA 6D00000F6406000F
10920 DATA 642404FE02FC00FA02FC04FEFCFEF
EFC00FAFEFCFCFE
10930 DATA 662000002030203FE02FE03FE03020
002
```

Installing a Help System

If you need a Help system for your machine then look no further. Alan Webster's program provides fast and efficient help on any subject, and you can use almost any word processor to generate your own text.

This article and accompanying programs will allow you to set up and use a random access 'HELP' facility, accessed via a new star command, *HLP. The assembled machine code will access an ASCII text file on disc, and search via an index for the specified string and display the help text associated with it.

Type in the 'Help' program and save it before running. If the program assembles correctly, a message is displayed. Press f0 to save the machine code with the name QUERY.

Next, create your help text using any suitable word processor (e.g. View or Wordwise; but spooled text with Interword). Your text should be in ASCII format and comply with the following conventions:

- Each heading (or command) MUST start with a tilde character (~) and end with a carriage return.
- The descriptive text for each heading should NOT contain the tilde character.
- The whole text should end with two tilde characters (~~) as an end-of-file marker.

The tilde character can be replaced by any character of your choice simply by changing line 120 in the main program.

You will then need to type in the 'Convert' program, and run this. Enter the name of your text file in response to the first filename, and enter a second filename for the output file. The main machine code is programmed to look for the file 'H.DATA', so use this name (or change line 130). The Convert program will then search your text for each heading, and create an index at the start of the text file for use with the QUERY program.

To install the help system, make sure that both the program QUERY and the converted text file H.DATA are on the same disc. *RUN QUERY will then activate the *HLP command. You should then find that typing *HLP <Return> displays a list of all headings in your help text.

To display the help text for any individual heading or command, simply type *HLP <heading> and press Return. The heading can be abbreviated to its first two characters (no full stop needed). For example, to display the text under a heading of 'Database', you could type either of:

   *HLP Database
   *HLP Da

(and both the 'HLP' and the heading can be in upper or lower case).

```
  10 REM Program Help
  20 REM Version B0.5
  30 REM Author  Alan Webster
  40 REM BEEBUG  April 1987
  50 REM Program subject to copyright
  60 :
 100 ON ERROR GOTO 2190
 110 MODE 7
 120 marker=ASC("~")
 130 file$="H.DATA"
 140 PROCassemble(&900)
 150 X%=0:Y%=&C:$&C00=A$:CALL &FFF7
 160 END
 170 :
1000 DEFPROCassemble(Q%)
1010 FOR pass=0 TO 3 STEP 3
1020 P%=Q%:[OPT pass
1030 .start
1040 STA &70:STX &71:STY &72:LDX#&FF
1050 .loop:LDY#&FF:.inner:INY
1060 LDA(&71),Y:CMP#&2A
1070 BEQ inner:INX:AND#&DF
1080 STA &73:LDA table,X
1090 BEQ zerobyte:CMP &73:BEQ inner
1100 .again
1110 INY:INX:LDA table,X
1120 BNE again:INY:INX
1130 LDA table,X:CMP#&FF
1140 BEQ exit:INY:INX:JMP loop
1150 .exit
1160 LDA &70:LDX &71:LDY &72
1170 CLI:JMP (&230)
1180 .zerobyte
1190 INY:INX:LDA table,X
1200 CMP#&FF:BEQ exit:STA &80
1210 INY:INX:LDA table,X
1220 STA &81:DEY:DEY:JMP(&80)
1230 .table
```

```
1240 OPT FNs("HLP"):OPT FNb(0)
1250 OPT FNw(query):OPT FNb(0)
1260 OPT FNb(255)
1270 .name
1280 OPT FNs(file$):OPT FNb(13)
1290 .query
1300 LDA#&00:STA &76:LDA(&71),Y
1310 CMP#&0D:BEQ allof
1320 .spaces
1330 INY:LDA(&71),Y:CMP#&20
1340 BEQ spaces:STY &74
1350 .allof
1360 LDA(&71),Y:CMP#&0D
1370 BNE openfile:INC &76
1380 .openfile
1390 LDA#&C0:LDX#name MOD 256
1400 LDY#name DIV 256:JSR &FFCE
1410 BNE filefound:BRK:OPT FNb(214)
1420 OPT FNs("File '"+file$+"' not foun
d"):BRK
1430 .filefound
1440 STA &73:LDA &76:BEQ find
1450 .repeat
1460 JSR getbyte:BEQ skip:BMI endoflist
1470 JSR printch:JMP repeat:.skip
1480 JSR &FFE7:JSR getbyte
1490 JSR getbyte:JMP repeat
1500 .endoflist
1510 JSR &FFE7:JMP close
1520 .find
1530 LDY &74:DEY:LDX#&00
1540 .findloop
1550 INX:INY:STY &75:JSR getbyte
1560 CMP#&00:BEQ found:CMP#&FF
1570 BEQ endfile:LDY &75:STA &77
1580 LDA(&71),Y:AND#&DF:CMP &77
1590 BEQ findloop
1600 .getnext
1610 JSR getbyte:CMP#&00
1620 BNE getnext:CPX#&03
1630 BEQ found:JSR getbyte
1640 JSR getbyte:JMP find
1650 .endfile
1660 JMP endoffile
1670 .found
1680 JSR getbyte:STA &77
1690 JSR getbyte:STA &78
1700 LDA#&00:STA &79:STA &7A
1710 LDX#&77:LDY &73:LDA#&01
1720 JSR &FFDA:JSR getbyte
1730 .printloop
1740 JSR getbyte:CMP#marker
1750 BEQ endprint:JSR printch
1760 JMP printloop
1770 .endprint:JSR &FFE7:JMP close
1780 .getbyte:LDY &73:CLC:JSR &FFD7
1790 BCS endoffile:RTS
1800 .endoffile
1810 LDA &76:BNE close:LDY#&FF
1820 .nosuch
```

```
1830 INY:LDA notxt,Y:JSR &FFEE
1840 BNE nosuch:JMP close
1850 .notxt
1860 OPT FNs("No such reference - try *
HLP")
1870 OPT FNb(13):OPT FNb(10):OPT FNb(0)
1880 .close
1890 LDA#&00:LDY &73:JSR &FFCE:RTS
1900 .execute
1910 LDX #start MOD 256
1920 LDY #start DIV 256
1930 CPX &208:BNE store:CPY &209
1940 BEQnoway
1950 .store
1960 LDA &208:STA &230:LDA &209
1970 STA &231:STX &208:STY &209
1980 .noway:RTS
1990 .printch
2000 CMP#&0D:BNE other:JSR &FFEE
2010 LDA#&0A:JMP vdu
2020 .other:CMP#&20:BCSvdu:LDA#&20
2030 .vdu:JSR &FFEE:RTS
2040 ]
2050 NEXT
2060 A$="*SAVE Query "+STR$~Q%+" "+STR$
~P%+" "+STR$~execute+" "+STR$~Q%
2070 PRINT''"To save the machine code t
ype:"'A$:A$="K.0 "+A$+"|M"
2080 ENDPROC
2090 :
2100 DEFFNb(A%)
2110 ?P%=A%:P%=P%+1:=pass
2120 DEFFNw(A%)
2130 !P%=A%:P%=P%+2:=pass
2140 DEFFNs(A$)
2150 $P%=A$:P%=P%+LENA$:=pass
2160 :
2170 ON ERROR OFF:MODE7:IF ERR=17 END
2180 REPORT:PRINT" at line ";ERL
```

Convert Program

```
  10 REM Program Text Convert
 100 MODE7:DIM index &200,text &4000
 110 INPUT''"File to convert :"F$
 120 INPUT''"File to save as :"C$
 130 A$="LOAD "+F$+" "+STR$~text
 140 X%=0:Y%=9:$&900=A$:CALL&FFF7
 150 PRINT':A%=text:I%=index
 160 REPEAT:R%=?A%:A%=A%+1
 170 UNTIL R%=ASC("~")
 180 IF?A%=ASC("~")THEN 250
 190 A$="":R%=?A%:REPEAT
 200 A$=A$+CHR$R%:A%=A%+1:R%=?A%
 210 UNTILR%=13
 220 $I%=A$:I%=I%+LENA$:?I%=0
 230 I%=I%+1:!I%=(A%-index)
 240 I%=I%+2:PRINTA$:GOTO160
 250 VDU7:?I%=-1:A$="SAVE "+C$+" "+STR$
~index+" "+STR$~A$
 260 X%=0:Y%=9:$&900=A$:CALL &FFF7
```

Turn your BEEB into a Chart Recorder
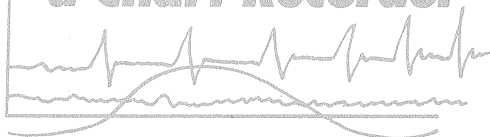
David Peckett, author of the popular Oscilloscope article in Vol. 5 No. 5, has not been idle. This time he shows how to turn your Beeb and printer into a four-channel chart recorder.

The October and November 86 issues of BEEBUG (Vol.5 Nos.5 & 6) contained a two-part article which showed how the Beeb could be used as a digital storage oscilloscope. The interest provoked by that article has led to the development of this new application, a four-channel chart recorder.

A chart recorder, for those who have not met one, is a device which inputs electrical signals and plots them in graphical form on long paper charts. Usually, several different inputs can be shown at once, and the instrument is normally used to monitor relatively slowly-changing signals (no more than a few tens of Hz). Heart monitors and lie detectors often send their outputs to chart recorders, and in industry they are used, for example, to monitor how temperature, pressure, etc change over periods varying from minutes to weeks.

Like the oscilloscope, this program allows you to build a chart recorder without adding anything to your computer. You must, however, have a printer, and the program assumes that it is an Epson or Epson-compatible dot-matrix machine – a daisy-wheel printer would be unsuitable here. Using Epson codes should suit the vast majority of printers but, unfortunately, there are some which will misbehave; it would not be too difficult to adapt the program to these, however.

When the program runs, it reads the analogue port, and saves the data to memory, unless there is too much. Alternatively, data can be saved directly to disc (the best option) or to cassette. You can also do a few things without any additional storage.

Like the oscilloscope, the recorder needs you to connect some sort of transducer(s) to the analogue port to give you something to measure (running the program with nothing connected will give a 'noise' pattern). For instance, a simple strain gauge network could measure small movements; though for testing, joysticks may be used.

The program is in two parts. The main section, this month, allows you to read and plot data, and to save the data to disc as it is read. Next month, we'll add routines to give a more flexible way of saving data, to re-plot data from memory or disc, and to use the Beeb's 'star' commands. I'll also provide some technical details of how the program works, and how you might adapt it for other printers.

Although this month's program appears to offer these extra facilities, they are only represented by dummy routines. If you
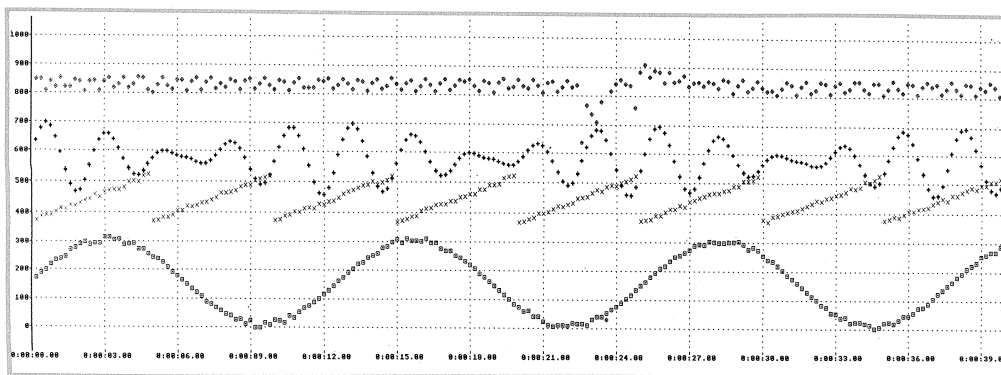
try to use them, you will get a beep and go straight back to the main menu which controls the program.

USING THE PROGRAM

When you first run the program, you will see a menu giving you the choice of reading and plotting data, or of quitting (and the options to be implemented next month).

Press "R" to read and plot data, and the program goes into a dialogue to set the session up. You have to select which of the 4 analogue inputs you are going to use, and then how often data is to be read (in seconds). The fastest the system can take data in is 5 times per second.

You must also define how long the recording is to go on for. You enter this in hours, minutes and/or seconds, and must put it in the format:

hours:mins:seconds

The colons (:) are vital. If you don't use them, the system takes the input as seconds, while a single colon just separates minutes and seconds. Thus, you would enter ten and a half minutes as "10:30" and two hours as "2:0:0".

BEWARE: the routine to read the data (FNRunTime) is NOT error trapped, and so you must get the format right. A proper parsing and checking routine would have been too long.

Next, choose either "draft" or "quality" output; draft doesn't look so good but runs faster. Here you encounter a limitation. A printer is far too slow to plot data at the full sampling speed. In fact, you can't print with draft quality at anything faster than one sample every 2.75 secs, while quality printing slows things to once every 4 seconds (about right for an Epson type printer such as the MX80 - an FX80 would probably be faster still). Thus, if you want to sample faster than that, you have to read first, and plot later, either from the data which has been saved in memory, or saved to disc as the program ran. Here's the catch - you need next month's routines to do this post-sampling plotting. If you keep the speed down, however, you can plot as you read (try a 5 second interval).

If your printer is very much faster or slower than the figures quoted, you can

change the program to suit. The line to change is 1330; the "2.75" in the listing corresponds to the "draft" speed, while the "1.25" comes from the difference between "draft" and "quality". When the program is running, it displays its progress and run time, which should let you estimate whether you need to alter anything.

To complete the set-up, Master users have the option of using the Master's real-time clock to set a future start time, rather like an alarm clock. A similar, though much less accuracte, system could probably be implemented on the model B or Compact using the TIME pseudo-variable. In practice there can more be quite large variations in time-keeping from one machine to another. Lastly you can specify a title, which will appear at the start of the chart recording to identify it, and whether the data recorded is also to be saved to disc.

Assuming that you are plotting, the program checks that all the parameters are correct, and then prints the title and the left-hand edge of the recorder output. This gives the "y-axis" graduations, with the numbers printed sideways. It then waits for you to press the space bar, when it will start to sample and, if required, save the data to disc and print it. On a Master, if a delayed start has been selected, the display counts down to the starting time.

The chart recorder output is made up of symbols printed on the chart. Four different symbols are used to identify the analogue inputs as overleaf.

Channel 1: ▣
Channel 2: ✕
Channel 3: ✦
Channel 4: ◈

The centre of each symbol shows the precise value.

The output also has vertical and horizontal graduations to help you to read data off it. The vertical graduations are drawn every 15 samples, and are marked to show the time from the start which they represent.
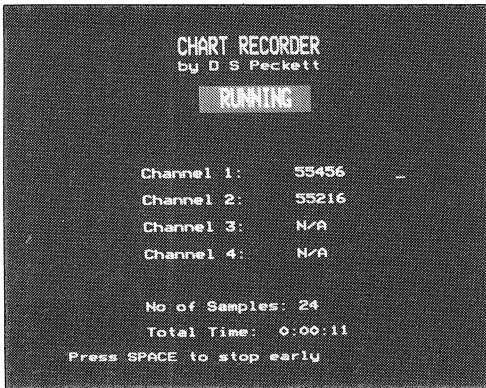
Press "space" at any time to cancel the session and return to the main menu (you may need to keep the space bar pressed down for this). From there, you can perform other manipulations on the data and re-plot it. I'll go into those other routines in more detail next month. During that re-plotting, you can set the markings on the left axis of the graph to represent the values of whatever it is you are actually measuring (degrees? pounds?). During the initial run, the program has no scaling data, and so shows the actual voltages being input to the ADC. These are in the range 0 to 1.8 Volts.

```
 10 REM Program CHART
 20 REM Version B2.6
 30 REM Author  David Peckett
 40 REM Beebug  April 1987
 50 REM Program subject to Copyright
 60 :
100 MODE 7
110 ON ERROR GOTO 240
120 PROCInit
130 REPEAT
140 PROCMenu
```

```
 150 IF Opt$="*" THEN PROCStar
 160 IF Opt$="O" THEN PROCOld
 170 IF Opt$="P" THEN PROCPlot
 180 IF Opt$="R" THEN PROCRead
 190 IF Opt$="S" THEN PROCSave
 200 UNTIL Opt$="Q"
 210 MODE 7:@%=A%
 220 END
 230 :
 240 ON ERROR OFF
 250 VDU3
 260 IF ERR=17 THEN CLS:GOTO 130
 270 @%=A%:CLOSE#0:*FX15
 280 CLS:IF ERR<128 THEN MODE 7
 290 REPORT:PRINT" at line ";ERL
 300 IF ERR>127 PRINT"Press SPACE to r
eturn to menu";:IFGET
 310 IF ERR>127 THEN 130
 320 END
 330 :
1000 DEF PROCInit
1010 mast=1 AND INKEY-256=253
1020 Free=((HIMEM-TOP-2000)DIV 2)*2
1030 DIM CBuff 9,PBuff 480,DBuff Free,o
sclibuf 30,IP(4),Tick$(4),Inval(4)
1040 FOR I%=1 TO 2
1050 PRINT CHR$129 CHR$157 CHR$131 CHR$
141 TAB(12) "CHART RECORDER"
1060 NEXT
1070 PRINT CHR$129 CHR$157 CHR$131 TAB(
12) "by D S Peckett"
1080 VDU 28,0,24,39,4
1090 VDU 23,250,&7E00;&5A42;&425A;&7E;
1100 VDU 23,251,&4200;&1824;&2418;&42;
1110 VDU 23,252,&1800;&7E18;&187E;&18;
1120 VDU 23,253,&1800;&5A24;&245A;&18;
1130 FOR I%=1 TO 4:Tick$(I%)=CHR$(249+I
%):NEXT
1140 N%=0:A%=@%:@%=&0100050A:RAM=FALSE
1150 ENDPROC
1160 :
1170 DEF PROCMenu
1180 PROCTitle("OPTIONS")
1190 X%=9:VDU23,1,255;0;0;0;
1200 FOR Y%=4 TO 20:PRINT TAB(X%-1)CHR$
131 " " CHR$135:NEXT
1210 PRINT TAB(X%,4) "R" TAB(X%,11) "O"
TAB(X%,13) "*" TAB(X%,16) "Q"
1220 IF N% AND RAM THEN PRINT TAB(X%,6)
"P" TAB(X%,9) "S"
1230 X%=X%+4
1240 PRINT TAB(X%,4) "Read/Plot new Dat
a" TAB(X%,11) "Old Data from Disc" TAB(X
%,13) "Star Command" TAB(X%,16) "Quit Pr
ogram"
1250 IF N% AND RAM THEN PRINT TAB(X%,6)
"Plot Latest Data" TAB(X%,9) "Save Late
st Data"
1260 PRINT TAB(10,20) "Which Option? ";
```

```
1270 IF N% THEN Opt$=FNGetch("*OPQRS")
ELSE Opt$=FNGetch("*OQR")
1280 ENDPROC
1290 :
1300 DEF PROCRead
1310 LOCAL I%
1320 REPEAT
1330 PROCParams
1340 IF RAM AND P1 THEN PROCRead1
1350 IF RAM AND NOT P1 THEN PROCRead2
1360 IF NOT RAM AND P1 THEN PROCRead3
1370 IF NOT RAM AND NOT P1 THEN PROCRea
d4
1380 IF ToDisk THEN PROCOpenOutFile
1390 VDU 7:PRINT TAB(0,20) "Is Everythi
ng OK? ";:OK=FNYes
1400 UNTIL OK
1410 IF NOT OK AND ToDisk THEN CLOSE#F%
1420 PRINT TAB(0,20) SPC35;
1430 T%=Tsamp*100+.5:U%=T%
1440 hival=-1:loval=99999
1450 PROCStartIt
1460 PROCRun
1470 IF ToDisk PROCCloseOutFile
1480 IF Plot THEN VDU 2,1,12,1,27,1,51,
1,32,3:REM Epson printer codes for form
feed, and 32/216 inch line spacing
1490 ENDPROC
1500 :
1510 DEF PROCRead1
1520 PRINT TAB(5,16) "Do you want to pl
ot it? ";
1530 Plot=FNYes
1540 PRINT TAB(5,17) "Save to disc whil
e reading? ";
1550 ToDisk=FNYes
1560 ENDPROC
1570 :
1580 DEF PROCRead2
1590 PRINT TAB(5,15) CHR$7 "Data rate t
oo fast to plot"
1600 PRINT TAB(5,16) "Save to disc whil
e reading? ";
1610 ToDisk=FNYes
1620 Plot=FALSE
1630 ENDPROC
1640 :
1650 DEF PROCRead3
1660 PRINT TAB(5,16) "Do you want to pl
ot it? ";
1670 Plot=FNGetch("YN")="Y"
1680 PRINT TAB(5,17) CHR$7 "No room in
RAM for Data"
1690 PRINT TAB(5,18) "Save to disc whil
e reading? ";
1700 ToDisk=FNYes
1710 ENDPROC
1720 :
1730 DEF PROCRead4
1740 PRINT TAB(5,15) CHR$7 "Too fast to
plot, no room in RAM"
1750 PRINT TAB(5,16) "You MUST save to
disc"
1760 ToDisk=TRUE:Plot=FALSE
1770 ENDPROC
1780 :
1790 DEF PROCParams
1800 PROCTitle("READ/PLOT NEW DATA")
1810 N%=0
1820 REPEAT
1830 NChan=0
1840 FOR I%=1 TO 4
1850 PRINT TAB(5,2+I%) SPC30
1860 PRINT TAB(5,2+I%) "Read data from
Channel ";I%;" ? ";
1870 IP(I%)=FNYes
1880 IF IP(I%) THEN NChan=NChan+1
1890 NEXT
1900 UNTIL NChan
1910 REPEAT
1920 PRINT TAB(5,7) SPC30
1930 INPUT TAB(5,7) "Sample rate (sec,
>0.2)? "Tsamp
1940 PRINT TAB(0,20) SPC35;
1950 IF Tsamp<0.2 THEN VDU 7:PRINT TAB(
0,20) "MINIMUM TIME IS 0.2 SECS";
1960 UNTIL Tsamp>=0.2
1970 REPEAT
1980 PRINT TAB(5,9) SPC30
1990 PRINT TAB(5,9) "Total sampling tim
e"
2000 PRINT TAB(5,10) "(Enter as hh:mm:s
s.s)"
2010 PRINT TAB(24,9);
2020 Tmon=FNRunTime
2030 PRINT TAB(0,20) SPC35;
2040 IF Tmon<Tsamp*2 THEN VDU 7:PRINT T
AB(0,20) "MUST TAKE AT LEAST 2 SAMPLES!"
;
2050 UNTIL Tmon>=Tsamp*2
2060 NSamp=NChan*Tmon/Tsamp
2070 RAM=(NSamp*2)<Free
2080 PRINT TAB(5,11) SPC20
2090 PRINT TAB(5,11) "Draft or Quality
Print (D/Q)? ";
2100 QPlot=(FNGetch("DQ")="Q")
2110 P1=Tsamp>(2.75-1.25*QPlot)
2120 IF mast THEN PRINT'TAB(5,12)"Set f
uture start time? ";:MC=FNYes
2130 IF mast:IF MC THEN INPUT'TAB(5,13)
"Enter time (hh:mm:ss)? "TM$
2140 INPUT'TAB(5,14)"Title? "Title$:Tit
le$=STRING$((80-LENTitle$)/2,CHR$32)+Tit
le$
2150 ENDPROC
2160 :
2170 DEF PROCStartIt
2180 IF Plot THEN PROCtitle(Title$)
2190 PROCTitle("RUNNING")
2200 VDU23,1,0;0;0;0;:FORI%=1 TO 4
```

```
2210 PRINT TAB(8,I%*2+4) "Channel ";I%;
":"
2220 IF NOT IP(I%) THEN PRINT TAB(23,I%
*2+4) "N/A"
2230 NEXT
2240 PRINT TAB(8,16) "No of Samples:"
2250 PRINT TAB(8,18) "Total Time:"
2260 IF Plot THEN PROCHeader(0,65536,0,
1.8,0)
2270 IF mast:IF MC THEN REPEAT PRINTTAB
(5,20);"Waiting: ";RIGHT$(TIME$,8);:UNTI
L FNtimecomp:GOTO2300
2280 VDU 7:PRINT TAB(0,20)"Ready -"CHR$
136"SPACE"CHR$137"to start ";
2290 REPEAT UNTIL INKEY-99
2300 ENDPROC
2310 :
2320 DEF PROCRun
2330 PRINT TAB(0,20) "Press"CHR$136"SPA
CE"CHR$137"to stop early";
2340 P%=0:st%=1
2350 TIME=0
2360 REPEAT
2370 N%=N%+1
2380 FOR I%=1 TO 4
2390 IF IP(I%) THEN PROCHandleData
2400 NEXT
2410 PRINT TAB(23,16);N%
2420 PRINT TAB(20,18);LEFT$(FNTime(TIME
),8)
2430 IF Plot THEN PROCGraphIt(0)
2440 REPEAT UNTIL TIME>=U%
2450 U%=U%+T%
2460 UNTIL TIME>=Tmon*100 OR INKEY-99
2470 ENDPROC
2480 :
2490 DEF PROCOpenOutFile
2500 PRINT TAB(5,18);:INPUT "Name of di
sc File? "FilName$
2510 PRINT TAB(5,19) "Insert disc,"CHR$
136"SPACE"CHR$137"when ready ";
2520 REPEAT UNTIL INKEY-99
2530 F%=OPENOUT(FilName$)
2540 PTR#F%=60
2550 ENDPROC
2560 :
2570 DEF PROCCloseOutFile
2580 PTR#F%=0
2590 PRINT#F%,N%,Tsamp,Tmon,hival,loval
,IP(1),IP(2),IP(3),IP(4)
2600 CLOSE#F%
2610 ENDPROC
2620 :
2630 DEF PROCHandleData
2640 LOCAL V%
2650 V%=ADVAL(I%)
2660 hival=FNMax(hival,V%)
2670 loval=FNMin(loval,V%)
2680 IF RAM THEN DBuff!P%=V%:P%=P%+2
2690 IF ToDisk THEN BPUT#F%,V% MOD 256:
BPUT#F%,V% DIV 256
2700 PRINT TAB(23,I%*2+4);V%;SPC5
2710 Inval(I%)=V%
2720 ENDPROC
2730 :
2740 DEF PROCHeader(ADlo,ADhi,anlo,anhi
,tim)
2750 LOCAL I%,J%,step
2760 NxtTim$=FNtime(tim*Tsamp*100)
2770 step=(anhi-anlo)/10
2780 VDU 2,1,27,1,51,1,24:REM Epson pri
nter codes for 24/216 inch line spacing
2790 FOR I%=1 TO 10
2800 PROCClrBuffa
2810 FOR J%=0 TO 10
2820 PROCPosChar(MID$(RIGHT$(STRING$(10
," ")+STR$(anlo+J%*step),10),I%,1),50+J%
*40)
2830 NEXT J%
2840 IF I%>5 THEN PROCPosChar(MID$(NxtT
im$,I%-5,1),10)
2850 PROCPrtLine
2860 NEXT I%
2870 PROCLeftAxis
2880 PROCPrtLine
2890 VDU 3
2900 Yscale=400/(ADhi-ADlo)
2910 L%=1
2920 ENDPROC
2930 :
2940 DEF PROCGraphIt(lo)
2950 LOCAL I%,L1%
2960 VDU 2
2970 PROCClrBuff
2980 FOR I%=1 TO 4
2990 IF IP(I%) THEN PROCPosChar(Tick$(I
%),(Inval(I%)-lo)*Yscale+50)
3000 NEXT
3010 L1%=L% MOD 15
3020 IF L1%=8 THEN NxtTim$=FNtime((L%+s
t%+6)*Tsamp*100)
3030 IF L1%>9 OR L1%<6 THEN PROCPosChar
(MID$(NxtTim$,(L1%+6)MOD 15,1),10)
3040 IF L1%=0 THEN PROCVertGrad
3050 PROCPrtLine
3060 L%=L%+1
3070 VDU 3
3080 ENDPROC
3090 :
3100 DEF PROCGetFont(ch$)
3110 LOCAL A%,X%,Y%
3120 ?CBuff=ASC(ch$)
3130 A%=&A
3140 X%=CBuff MOD 256
3150 Y%=CBuff DIV 256
3160 CALL &FFF1
3170 ENDPROC
3180 :
3190 DEF PROCClrBuff
```

→38

Computer Concepts casts a Spell

Computer Concepts has at long last released Spell Master, its ROM/RAM based spelling checker for the Beeb. Mike Williams, dictionary in hand, spells the beans.

> Product : Spell Master
> Supplier : Computer Concepts,
> Gaddesden Place,
> Hemel Hempstead,
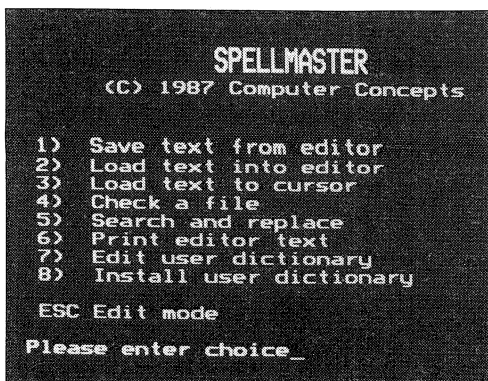> Herts HP2 6EX.
> Tel. (0442) 63933
> Price : £59.00 inc VAT.

Special Offer-£49.15 to members!

Word processing must be one of the most popular applications on any micro. And anyone who uses a word processor soon learns to value the help of a computerised spelling aid, particularly if the resulting document is to have more than purely personal circulation.

Beeb users are well blessed with word processors (View, Wordwise, Edword and Scribe are a few that spring to mind). There is less choice when it comes to spelling checkers, but BEEBUG's own Spell-check, and Acorn's more recent ViewSpell have both provided sterling service. Into this established market comes Spell Master, and with a hefty £59 price tag. What does this newcomer have to offer, and can this price be justified?

First of all, let's be clear what Computer Concepts provides. Spell Master contains a whopping 128K of ROM in the form of two 64K chips mounted on a small circuit board. This fits into any normal ROM socket, and overhangs the ROM to its right. The 128K contains not only the software, but a 57,000 word dictionary as well. Compared to the likes of ViewSpell and Spellcheck, both with disc based dictionaries, that gives Spell Master an immediate and significant speed advantage.

Spell Master offers two further, major advances over any other spelling checker for the Beeb. Most usefully, it can be

SPELLMASTER
(C) 1987 Computer Concepts

1) Save text from editor
2) Load text into editor
3) Load text to cursor
4) Check a file
5) Search and replace
6) Print editor text
7) Edit user dictionary
8) Install user dictionary

ESC Edit mode

Please enter choice_

called from within most word processors (including View, Wordwise and Interword). No longer is it necessary to save the text from your word processor, transfer to the spelling checker, reload the text into memory, then check the spelling, typically saving the corrected text before returning to the word processor and reloading the text. With Spell Master, you can immediately check the spelling of your current text.

IMMEDIATE MODE CHECKING

Furthermore, Spell Master allows words to be checked automatically as they are entered or edited with your word processor. Once immediate checking is switched on (Ctrl-J), every word entered or edited (except one and two letter words which Spell Master ignores) are instantly checked against the dictionary. There really is no discernible time delay.

Incorrect spellings are indicated by a 'bleep'. Moving the cursor back onto the word and pressing Ctrl-C forces a re-check. At this point, a three-line option window appears. This allows you to continue word processing, to add the word to a user dictionary (more on this later), or to correct the spelling. If the latter option is chosen, Spell Master displays a short segment of its dictionary, high-ighting the word nearest in spelling to the incorrect one. As you begin to amend the spelling, Spell Master homes in unerringly on the correct version. As soon as Spell Master has reached the correct word, pressing Return will result in that word being substituted for the original one without further ado.

If you do intend to rely on immediate mode checking, and there would appear to be very strong reasons for so doing, then you must ensure that it is always active. Any subsequent editing session that does not use this facility opens up the possibility of typing mistakes (or plain mis-spellings) which go unnoticed, and the entire text will then need to be checked from beginning to end.

Still within your word processor, you can check all the text, either from the start of the file or just from the current cursor position, while Ctrl-B will allow you to browse through Spell Master's dictionary. The up and down cursor keys, by themselves and in conjunction with Shift and Control, provide fast scanning through the dictionary, allowing any word, or part of the dictionary, to be located very quickly indeed.
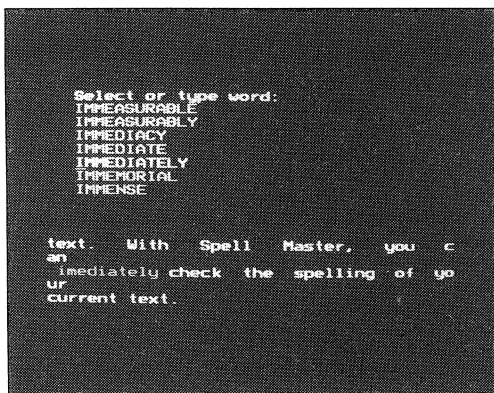
USER DICTIONARIES

I have already mentioned, though briefly, user dictionaries. The approach adopted by Computer Concepts will certainly not endear them to a good many model B users. Spell Master dictionaries can be held ONLY in sideways RAM (or if you are sufficiently confident in your own dictionary, up to 3000 words or so may be blown into an EPROM). If you have a standard model B, with no sideways RAM, then user dictionaries are definitely out.

With sideways RAM fitted, or on a Master or Compact with built in sideways RAM, user dictionaries are checked quite automatically once loaded, if a word cannot be found in the main dictionary.

This action is entirely transparent to the user. When in immediate mode checking, new words, as mentioned before, can be added directly to a user dictionary.

There are also facilities for creating and editing a user dictionary, and for adding groups of words (as opposed to single words). These options are selected from Spell Master's main menu. Spell Master even provides star commands (*DSAVE and *DLOAD) for transferring user dictionaries between disc and sideways RAM.

There is a further practical point here, again of particular importance to prospective purchasers with a model B. I have always found Spellcheck's 'temporary save' facility very useful, but no such option is explicitly provided by Spell Master. The only alternative would seem to be to have a 'garbage' user dictionary to which all odd or special words may temporarily be added as they are first found, but again you MUST have sideways RAM to do this.

SPELL MASTER EDITOR

There is still more to Spell Master. *SPELL calls up a mode 7 menu (like Wordwise and Interword). This includes an editor with many similarities to Wordwise, including the same block delete, move and copy, though no formatting commands. One option in this menu allows text files on disc, too long for memory, to be checked. Other options provide for the creation and editing of user dictionaries.

Lastly, and perhaps mainly for crossword addicts, there are a number of star commands, including the facilities to locate anagrams, and to match partly completed words, though I must tell you that using Spell Master takes all the fun and challenge out of solving crosswords. In future, I think I shall just place my Grauniad (try *ANAGRAM on that) in front of the micro and let it get on with it.

It is impossible in the space available to describe every feature of Spell Master. Suffice it to say that this just has to be the fastest and most convenient spelling checker for the Beeb today, but only if you have some form of sideways RAM. The manual is clear enough (I did find one spelling mistake!) and the price – well I leave you to decide that for yourself. 🅱

Sciways for Scientists

Scientific and mathematical notation can be next to impossible with most word processors. Now, Sciways can work with View or Wordwise (and many others) to produce a wealth of new symbols. And if you don't like the ones supplied, you can always design your own.

> Product : Sciways ROM
> Supplier : Mayhew Telonics,
> 376A Ringwood Road,
> Poole, Dorset BH12 3LT.
> Tel. (0202) 747695
> Price : £38.52 inc VAT, p & p.
> 16K ROM, 40/80 track disc
> and manual.

In the midst of ROMs for alternative character sets, ROMs for fancy graphics and ROMs for desktop publishing, Mayhew Telonics has spared a thought for the scientific user of the BBC micro and Master, and has provided a ROM to service him (or her) with just about any conceivable special character. Here are integrals, sets, derivatives, super-scripts, subscripts and two complete Greek alphabets. Not only this, but each character can be displayed on screen as well as being sent to the printer.

Unlike the majority of NLQ character sets, each Sciways character is built from an 8 x 8 cell. Printed copy is therefore of lower resolution, but you can see what you're going to get before you start to print. This is just as well, as some tall mathematical symbols, such as integrals and matrix brackets, have to be printed in sections over more than one line. Lining up the various segments of such characters would be very haphazard if you had to take test prints to check the positioning each time. Regretably, you cannot fully edit any of the special characters directly, but only in the form of trigger characters and associated keyboard symbols. The full character display is more of a preview display, but none-the-less very useful.

You can incorporate Sciways characters into any word-processor, or into Basic

listings. The ROM works by looking for the 'trigger' symbols, which are put into your text in pairs with ordinary keyboard characters. The character pair is compared with a look-up table in the Sciways ROM, and the appropriate character is substituted by the ROM when displaying or printing the document. The trigger characters have been selected to occur fairly rarely in scientific documents, but if you need to use them, you can turn the Sciways characters on and off with separate star commands.
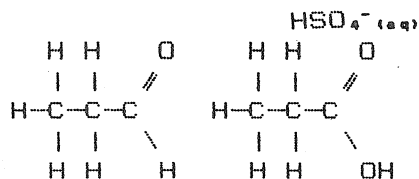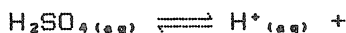
PLINGS AND HASHES

Sciways is extremely easy to use. You install the ROM in any vacant socket (it can be used from an expansion board) and type *CHON to activate the extra characters. From then on over 350 extra characters are available. They are accessed by preceding characters on the keyboard with the trigger characters !(pling), #(hash), % and &. Pling is used for the Greek character set, and *GREEKI changes all the characters from upright to italic. Hash is reserved for symbols, and there is a complete table of these in the excellent 40 page manual provided.

% and & are used for superscripts and subscripts, and it is possible to produce superscripted subscripts and subscripted superscripts. If you don't understand these terms, you probably won't have occasion to use them! Superscripts and subscripts are produced using the Epson's own fonts, and look very professional. They don't work with NLQ print selected, though.

As well as the characters provided within the ROM, you can define up to 26 of your own, using the character definer provided on the accompanying disc. You can load files of your own characters at any time and incorporate them into your work by using the $ sign as a trigger symbol. You could use the facility, for instance, to add foreign language accents to letters.

In addition to the facilities already described, Sciways provides a neat alternative to sending control characters to your printer. Using the trigger character @, you can set any of the standard Epson printer functions (excluding NLQ) with just two characters – very handy.

$$H_{2(g)} + \tfrac{1}{2}O_{2(g)} \longrightarrow H_2O_{(g)}$$

$$H_2SO_{4(aq)} \rightleftharpoons H^+_{(aq)} + HSO_4^-{}_{(aq)}$$

$$\begin{array}{ccc}
H & H & O \\
| & | & \| \\
H\text{--}C\text{--}C\text{--}C & & \\
| & | & \backslash \\
H & H & H
\end{array}
\qquad
\begin{array}{ccc}
H & H & O \\
| & | & \| \\
H\text{--}C\text{--}C\text{--}C & & \\
| & | & \backslash \\
H & H & OH
\end{array}$$

$$\text{As } a \to \infty \quad 1 \div a \to 0$$

$$\sim(A \vee B) \equiv \sim A \wedge \sim B$$

$$a^2 = b \iff a = \pm\sqrt{b}$$

$$|\alpha| \geqslant 1 \implies \alpha \geqslant 1 \text{ or } \alpha \leqslant -1$$

$$1 \div x \leqslant 1 \impliedby x \geqslant 1$$

$$\int_0^{2\pi} \theta^2 \, d\theta = \left[\tfrac{1}{3}\theta^3 \right]_0^{2\pi}$$

ROM MANAGER

Sciways has to intercept the WRCHV vector to check for the trigger character, and claims workspace in the cassette buffer and in envelopes 3 and 4. In most cases this won't cause any problems, but to be on the safe side Sciways includes a simple ROM manager which allows you to 'unplug' any ROMs which cause a conflict. There was a conflict on my BBC B, but ironically this only showed up when trying to use Sciways' ROM manager! Mayhew Telonics specifically mention graphics ROMs as possible 'baddies'.

TAXED TAXAN

Sciways is designed to work with an Epson FX or MX printer, and with these it may well do the job. It is also intended to work with Epson compatible printers, including the Kaga Taxan, which is the work horse I press into service. Although the output is quite adequate (witness the sample printouts accompanying this review), some kind of contretemps between the Sciways ROM and the Kaga printer forced its head to jigger quite alarmingly while printing. It seems to back up a character after each Sciways character is printed, as if it was going to embolden it. The end result is that I wouldn't give much for the life of the head mechanism if you were using a Sciways/Kaga combination for any long print jobs (a Ph.D. thesis, for instance).

VERDICT

Sciways is a very well thought out product. It gives scientific users the opportunity to construct documents which contain all kinds of mathematical and chemical symbols. Up until now, such documents had to be constructed painstakingly with user-defined characters, or added by hand after printing the document. The Greek character set is very useful in its own right (mathematicians and scientists use many Greek symbols as standard notation), and the abbreviated control sequences for Epson printers are a worthwhile bonus. The advantage of being able to see the characters in preview mode on Wordwise and View helps considerably in laying out complex formulae.

Weighed against the advantages of the ROM is the fact that none of the characters are NLQ. Universities have varying rules, but I know of several that won't allow theses and final year projects to be submitted in 'ordinary' dot matrix print, insisting instead on NLQ or daisywheel copy. I suspect there are many other institutions with similar restrictions.

There's no doubt that Sciways is a valuable product for anyone engaged in scientific writing, and must be about the cheapest solution to creating a 'scientific' word processor. The same effects on an IBM PC would cost you four or five times as much.

THE COMMS SPOT
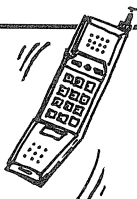
Peter Rochford has been running up a large phone bill in his attempts to bring you the latest on Epnitex, first revealed in the Jan/Feb issue. BEEBUG's own database on Micronet, plus information on the BEEBUG Command ROM.

Epnitex

In the last Comms Spot, we announced preliminary details of a new dial-up viewdata service to be launched by Timefame International, called Epnitex. This service, according to Timefame, was to be made public in January 1987. At the time of writing (early February), the service was still to be launched, though this did appear imminent. This slow progress I find disappointing.

I have, however, now been allowed access to the trial service on several occasions. Although I promised last time to give you full details of what Epnitex offers, until Timefame makes a firm announcement about the future of Epnitex, I will not use up too much valuable space here in describing the system in fine detail.

At present, the service operates via a phone number in Lincoln. So for me, living in London, even a short spell wandering around the pages of Epnitex on off-peak time damaged my phone bill quite heavily.

However, the expense was well worth it and my experience of using Epnitex was a real pleasure. There is not a lot to see and read yet on the system, but just enough to enable you to get a feel for using it, and an appreciation of the facilities offered.

In my last Comms Spot, I said that Timefame boasted that the software to drive Epnitex was sophisticated, and had facilities far in excess of Prestel. Their claims are well founded and the whole system is a dream to use. After using Epnitex, Prestel looks and feels archaic.

There is an enormous amount of potential for both IPs and subscribers to Epnitex. The system is remarkable and offers some wonderful facilities, many of which I described in the last feature. However, as I said earlier, Timefame has been dragging its heels over the launch, and the all important details of costs (thought to consist of an annual subscription in excess of £100). Many other questions also remain unanswered.

I hope the system does see the light of day, and look forward to more concrete answers from Timefame in the near future. When that happens, I·shall look at Epnitex again in more detail in these pages.

Beebug on Micronet

Meanwhile, I want to announce some really exciting news of another new database. I am referring to BEEBUG's own
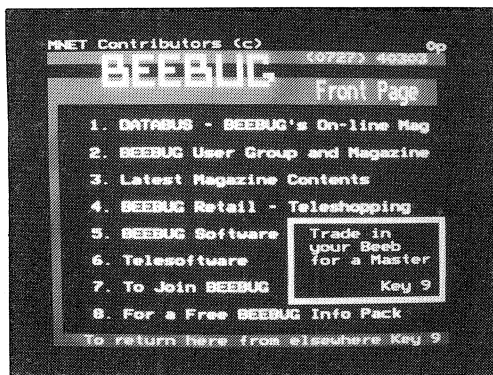
database that is now available to those of you who subscribe to Micronet on Prestel.

Just key *800909# once logged-on to Prestel to get to the BEEBUG main index. Here you will find details of all the services and features available. As you can see from the screen pictures these are already quite comprehensive, but in these early days you may find that some of the routes do not lead as far as they might. The database is still being worked on, so please be patient.

We hope that the BEEBUG database will complement the magazine and make membership of BEEBUG even more worthwhile. As you will see from the main menu, we are running an on-line magazine called Data-Bus. This will feature articles, programs, hints and so on. Many will derive, in the first instance at least, from BEEBUG magazine. And it is hoped to put programs into the telesoftware area on a regular basis. It may also be that as time goes on, contributors will be able to submit work specifically for Data-Bus.

The BEEBUG Retail section allows you to do a spot of tele-shopping. A large selection of the goods stocked by the BEEBUG shop are listed here along with pricing. Once you have decided what you want, you can make your way to the response frame and order direct using your credit card. This will certainly help speed up your order. In the future it is hoped to give much fuller descriptions of some of the products offered, so as to provide much more than just a catalogue of items and codes.

All-in-all, the launch of BEEBUG's own database reflects the steadily growing interest in communications. We hope that you will both read and contribute to it.

Beebug Command ROM

Finally this month, I am going to plug one of BEEBUG's own products. We naturally don't review our own commercial software in the BEEBUG magazine. However, I feel that this is a very useful new product that all comms users should be aware of. For a full product description plus sample screen displays, users of Micronet can key *800909201#. The Command ROM has also received favourable reviews in other publications. If you want to get started in comms then this is clearly a ROM worth considering.
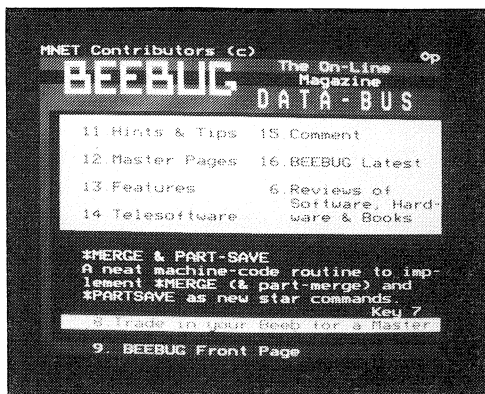
In recent months I have always used Soft Machinery's excellent Commsoft ROM for most of my comms needs. I am now finding myself using Command more and more instead.

At the moment, Command in its standard form will only auto-dial BEEBUG's own Magic Modem, and other similar modems (e.g. Kirk Enterprise, Watford Apollo, and Voyager 7). It will also auto-dial the Demon modem if the modem is first switched to auto baud selection. This doesn't, however, prevent you from using the ROM with other modems using manual dialling. A disc of Command applications programs is also expected soon.

Streamlining Data Entry on the BEEB

For many programs that process data in some way or other, the chore of entering and editing your data can be the biggest problem of all. C.R. Woodings provides a utility that allows ViewSheet to be used for many of your data entry requirements.

If you have ever tried to write a program to draw graphs or to analyse a lot of data you will have discovered, perhaps to your surprise, that the data entry, storage, and editing routines need to be quite long and complex. Putting the results in data statements, or having a direct INPUT into an array simply won't do when you have more than a few data points, or when you want someone else to be able to run the program.

The availability of ViewSheet, and particularly the arrival of the Master 128 with its built in ViewSheet, ought to have changed all this, by providing a ready made vehicle for all manner of data entry requirements. In reality, there appears to be no really convenient way to interface ViewSheet data files with your own data analysis programs.

ViewSheet does have its WRITE and READ statements which help you to transfer data between spreadsheets, and in the process create an array on disc of whatever blocks of data you care to choose. However, if you've tried these with more than a tiny spreadsheet, you will have noticed that you need in effect two complete sheets in memory at once. You need the original data spreadsheet and a complete image of it in WRITE statements alongside. This is not only very wasteful of memory, cutting the maximum size of sheet by more than half (a very serious problem if you are using ViewSheet on the Model B), but it is also tiresome to create, and very slow to use. Every time you re-calculate, ViewSheet writes each cell to disc, and this can take several minutes. You then need a special decoder routine in your own program to read this data from file into your own routines.

There is, however, a much more flexible, space saving, and faster route available. It makes use of printer windows, the SCREEN command, and SPOOLING, as described in the manual for transferring ViewSheet tables to View. This will put any selected part of a spreadsheet onto disc as an ASCII file. The listing provided with this article (the program DECODER) can then be used to decode this file, and create separate data files from each of the columns of data in the spreadsheet. These data files may then be accessed by your own programs.

ViewSheet is quite easy to use, providing excellent facilities for entering and editing data. It also allows easy printout of data for subsequent checking. Thus ViewSheet provides an ideal data entry environment for many users, which can now be interfaced to your own Basic programs.

USING DECODER

Once you have typed in the program, and saved a copy, proceed as follows:

1. Select ViewSheet and enter your data in the usual way. Bear in mind that this version of DECODER will give you access to data in column order. You can use headings and labels with the limitation that column headings that have been created with Shift-f3 become disc filenames, and must be legal for your DFS.

2. Decide which area of the sheet you want to use, and avoid including too many labels. DECODER will not actually stall on labels, but will just record them as zeros, and this could disrupt the appearance of subsequent graphs.

3. Set up a PRINTER WINDOW as described in the ViewSheet manual, covering the area of interest. Make a note of the number of columns and rows in the window, and the column width you have used (DECODER will need this information later). The disc system in use sets limitations on the size of the window you can use. You will use one file for the window and one for each of the columns in the window. So on a clean disc, the maximum number of columns is 30 for the DFS and 46 for the ADFS (only one directory is used). These numbers are diminished by the number of files already on the disc (or in the ADFS directory.)

4. Press Escape to get to the command screen and put into your drive the appropriate disc. Type:
         SAVE <filename1>
to save the whole sheet and window definition, followed by:
         *SPOOL <filename2>
          SCREEN
         *SPOOL
to save the data in your chosen window.

You now have a disc file which contains the data you need for your own programs, and which can be interpreted by DECODER.

5. Return to Basic and CHAIN "DECODER" (if you are using the ADFS, create a directory with the name D first). Respond as appropriate to the prompts for filename (i.e. use <filename2>), rows and columns. These latter inputs are used to set up a temporary array, and any numbers larger than those actually used, and less than the 'Bad DIM' error point (about 1200 cells on the Master) will do. You need to enter the column width precisely; getting it wrong will give you unexpected results!

6. The decoded file will be displayed in mode 3 (alter line 670 to change this). Inspect it carefully to ensure you have the data you wanted, and if all is well press any key to save the columns on disc. If you don't like what you see, re-enter ViewSheet, reload <filename1> and re-edit, and/or change your printer window definition. If it looks nothing like what you put in, you probably entered the wrong column width, so just re-run DECODER and try again (7 is the default width in ViewSheet).

If you catalogue the disc you will now see one or more files in the D directory. (their names will be A, B, C etc. if you used the default column headings in ViewSheet). The first item in each file is the number of rows of data held in the file. This is followed by the data itself in down-the-column order.

The second listing is a procedure which you can add to your own programs to load a data file thus created into a previously defined array. It first lists the data files available on disc and prompts you for a filename. The data from the specified file is loaded into the array

data(), which must have been dimensioned previously. It also loads the number of data points (the first data item) into the variable 'rows'.

PROGRAM NOTES
    It is important that you match up exactly the numbers of rows and columns in any spreadsheet that you create with the values that you enter when running the DECODER program. At worst, any discrepancy will give rise to a 'Subscript' error message indicating that the arrays created by the DECODER program are too small for the spreadsheet specified.

    If the use of the directory 'D' is inconvenient, then amend line 830 of DECODER (either change 'D.' to your preferred alternative, or omit all together. Likewise, amend line 10020 in the loader procedure.

    Lastly, PROCoscli called at line 150 satisfies the needs of Basic I users. Those with later versions of Basic could replace this with a call to the OSCLI function, though this is by no means essential.

CONCLUSION
    This approach to creating and editing data files using ViewSheet works well, and allows you to spend more time on programming the application itself rather than on data entry. Using the DECODER program, ViewSheet becomes a general purpose data entry system that can look after the data needs of many application programs.

DECODER Program

```
 10 REM Program Viewsheet Decoder
 20 REM Version B0.1
 30 REM Author  C.R.Woodings
 40 REM BEEBUG  April 1987
 50 REM Program subject to copyright
 60 :
100 DIM O$ 40:ON ERROR GOTO 920
110 HIMEM=&4000:REM start of buffer fo
r Viewsheet screen file.
120 MODE7:INPUT"Enter SCREEN filename
",file$
130 start=HIMEM
140 pointer=start
150 PROCoscli("LOAD "+file$+" "+STR$~s
tart)
160 IF LEFT$($start,2)<>"=>" PRINT"Thi
s is not a Viewsheet Screen File":a=INKE
Y(200):GOTO 120
```

```
 170 INPUT"Enter Max. Columns Needed ",
cols
 180 INPUT"Enter Max. Rows Needed (99=m
ax) ",rows
 190 DIM datarray(cols,rows),filename$(
cols)
 200 INPUT"Enter Column Width (Normally
 7) ",width
 210 PRINT'"Decoding";
 220 :
 230 REM pointer moved past file header
 240 pointer=pointer+LEN($start)
 250 :
 260 REM get column headings if used
 270 colcount=0
 280 REPEAT
 290 a$=""
 300 colcount=colcount+1
 310 REPEAT
 320 pointer=pointer+1
 330 UNTIL?pointer>&40
 340 REPEAT
 350 a$=a$+CHR$(?pointer)
 360 pointer=pointer+1
 370 UNTIL?pointer=&2E OR ?pointer=&0A
 380 filename$(colcount)=a$
 390 UNTIL?pointer=&0A
 400 :
 410 REM get data from table
 420 rowcount=0
 430 REPEAT
 440 rowcount=rowcount+1
 450 :
 460 REM discards row title
 470 REPEAT
 480 pointer=pointer+1
 490 UNTIL?pointer=&20
 500 :
 510 REM gets data in columns
 520 FOR col%=1 TO colcount
 530 data$=""
 540 FOR char%=0 TO width
 550 data$=data$+CHR$(?pointer)
 560 pointer=pointer+1
 570 NEXT
 580 IF VALdata$=0 data$="0":REM makes
text or blanks zero for EVAL
 590 datarray(col%,rowcount)=EVAL(data$
)
 600 PRINT".";
 610 NEXT
 620 :
 630 REM check for end of SCREEN file
 640 UNTIL LEFT$($(pointer+2),2)="=>" O
R ?pointer=0
 650 :
```

```
 660 REM display info. in datarray
 670 MODE3
 680 FOR N=1 TO colcount
 690 PRINTTAB(POS+9)filename$(N);
 700 NEXT
 710 PRINT
 720 FOR Y=1 TO rowcount
 730 FOR X=1 TO colcount
 740 PRINTdatarray(X,Y);
 750 NEXTX
 760 PRINT
 770 NEXTY
 780 :
 790 REM Save columns on disc for later
 plotting and processing
 800 PRINT"Any Key to Save Columns: ESC
to Quit "
 810 A=GET
 820 FOR N=1 TO colcount
 830 D=OPENOUT("D."+filename$(N))
 840 PRINT#D,rowcount
 850 FOR M=1 TO rowcount
 860 PRINT#D,datarray(N,M)
 870 NEXT M
 880 CLOSE#D
 890 NEXT N
 900 END
 910 :
 920 ON ERROR OFF:CLOSE#0:MODE7:IF ERR<
>17 REPORT:PRINT" at line ";ERL
 930 :
1000 DEF PROCoscli($os)
1010 LOCAL X%,Y%
1020 X%=os:Y%=os DIV 256:CALL &FFF7
1030 ENDPROC
```

LOADER Program

```
   1 REM Procedure to load the Viewshee
t column files created with DECODER
   2 REM BEEBUG April 1987
   3 :
10000 DEF PROCdataload
10010 PRINT'"Data files on this disc:"
10020 *INFO D.*
10030 INPUT"Enter a filename from list:
"filename$
10040 channel=OPENUP(filename$)
10050 INPUT#channel,rows:REM 'rows' is l
oaded with the number of data points on
file.
10060 FOR point=1 TO rows
10070 INPUT#channel,data(point):REM arra
y must have DIM in main program
10080 NEXT
10090 CLOSE#channel
10100 ENDPROC
```
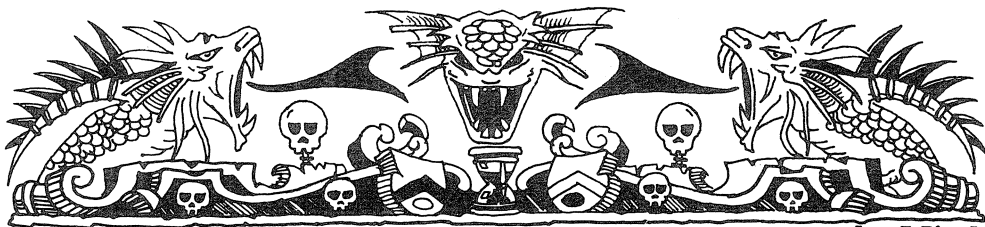
TURE GAMES ADVENTURE GAMES ADVEN

Title : **Myorem**
Supplier : **Robico Software,**
3 Fairland Close,
Llantrisant,
Mid Glamorgan CF7 8QH.
Tel. (0443) 227354
Price : **£9.95 (cassette), £12.95 (disc)**

Like the unstoppable hero of their own adventures, Robico have produced the third part of their 'Rick Hanson' adventure trilogy. This game takes place deep in a South American jungle which is crawling with spiders, alligators and trigger-happy cocaine smugglers. As with Robico's former games, there are innumerable locations and objects, plus acres of wittily written text to amuse and confuse. As a mark of its sophistication, the game correctly anticipates many of your wrong solutions and leads you a merry dance before breaking the bad news that you are once again up that creek! In Myorem, the problems, if not the solutions, are usually clear, allowing you to see the wood amongst the jungle palms. Wisely, however, Robico does not hide its telephone number, as they expect and welcome your call for help.

I confess that I did require some assistance in the early stages of the game, but once off and running, I managed finally to crawl victorious to bed in the early hours - and I have the hard-boiled eyes to prove it! This in itself is proof of the game's appeal, as I confess that second rate adventures no longer last long around the Dungeon. After staring at "THE PRICE OF MAGIC" from Level 9 for three brain numbing evenings I realised that I'd spent more enjoyable nights chatting to the Troll! Tossing that game into the far corner of the cave, I put the dragon out and went thankfully to bed.

Myorem is built around the novel theme

that all the action is taking place in flashback, and should your concentration waver for a moment then you will return to a very unhappy present situation! As Dr. Johnson once remarked "Nothing concentrates the mind of a man more than the knowledge that he is to be hanged in the morning!".

The game has what appears to be two mazes, but thankfully they are solved by methods other than mapping. One clue I will let slip - SEARCH is a word Robico loves.

Myorem is certainly back up to the standard we have come to expect from Robico and it certainly deserves a place in your library. It may confuse but it won't bore.

FANZINES

So what do you do when you've read all the Adventure columns in the glossy mags and you still want more? What do you do with that adventure game you finally wrote, and where do you turn for advice while wrestling with the Quill? Perhaps an Adventure Club is the answer.

Specialist adventure clubs with their own magazines (fanzines to you), written and controlled by dedicated fans, can be either miserable rip-offs or truly professional efforts. 'Adventure Contact' run by Pat Winstanley, 13 Hollington Way, Wigan WN3 6LS appears to be one of the better efforts. There is a bulky, monthly newsletter which is full of letters and articles of interest to adventure fans and writers - cost: £1 for a sample copy. The membership is mostly Amstrad and Commodore users at present, but BBC members will be welcomed with open arms.

In addition there is 'Adventure

Probe', which is also run by a member of the fair sex – Sandra Sharkey, 78 Merton Road, Highfield, Wigan WN3 6AT. This is an associated club with a similar setup which can provide help and advice on all aspects of adventuring. Contact either ladies enclosing a S.A.E., and preferably money (£9 for a year, or £1 for a sample copy).

It is heartening to see all this expertise alive and kicking. With luck, more BBC/Electron users will be inspired to produce epics which will keep us glued to the keyboard. I was amazed to read all the letters concerning clever tricks with the Quill and Graphic Adventure Creator for would-be authors, plus the lively discussion on how to market the finished games. Of course, the Quill and G.A.C have been available for other micros for some time now, and BBC owners could benefit from the experience of others. Various add-ons, professional and amateur, are available for these adventure generators and you can pick up much useful advice on how to use them to individualise your own creations.

Title   : **Winter Wonderland**
Supplier : **Incentive Software Ltd.**
            2 Minerva House, Calleva Park,
            Aldermaston, Berks RG7 4QW.
            Tel. (07356) 77288
Price   : **£7.95 (cassette or disc)**

When Incentive released their Graphic Adventure Creator software they mentioned that they would be interested in seeing any quality game produced using the package. Home grown games are marketed as the 'Medallion' range, and two have now been released of which Winter Wonderland is the first. As usual, the BBC game appears to be a cut-down of other versions involving quite a few locations with no graphics, and those which are illustrated are almost identical to each other.

The game is extremely simple and any grizzled old adventurer will hack his/her way through the puzzles without any difficulty, in two or three sessions. The library of commands understood by the game is very limited and the puzzles trivial.

The game's story line concerns your crash landing and subsequent wanderings around the icy wastes of old Tibet. You quickly stumble into Shangri-La which is complete with Shopping Centre, Skate Hire

Shops and Hotels, and a Cash Card with enough of the built-in readies to pay off Tibet's national debt!

The low chuckling noise which you will hear throughout the game is probably caused by the staff of Level 9 and Robico Software as they assess the damage that future games of this range will have on their sales!

This game appears to be just what it is – a short, unsophisticated game written by an amateur. If this is the standard of future games in the series then to be successful they must either be advertised as suitable for beginners, or sold as budget software with a corresponding drop in price.

Title   : **The BBC Handbook**
Supplier : **H & D Services,**
            1338 Ashton Old Road,
            Higher Openshaw,
            Manchester M11 1JG.
Price   : **£1 monthly**

Whilst billed as an Adventurer's Handbook, this monthly magazine is in fact concerned with adventure and other games, providing hints, solutions and reviews. Being the same size and appearance of an early Beebug, the mag has approximately 40 pages of maps and solutions to BBC adventures. Issue 14, for example, contained solutions to 'The Hobbit', 'Dennis through the Drinking Glass', 'Sorceror of Claymorgue Castle' and 'Gisburnes Castle'.

The style and standard of writing is extremely high compared with many 'fanzines', and I found the articles witty and interesting. Issue 14 also contained a review of 'Winter Wonderland' and I was pleased to see that the reviewer agreed with my own assessment of the game, (obviously a gentleman of taste).

Being a small concern, the magazine seeks lots of reader participation to fill its pages and it actively encourages members to write articles and submit maps and solutions to BBC adventures.

Well that's really all my allotment of space used up, and in any case the dragon is getting hungry and needs to be fed. If you have any views, comments or questions, then do drop me a line c/o BEEBUG.

Personal Ads

For Sale (dirt cheap!): Aries B20 RAM board £45. ROMs: Wordwise Plus £25. Beebugsoft Toolkit £10, Pace Commstar £15, BS-DOS DFS £20, AMX Art £20, Printmaster £15. Discs (£5 each): Beebugsoft Discmaster, Magscan Bibliography, Superior Software Speech!, Clares Beebsynth, Replica 2. All documentation included. Tel: Mick 01-743 7523 anytime.

Toolkit Plus in original pack - complete - £18. Tel: Southampton (0703) 845104.

Aries B20 RAM expansion board and manual for sale. Wanted 8271 FDC chip. Tel: Denis, Doncaster 22478.

Wanted: Watford 32K RAM board and monitor (colour or green). Tel: Derby (0332) 556381.

Solidisk 128K RAM including utility discs £40. Tandy CGP115 £30. Ultracalc £20. Toolkit Plus £15. GXR (B) £15. CC Graphics ROM £15. Sleuth £10. Lord of the Rings (disc) £10. Tel: 01-960 8820.

Software: All as new and boxed at less than half price. AMX Mouse, ROM and Art disc £35. Sleuth ROM £12. Computer Concepts Interword word processor ROM £25. Gemini "LBO" ROM: 5 year diary/planner £9. Gemini DDD-Calc spreadsheet (40/80 track disc) £14. GCC (Cambridge) STARdataBASE ROM £20. Postage included. Tel: 051-677 1518.

Solidisk 1770 double density disc interface with DFS and ADFS ROMs £55. Wordwise Plus unused £25. Both for BBC. Tel: Gravesend (0474) 63503.

For sale: 30+ original games, from £1.99 or all for £50. Details from T.Verschoyle, 134 Hook Rise North, Tolworth, Surrey KT6 7JU.

BEEBUG magazine cassettes Vol.1 No.10 and Vol.2 complete £5. Image V.2 tape copier, unused £1. Sleuth ROM, as new, £10. Tel: (0920) 5406

Interword, in original packaging. Unwanted gift, £35 ono. Contact D.Wray, 31 Australia Road, Heath, Cardiff. Tel:

(0222) 621255 after 6pm or (0222) 874000 ext. 5378 during daytime.

Beebugsoft Romit as new £20. ATPL ROM board, fitted with 16K SWR £25. Solidisk 16K SWR and mini ROM cartridge with ROM software loader £20. Tel: Michael, Reigate (07372) 47791.

Wanted: Basic or machine code programs on 3.5 inch disc, Master Compact compatible - send me your list and price to: Quasar Software Club, c/o Cardito Giuseppe, PT Succ. 14, 25125 Brescia, Italy.

View A2.1 £25. Communicator 1.04A £25. CC Graphics ROM (latest) £19. Wordwise PLus (latest) £25. Printmaster (Epson) £19. Oxford Pascal £35. BEEBUG Toolkit £12. Spellcheck II £12. Acornsoft Forth tape £2. Atom Word pack £8. Watford Electronics 32K shadow RAM board £40. Tel: (0273) 475499 evenings.

BEEBUG Toolkit ROM, Masterfile I and II, Spellcheck, Sprite Utility, Design, Paintbox. Micro User Magic Sword, Children's adventure and Mini Office. IJK Candyfloss, Startrek, 3-D maze. Offers - will split. Tel: 061-766 9835.

BBC B, Acorn DFS, OS 1.2, Basic 2, ZIF socket, never gone wrong, with manual and tapes £210. Tel: 01-500 7476.

View version 2.1 ROM with 'Into View' and 'View Guide' manuals £25. Tel: Toddington (05255) 3511.

Solidisk 32K board with software (5 discs), also Commstar ROM £25 the lot. Tel: 043-879 514.

Watford CD400 twin double sided disc drives (40 track 400K) with PSU, as new, cost £230, bargain at £125. Cassette programs: Elite, Aviator, Mini Office, Fruit Machine/Honest Joe, Solitaire/ Knockout; £15 the lot. Tel: Harpenden (05827) 4065.

Phillips amber screen monitor and cable for BBC computer. Only purchased mid 1986. As new £60. Will consider swap for 2nd hand Acorn Teletext adaptor and

software. Wordwise 1.17, demonstration tape and instruction book £15. Tel: Medway (0634) 405720.

BBC B, issue 7, Acorn DFS, OS 1.2, 32K sideways RAM, Advanced User Guide, boxed, as new £250. Tel: Wokingham (0734) 784317.

Unwanted BBC games, ROMs and utility software at 1/2 price or less. Waterloo, Defender, Graphics ROM, Printmaster, Beebcalc, Beta Base and many others. All originals. Tel: St.Albans 30264.

Beebugsoft Sleuth ROM with manual £15. Computer Concepts Disc Doctor ROM with manual £15. Clares Replica III 40/80 track disc with instructions £8. P&p extra. A.Smith. Tel: (0924) 826483.

Complete Acorn disc interface including Intel 8271 chip and 1.2 DNFS ROM new and unused £35. Tel: 01-776 1033.

Software for BBC. Scrabble cassette £5. Collins complete starter pack, including 2 cassettes £5. Mini Office cassette £8. Beebugsoft Printwise 80 track disc £10. Tel: Chelmsford (0245) 252970 anytime.

BBC B micro, Basic II, Acorn DFS and speech system fitted £220. HRC Electronics external RAM/ROM unit with inbuilt power supply and 2 boards. Will hold up to 28 ROMs, and up to 32K sideways RAM £50. Tel: 021-745 7458.

BBC Master 128K for sale £390, Turbo co-processor £85, UFD disc drive £100 (2 months old). Phillips monitor £55. Master Bridge £10. Tel: (0865) 723989.

Torch Z80 second processor, includes CP/N, Perfect software with manuals. Cased with own power supply £185. Tel: Peter 01-857 7369 evenings.

BBC B, DFS, Cumana 100K disc drive, Magic modem and software, Joysticks, cables, manuals and a substantial amount of ROMs, software, discs and magazines. Offers around £325. Tel: Yateley (0252) 874801 evenings.

Software sale: ROMs - Advanced Disc Investigator £18, Brom £14, Max £13, STARdataBASE with registration card £30, CC Graphics ROM £10. Disc - The Hobbit with graphics £10. All as new in original package. Tel: (0527) 24153.

Acorn Z80 second processor plus extra software, boxed £225. Computer Concepts Interword £35. Disc Doctor £10. Master Reference manuals 1 and 2 £22 for both. The Master for High Fliers £6. Disk Programming Techniques for the BBC £4. Tel: Lawrence 01-455 1069 evenings.

Graphics ROM (Computer Concepts) with manual £15 ono. Tel: Southend (0702) 714298 (evenings) or write Luke Bosman, 116 Flemming Avenue, Leigh-on-Sea, Essex SS9 3AX.

Business Ads

Digit image analysis software for BBC micro and Grafpad2. Digitise shapes, point counter, make accurate measurements of irregular shapes: distances, areas, perimeters, Feret diameters, C of G, shape factors. Correct, print, display tracings, edit data, display, print as histograms with statistics. BBC B/B+/Master, disc and manual: £70. DISTIT disc analyses cell/particle distributions: £30. Institute of Opthalmology, Judd Street, London WC1H 9QS. Tel: 01-387 9621 extension 224.

Wine makers software. Written in Basic for the BBC model B, 'VINSOFT' contains essential data for 88 commonly used ingredients. It calculates alcohol yields and automatically corrects your recipe to suit your choice of wine style. Endorsed by leading winemakers. Send cheque for £8.95 to I.D.McFarlane, The Cayo, Shirenewton, Gwent NP6 6RL, for cassette or disc and explanatory booklet.

DYNAMIC DISCS

FROM BEEBUG

Our range of top quality 5¼" blank discs for the BBC Micro and Master 128 are now available at unbeatable prices.

These discs are from the UK's leading disc manufacturer, individually tested and guaranteed for life. They are specially produced and boxed for Beebug. Don't settle for anything less than the best.

Beebug members may claim a 5% discount on the price. If you are not yet a member please phone for an application form.

48 TPI DOUBLE DENSITY

		PRICE	MEMBERS
10	S/S	£9.90	£9.40
25	S/S	£23.00	£21.85
50	S/S	£48.00	£45.60
10	D/S	£10.42	£9.90
25	D/S	£26.47	£25.15
50	D/S	£50.00	£47.50
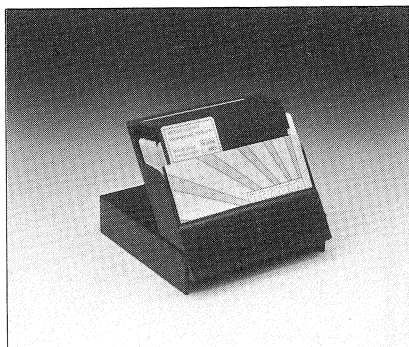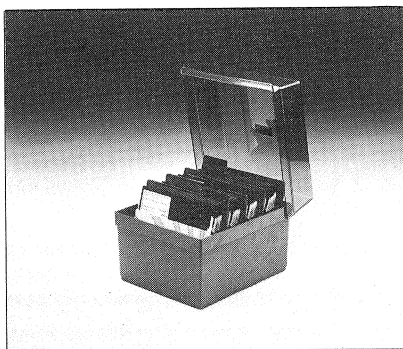
96 TPI QUAD DENSITY

		PRICE	MEMBERS
10	S/S	£10.42	£9.90
25	S/S	£26.47	£25.15
50	S/S	£50.00	£47.50
10	D/S	£10.90	£10.35
25	D/S	£29.00	£27.55
50	D/S	£53.00	£50.30

- Prices include FREE Storage Box
- Official Orders welcome
- ALL PRICES INCLUDE V.A.T.

- UK Postage 10's — £1.00; 25's & 50's — £3.75

5% MEMBERS DISCOUNT

ORDER FORM

Please send me_____Qty of _____ @ £_____ (Plus Postage £_____)

I enclose a cheque for £_____
or Please debit my Access/Visa No. [ ][ ][ ][ ] [ ][ ][ ][ ] [ ][ ][ ][ ] [ ][ ][ ][ ]

Name: _____Membership No:_____

Address: _____

_____

_____Post
_____Code: _____

NOTE: To claim members prices it is essential to quote membership no. If not yet a member add **£12.90** membership fee, to claim members discount.

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

DRUM ROLL
FOR RICE

Ian Waugh, an experienced enthusiast of computer-based music, swings to the beat of the Rice computer drum.

Product : Rice Computer Drums
Supplier : Rice Computer Electronics,
P.O. Box 2, Morecambe,
Lancs LA4 4PP.
Prices : £45 (software on cassette)
£47.50 (software on disc)
Add £12.50 for Trigger
Interface.

The Spectrum has its Specdrum, the Amstrad has its Amdrum and the Commodore has numerous software-driven sampled drum units. Now the BBC has a drum machine, too, in Rice's Drum Unit, a hardware device which plugs into the Beeb's user port, and with supporting software.

Unlike most other drum units, the Rice version doesn't use pure sampled sounds, but rather waveforms produced by a combination of digital and analogue techniques. Rice have actually developed formulae which can produce almost any kind of drum sound. The resulting waveforms are very low in noise, making them ideal for recording.

Like the other drums, sounds are produced by feeding the waveforms, which are stored in memory, through a D/A (digital to analogue) converter. This is the unit that plugs into the Beeb's user port, and a jack plug connects the converter to an amplifier or hi-fi system.

There are currently thirteen drum sounds including bass, snare, toms,
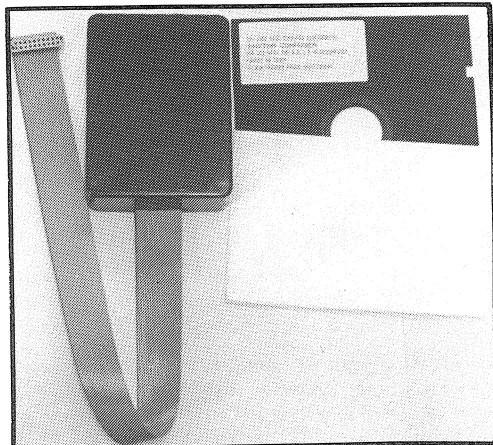
cowbell and hihats. The quality is excellent and belies the unit's modest cost. As the sounds are software-based it should be simple enough to add to the range, and Rice are already expanding the sound library.

The Rice Drums are programmed in a similar fashion to all other drum machines, both dedicated and software-driven. The first stage is to create a number of one-bar drum patterns, and the second stage is to chain these together to form a complete drum track. Drums are inserted into a pattern with the function keys and can be entered in real-time or step-time. The software has recently been updated and both versions are included with the package. In version one the drums play during step-time input, so you can hear your pattern build up, but they stop in version two, which is not as helpful.

Version one can store 64 patterns but each can hold a maximum of only 16 steps, which is just not enough for some rhythms. Version two can only store 32 patterns, but with 32 steps so it's far more flexible. Both versions can store 12 complete tracks of up to 80 patterns which should be long enough for an average song. In version two you can automatically insert the bass drum every 4, 6 or 8 steps.

Programming patterns is very easy and lots of fun. There are many aids to programming, such as clearing the pattern,

clearing drum lines, filling the hihat line, auto insertion of bass drum beats, and the selection of various grid references along the top of the screen. You can move quickly from pattern to pattern, and you can duplicate patterns. The editing facilities make the system a pleasure to use.

Tempo can be varied from 1 to 80 in software version one and from 1 to 255 in version two. The tempo setting affects the whole of the track, and patterns cannot have their own individual speeds, so tempo changes within a track are not possible.

The complete set of patterns and tracks are loaded and saved as one. Unfortunately, they are saved with the name 'DATA/' suffixed with a number from 1 to 8 which becomes pretty meaningless after your third or fourth save, and it restricts the number of files you can store on one disc to eight. The program
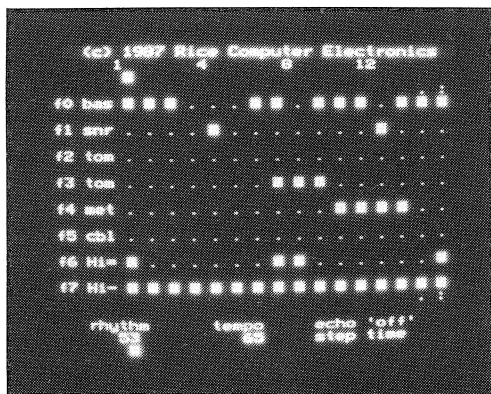
also switches off the key repeat, which makes certain operations tiring and frustrating. Perhaps it would have been better to leave such options to the user.

Version one has an echo facility to make the drums sound as if they are being played in a large hall - and it's very effective. This is missing in version two (shame!) along with the bright hihat sound.

An optional trigger interface is available which will allow you to exercise control over a Roland SH-101 or similar instrument. The trigger pulses required to drive it are set from a separate menu option in version one, but from the pattern screen in version two.

Although the system is easy to use, a few more words in the manual would not go amiss. There are lots of pre-recorded patterns and tracks for you to play around with, but a little more variety would have been nice. The software could do with a little tidying up - and Rice are keen to get feedback from users - but this doesn't detract from what must be one of the best value for money drum kits on the market. And as background noise is so low it makes it an ideal buy for penny-conscious home studio owners.

Rice are currently upgrading the package to include a built-in sequencer and bass guitar synthesiser which may appeal even more to impoverished musicians (aren't we all!). The drums and synthesiser will have separate outputs, another bonus for recording. This should be available soon for around £85.

```
3200 PROCClrBuffa
3210 FOR I%=50 TO 450 STEP 40:PBuff?I%=
24:NEXT
3220 ENDPROC
3230 :
3240 DEF PROCClrBuffa
3250 LOCAL I%
3260 FOR I%=0 TO 476 STEP 4:PBuff!I%=0:NEXT
3270 ENDPROC
3280 :
3290 DEF PROCPosChar(ch$,pos)
3300 LOCAL I%
3310 PROCGetFont(ch$)
3320 FOR I%=1 TO 8
3330 PBuff?(pos+4-I%)=PBuff?(pos+4-I%)
OR CBuff?I%
```

```
3340 NEXT
3350 ENDPROC
3360 :
3370 DEF PROCPrtLine
3380 LOCAL I%
3390 VDU 1,27,1,ASC("K")-QPlot,1,(1-QPl
ot)*480 MOD 256,1,(1-QPlot)*480 DIV 256
3400 FOR I%=0 TO 479
3410 VDU 1,PBuff?I%
3420 IF QPlot THEN VDU 1,PBuff?I%
3430 NEXT
3440 VDU 1,13
3450 ENDPROC
3460 :
3470 DEF PROCLeftAxis
3480 LOCAL I%
```
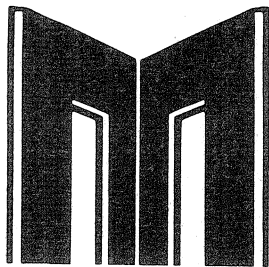
THE MASTER PAGES

Devoted to the Master Series Computers

This month's Master pages are from the pen of David Graham.

Please help to change this by sending us your contributions. Whether it is hints, reviews, programs or other articles on the Master and Compact, we would like to hear from you. **DEG.**

Contents

MASTER SERIES

Using The Master Editor

The Master's inbuilt Editor is a very powerful piece of software. But are you using it to the full?

The Editor which comes bundled with the Master 128 is a very powerful piece of software, yet by all accounts many Master owners make little use of it. Those few who do appreciate its potential often end up using it for every conceivable task, from editing programs to writing books.

In this short article, I will be looking at some of the features which make the Editor such a powerful and attractive tool, paying particular attention to its extensive search and replace facilities and to the use of embedded format commands.

EDITORIAL OVERVIEW

But where better to begin than with a broad statement of the Editor's features? The Editor works in all eight shadow modes of the Master, and can edit files of up to around 28K in length. Files appear on screen exactly as they are stored on disc. There is no file header, such as with InterWord, which means that you can produce and edit EXEC files, and even edit mode 7 screen dumps!

The Editor allows full text entry and editing in either 40 or 80 column modes; and while it does not offer a full WYSIWYG environment, its ability to run in 80 columns does make the generation of tables and so on a great deal easier than it is in Wordwise. Just like Wordwise, the Editor makes use of embedded commands for print formatting, though on occasion these offer greater power. For example it is possible to have three separate pieces of text on the same line, the first of which is range-left, the second centred, and the third range-right - very handy for page headers and letter headings.

Like View, the Editor provides a macro facility which may be used either for sets of embedded commands or paragraphs of text. Embedded commands are also used to provide a continuous processing capability which can link any number of separate text files, carrying over formatting commands and allowing contiguous page and section numbering.

The Editor is equally at home with Basic programs, but only after they have been converted to ASCII form. This is achieved by the command "EDIT" directly from Basic, and once a program has been edited, it must be converted back to tokenised form before it may be RUN or LISTed. This is achieved via Shift-f4 from within the Editor. Conversion is fast, with reasonably long programs taking only a second or two. As you may have heard, the

Editor makes little concession to Basic, and treats line numbers just like any other text. This is a little inconvenient in that it is therefore not capable of renumbering a program, nor taking you to the line number of your choice, but you soon get used to these restrictions.

For both programmer and text writer alike the Editor boasts an extremely powerful search and replace facility. Its only real limitation seems to be on the length of command string which it can handle. But more on search and replace in a moment.

Although the Editor is normally controlled through the function keys, it is possible to create so-called "command files" to make it perform any required sequence of tasks. This is useful for any series of operations which are frequently used, and might find application in the conversion of files from one format to another.

Finally, the Editor has a set of 10 numeric registers under the user's control. The contents of these may be inserted into documents in a variety of formats - including Roman numeral format - ideal for embellishing your Latin homework?

SEARCH AND REPLACE

The range of search and replace options is extensive, and as a consequence the rules for specifying strings are complex. But for simple operations, the syntax is straightforward. For example, the sequence:
    f5
    smith/JonesReturn
will replace all occurrences of "smith" regardless of case, with "Jones".

For more exacting operations the Editor offers a whole armoury of ambiguous character specifiers. Thus the following will find all occurrences of a single digit preceded by a space:
    f4
    Space#Return
To find strings of digits of any length, precede the "#" with a "^". Unimportant filler characters may be specified with "@" which matches any alphanumeric character, but not a space. Preceding it with "^" will match as many alphanumerics as possible in a string. In other words,

"^@" will match any word. The following will count the number of words in a piece of text:
    f5
    ^@Return
If you insert "/fred" before Return, it will, though not very usefully, replace each word in the text by the word "fred".

Because the specifier "#" will match any numeric character, it can be used to find line numbers in a program. As an example, the following string:
    ^#REM
will match program line numbers immediately followed by a REM. To increase the net so as to match REMs even if there are one or more spaces between the line number and the REM, we can use a second multiple character specifier "*". Thus:
    ^#*SpaceREM
The "*" specifier is used here in preference to "^", since the latter would require one or more spaces: "*" allows us to match zero or more spaces.

The above search string is still a bit loose in that it would also match the following:
    PRINT"4 REMARKABLE"
To avoid this we can specify that the number to be searched for must be preceded by a carriage return (signified by "$") plus zero or more spaces, so as to ensure that it is indeed a line number. The following sequence uses this ploy to remove REMs from a Basic program:
    f5
    $*Space^#*SpaceREM/$Return
though any text following the REM will not disappear until you return to Basic. Note though, that this sequence will not remove a REM which appears on the very first line of a program, nor will it remove REMs which are preceded by Basic commands on the same line.

The Editor is also capable of using logical OR operations in its searches. This is useful, for example, if we wish to remove all occurrences of Wordwise control codes from a piece of text, since these may either be terminated with a carriage return, or with a Ctrl-G character. Unfortunately, all the examples given in Reference Manual Part Two on the use of alternative specifiers (i.e. those where a logical OR operation is involved) are incorrect. Vital square brackets were apparently stripped out during the

printing process. If you wish to search for any occurrence of either 1, 3 or c, you should use the following:

    f4
    [13c]Return

and not

    13cReturn

as stated in the manual. To match any lower case letter other than d, use:

    [a-ce-z]

i.e. search for any character in the range a-c OR in the range e-z. The manual gives this as:

    a-ce-z

which would result in a search for a consecutive pair of characters, the first of which was in the range a-c, and the second e-z.

What makes the ambiguous search facilities of the Editor even more powerful is the ability to specify that the replace string should contain all or part of the search string. Two special characters & and % are used for this purpose, and we will take a closer look at these in a future issue.

PRINT FORMATTING

The Editor has a command set of some fifty embedded commands, most of which are specifically for formatting text on the printed page. They are listed on page S.4-1 of Reference Manual Part Two, and in an extremely useful Help option from within the Editor itself. To access the latter, press f8 (Print Text), then reply with "H".

```
         Major Format Defaults

Page Length    (.pl)      58
Line Length    (.ll)      76
Header         (.he)      blank line
Footer         (.fo)      2 lines
                          inc page no
Left Indent    (.in)      0
Page Offset    (.po)      0
Tabs           (.ta)      8,16,24,...96
Line Spacing   (.ls)      1
Justify        (.ju/.nj)  on

              Table 1
```

This extensive command set proves to be very flexible, and easy to use. There are even commands for setting up different headers and footers for left and right

hand pages, and the printing out in a variety of formats of the ten internal registers. The first of these, r0, holds the current page number; though as with all registers, it may be altered by the user to skip selected page numbers, and so on.

```
     Header for FX80

.co.>edhead1
.l160
.p155
.po5
.heHeader
.sp1
.en
.fo.sp2
.ce
Page .r0
.sp1
.ff
.en

        Table 2
```

The accompanying inset, table 1, shows the default settings of the major formatting parameters. To give an idea of the way in which some of these are used, table 2 gives a command header which will produce a left hand page offset of 5 spaces (po5), a line length of 60 and a header (called "Header") at the top left of each page, followed by a blank line. It then provides for 55 lines of text, followed by a 4 line bottom space with the page number printed centrally on the penultimate line. A form feed is then issued. This arrangement is well suited to the FX80. If you need to modify this for any reason, remember that the page length supplied (with pl) excludes the length of header or footer. Note also the way in which the top line of this file automatically supplies its own filename. See Master Hints for details.

HINTS AND SO ON

We have barely scratched the surface here, and yet we have already filled our present space allocation. A number of hints on using the Editor are included in this month's Master Hints, and if you have any hints of your own, or you are one of those fabled individuals who have written a book with the Editor, then drop us a line.

MASTER SERIES
Welcome Upgrades for The Master and Compact

Fix a bug on the Master's DFS, or install a DFS on the Compact — all you need is a new Welcome Disc.

MASTER DFS UPGRADE

As you may have heard there is an unfortunate bug associated with the two commands *CLOSE and CLOSE#0 on the Master's DFS on versions up to and including 2.24 (to check your version number type *HELP). Under certain circumstances these two commands will cause the wrong file length to be written back to the disc catalogue. This can be avoided to some extent by closing each file specifically (using CLOSE#n where n is the file handle).

In versions 2.25 and above the bug has been fixed, but of course, you cannot just unplug the DFS, not literally anyway, since all the Master's software is supplied on a single ROM. Moreover, this is sometimes soldered in. You can however get a software upgrade on disc, and load it into sideways RAM, and use *UNPLUG to remove the offending DFS. The software is supplied on the latest versions of the Master Welcome Disc, available at around £8 from all good stockists. Acorn also tell us that there are many other improvements on the new Welcome Disc.

DFS ON THE COMPACT

Unlike the Master 128, the Compact is supplied with the ADFS as the only filing system on ROM – the Master has both DFS and ADFS. But latest releases of the Compact Welcome Disc have a ROM image of the DFS, which may be loaded into sideways RAM, allowing Compact users the greater flexibility of an alternative filing system. Copies of the latest Welcome Disc for the Compact may be obtained directly from Acorn at a cost of £5.00 inclusive. Send to:

> Customer Services,
> Acorn Computers,
> Cambridge Technopark,
> 645 Newmarket Road,
> Cambridge CB5 8PD.

MASTER SERIES
Multiplying Function Keys

Not enough function keys for your needs? Why not pack three or more definitions into one?

Amongst last month's Master hints was one on doubling up on function key definitions by making a key perform a different action if pressed simultaneously with the Tab key. It was also promised to take this one stage further – well here it is.

The technique involves using the function key concerned to display a small menu of options from which the user makes his choice. The accompanying definition makes use of this principle. It prints a menu of three options:

P Progbackup
T Textbackup
O Oldbackup

These are in fact the names of three directories on a backup disc which I keep in drive one. Pressing P, T or O (upper or lower case) catalogues the corresponding directory. The user's response is detected with the GET function.

If you do not have enough space in your function key buffer for such a long definition, it could be shortened considerably. The screen blanking and the sound cue could be removed, and the directory names could be abbreviated using wild cards; and if you are really hard pressed, the ELSE can be abbreviated to "EL.". In any case, this example is intended only to illustrate the principle.

```
*KEY0 CO.0|MCO.7:P."P Progbackup"'"T Textb
ackup"'"O Oldbackup"'":;:SO.1,-15,20,2:a=
GET AND &DF:P.CHR$a:IFa=80 OS.(".:1.$.PROG
BACKUP") ELSE IFa=84 OS.(".:1.$.TEXTBACKUP
") ELSE IFa=79 OS.(".:1.$.OLDBACKUP")|M
```
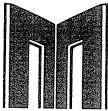
MASTER SERIES
Using CMOS RAM

How to access the Master's 50 bytes of battery-backed RAM, and how to use it to fingerprint your micro.

In the Jan/Feb issue, we squeezed in a routine for saving the contents of CMOS RAM to disc as an emergency backup for the Master's default settings. Here, we will take a closer look at this area of RAM, and provide a routine for reading it from Basic.

The Master's CMOS RAM is actually a part of the Hitachi HD146818 real-time clock chip which supplies the data for the Master's *TIME and TIME$ functions. The chip contains 50 bytes of battery-backed RAM at the disposal of the Master (that is to say Acorn, not you or me!). Acorn have recently re-allocated this as follows:

Byte no	Allocation
0-19	For machine configuration
20-29	For future use by Acorn
30-45	For ROMs 0-15 (one per ROM)
46-49	Reserved for the user

This allocation supersedes that given in the Reference Manual. Instead of bytes 30-39 being reserved generally for commercial software, and 40-49 for the user, there are now 16 bytes reserved, one for each ROM socket, and only 4 bytes left for the user. However, since ROM numbers 9-15 are machine-resident, and do NOT use CMOS locations above 19, this effectively leaves bytes 39-49 for the user - one more than originally allocated.

I want now to look at how these locations may be read and written to. Because the CMOS RAM does not form part of the Master's main memory map, it cannot be accessed by normal means, but Acorn have provided two OSBYTE calls for the purpose. These are 161 (read) and 162 (write). The second may be executed as a straight FX call. It has two parameters: the RAM location (1-49 since you cannot write to location zero), and the value to be written. Thus typing *FX162,49,255 <Return> will store 255 in location 49. Beware, however. Because the call will allow you to address any byte of the RAM,

you could upset your machine's default settings. You are strongly advised to take a copy of all settings using the Maintaining Status program from the Jan/Feb issue, before using this call.

Reading the RAM is a little more tricky because the full OSBYTE call must be performed. We have parcelled this up into a one-line function, as follows:

```
DEFFNcmos(X%):A%=161:=(USR(&FFF4)AND&F
F0000)DIV&10000
```

The call parameter is the CMOS RAM location number (0-49), and the function returns the contents of this location. Thus typing PRINT FNcmos(49) <Return> will print the value stored in location 49, and so on.

FINGERPRINTING

We have just enough space to suggest one application of CMOS RAM. It is to fingerprint your machine in such a way that software to be run on a number of different Master machines will be able to identify the machine on which it is running, and react accordingly. For example, it could avoid calling ROMs which might not be present on certain machines, or if used in the classroom context, it could prevent pupils from accessing the answers to work set for them, and so on.

The function listed at the end of this article may be used to identify a machine according to the contents of two bytes of CMOS RAM set previously with FX162. For example you might identify your own machine by giving the top two bytes (48 and 49) the ASCII values of your initials for the sake of argument. The function's parameters are as follows: L1 and L2 are the two locations (48 and 49 in this case), and X1 and X2 are their supposed contents. If the contents are correct, the function returns TRUE, and, if not, FALSE.

```
1000 DEFFNfinger(L1,X1,L2,X2)
1010 =(FNcmos(L1)=X1)AND(FNcmos(L2)=X2)
1020 DEFFNcmos(X%):A%=161:=(USR(&FFF4)A
ND&FF0000)DIV&10000
```

THE MASTER COMPACT

Although the Compact does not have the Master's clock, it can still retain 50 bytes of information in the form of an EEPROM (see Compact Welcome Guide p.B-10). The functions described apply equally to the Compact, but do remember that there is a limit, albeit quite large, to the number of times you can write to an EEPROM.

Another crop of hints for the Master and Compact.

THE EDITOR AND WORDWISE

As mentioned earlier, the Editor provides a very convenient way of creating tables that can be loaded directly into Wordwise. In such cases it is useful to be able to insert Wordwise codes from within the Editor. This is achieved as follows:

```
Wordwise          Editor
for f1 (green)    use Ctrl-B
for f2 (white)    use Ctrl-G
for Tab           use Ctrl-Shift-f0
```

For the latter to work, you must first execute *FX228,221.

512 TRANSIENT BYTES FREE

There is a 512 byte segment of memory within the filing system work space of the Master and Compact which may be used for machine code programs. According to the Master Reference Manual Part Two (Section 0.8-1), it is only overwritten by utilities such as *AFORM and *VERIFY on the Welcome disc. In other words the area is unused by all machine-resident software.

Its location is &DD00-&DEFF, and although this forms part of the Private RAM area, it does not need to be paged in before use. This is because the MOS firmware which shares this area is, somewhat unexpectedly, paged out during normal operations. On a Model B, by contrast, the MOS is permanently paged in (since there is no RAM at this address). To check it out, type:

```
A=?&DD00:P.A:?&DD00=A+1:P.?&DD00
```

If the result is two consecutive numbers, then you have RAM at location &DD00. If both numbers are the same, there is no RAM. On a model B with MOS 1.2, the result is 104, 104.

USEFUL FILENAME DEFAULTS

The Editor uses function keys f2 and f3 to load and save files. The filename is then supplied by the user, and upon pressing Return, the command is executed. But the Editor has two useful default options. If you press the Copy key followed by Return, when prompted for a filename, the save or load will be executed with the last-used filename.

A second default option allows greater flexibility. Any string on the top line of a document resident in the Editor which is preceded by ".>" is taken as the filename for a save OR load, if Return is pressed on a null filename. If you do not want this to appear when the document is printed out, it may be preceded by ".co". Alternatively, you might precede it with the word "Filename ", so that it is clearly printed and identified at the top of each printed document.

QUICK CHARACTER COUNT

The following sequence executed from within the Editor will display the total number of characters in a file:

```
<f5>
.<Return>
```

Don't put a slash (/) between the dot and the Return, or you will individually remove every single character from the file!

VDU STATEMENTS IN FUNCTION KEYS

The Master and Compact both allow VDU strings to be terminated with the vertical bar character "|", found to the left of the left cursor key. It has the effect of adding as many zeros as the VDU statement requires. Thus, while turning the cursor off on a Model B requires a horrendous:

```
VDU23,1,0;0;0;0;
```

Master Series users may use the almost elegant

```
VDU23,1|
```

but, unfortunately, not in function key definitions. The "|" character is used in function key definitions to indicate control characters. For example the sequence "|M" has the effect of Ctrl-M, which is the same as a carriage return. If you terminate a VDU statement in a function key with the vertical bar character, the operating system gets confused and issues a "Bad string" message. The only way to avoid this seems to be to stick to the old longhand method.

TIME FOR REAL GRAPHICS

Graphics packages for the Beeb always seem to attract attention, and the Real-Time Graphics System from newcomer Silicon Vision is no exception. The graphic and animated Geoff Bains articulates.

Product : Real-Time Graphics System
Supplier : Silicon Vision,
          47 Dudley Gardens,
          Harrow, Middlesex HA2 0DQ.
          Tel. 01-422 2274
Price : £79.95 inc VAT, p & p.

Probably more Beeb users have bought their machine for its graphics capabilities than for any other reason. One package that over the past year and a half has proved the value of the Beeb's display and speed is the 3D Graphics Development System from Glentop Publishing (see review in Beebug Vol.4 No.5). This package has now been enhanced by the author to produce the Real-Time Graphics System. This not only allows complex 3-D wire-frame objects to be created, but permits these to be animated at high speed.

Much of the new package is similar to the old. However, the Real-Time Graphics System is supplied on a 32K ROM (supplied as a single ROM on a carrier board) and no less than five discs. Not only do you literally get more for your money (ROM, five discs and a manual), but the capabilities are much improved too. The Real-Time Graphics System is not just 3D design and display software, but a complete real-time animation language.

There are two distinct parts to the Real-Time Graphics System. The first is the object model design software. This is largely unchanged (in operation, at least) from the previous package. The second is the real-time animation of 3D objects.

The designer is menu-based with constant checks against operator errors. Every move must be selected twice, first from a master menu and then again from a submenu which includes a backstep option. This gets very tedious. However, it does mean you are unlikely to accidentally wipe out a model full of data.
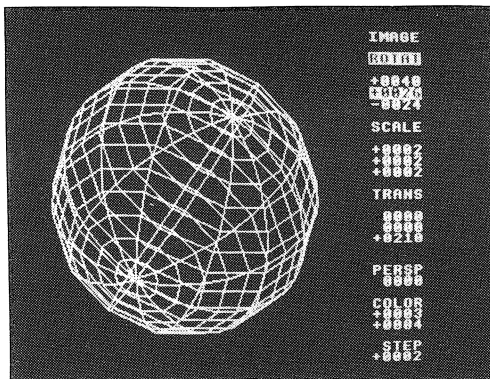
The system deals exclusively with wire frame projections on the screen. Such models are created in terms of 3D co-ordinates for each vertex of the object. These are entered into a table along with data to specify whether it is a move-to or draw-to point, a dotted or solid line and the colour used.

This is the hardest part of using the system. Entering any remotely complex object is not simple, and usually calls for the intervention of graph paper. It is a shame more help in this direction could not have been added. Something along the lines of Design Dynamics' Interactive 3D would be most useful. However, as we shall see, the way the model editor works reduces the data that has to be entered in this way.

Once the data is entered it can be listed, saved to disc, or edited in a simple step through fashion. The model can then be viewed. The viewing display is a mode 4 screen with a menu and data display down one side leaving a square display area. On this screen the model can be viewed in any position, from any angle, at any magnification, and with any perspective viewpoint.

With a suitable viewpoint selected and the other parameters set up as required, the view can be dumped to disc ready for dumping to a printer (an Epson compatible
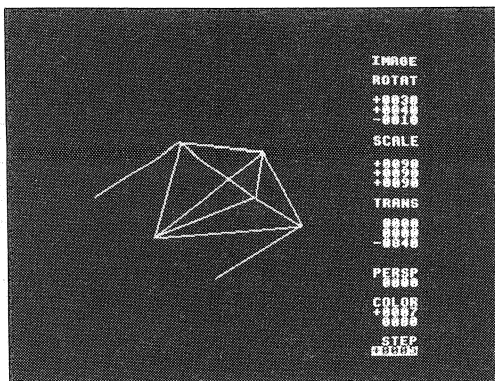
dump routine is included). The software usefully prompts for a single character to follow the core name of 'CADpic' for the saved screen. This means it is easy to cycle your model through a series of transformations saving a film frame at each stage.

The data for any small model can be translated and scaled or even rotated about an axis to form a 'solid of revolution' before saving it as a 'macro'. Just as assembler macros are used to build up complex assembler programs, so these are used to build up complex models. The data for simple shapes such as circles and squares is all you need enter into the data tables for many models. By judicious use of these shapes, the user can build up a more complex model with all the hard work done by the software.

When you have finally built up the shape you want and viewed it from every conceivable angle, you are ready to make use of the Real-Time Graphics System's other half. The ROM contains a number of routines to make use of the model data, and this is where the real-time nature of the software really comes in.

The ROM contains routines to plot perspective views of a model viewed from any angle and with any scaling or translation, all at breakneck speed. The Real-Time Graphics System ROM also has the facility to swap automatically between two mode 4 or mode 5 screens (there's no room for two screens in modes 0 to 2), drawing on one while displaying the other, and allowing very speedy and flicker-free real-time animation.

To draw a view of the model, the requisite viewpoint, scale, perspective value, and translation must be supplied to the ROM routines. This is where the system falls a little short. The Real-Time Graphics System communicates with Basic (or machine code) via a parameter block containing all the values defining the view of the model. To change the viewpoint the relevant value is changed either directly (a tedious process, but the only realistic one from assembler) or by using Basic variables such as xrot%, yscl%, and so on (only a little less inelegant).

When the Basic variables are used, a star command must first be issued to tell the system to get the new value from the Basic variable store and pass it to the system parameter block. This process can be made automatic but that slows down the calculation and drawing of the model a little as all parameters are transferred, unchanged or not.
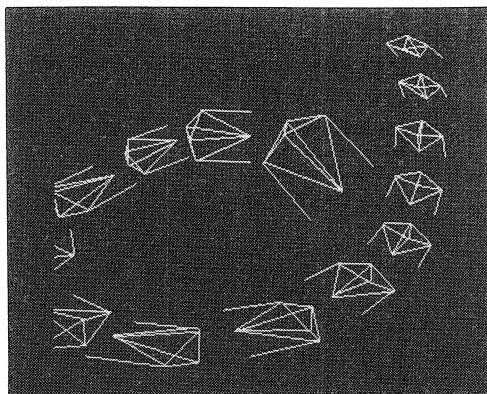
A method of directly passing parameters would be infinitely preferable. Something like *XROT 30 to give the value of 30 degrees to the X-axis rotation would be much easier to deal with and avoid the need for a wasteful Basic program header required to 'declare' all the Basic variables for the benefit of the ROM.

However, things are made easier by the numerous sample programs given with the Real-Time Graphics System. After a study of these, the technique of using the system soon becomes clear. The demonstration programs include several stages in the development of a program to

make a pyramid perform a complex revolving dance. Each aspect of the abilities of the Real-Time Graphics System are introduced one at a time.

The final result, which makes use of the system's own integer variables, automatic parameter transfer, reduced window size for faster clearing and dual screen operation, is very impressive. It is fast and smooth, reminiscent of the Aviator and Elite school of machine code games.

Things become more difficult with several models on the screen at once or with a more complex model. The limiting speed of the system is soon found. However, this is bound to occur with this kind of software. It is a generalised program for all applications and so it cannot take advantage of the short cuts, used by games for example, in specific cases.

For more specific use, the primitives used by the system are also available to the user. As well as the individual commands for each aspect of a model's transformation from 'raw' data to a perspective projection, the actual 3D co-ordinate calculating routines are accessible (though only via the parameter

block system), and commands are provided to select between dual screens for both drawing and display.

A set of 3D co-ordinates is passed to the system via some Basic variables, the relevant star commands issued to complete the transformation (having set up the parameters for that, of course), and a *CALC command issued. Further Basic variables then contain the transformed 3D co-ordinates and the projected screen co-ordinates for use as required.

For writing your own display routines or (from assembler) for passing data to a plotter or external display board, nothing could really be simpler.

The whole package is supported by a 100 page manual which confirms the package's history. This is split into two parts, the first dealing with the 'old' software - the model editor and associated programs - and a new section dealing with the real-time display language.

The first section is largely the Glentop package's manual (with all its faults) reproduced in a cheaper form. The second section is much better with far more in the way of useful examples and plenty of explanation of the demo programs. However, the whole documentation is lacking direction. It is only when you have read the whole work that you begin to see what the package is capable of and how to go about applying it to a problem. Then you can go back and learn the nitty gritty of actually using it. You get there in the end but it is hard work.

Nevertheless, this is a truly powerful piece of software. Although several aspects of the software reveal it as a program for 'programmers', it can, with effort, be used by comparative newcomers. For anyone interested in wireframe modelling or fast animation on the Beeb there is no other software to rival it. [B][B]
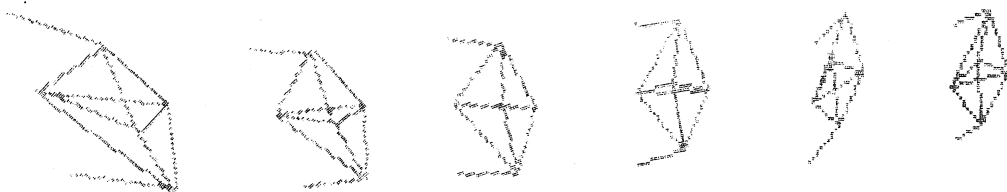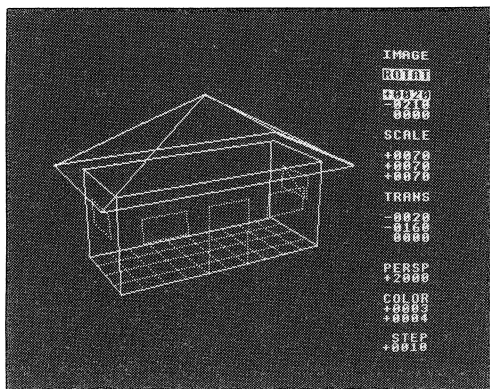
Installing
A Keyboard Blip

Bernard Hill, our sideways experts, presents an excellent keyboard blip routine for use with word processors. Next month he will coax this into sideways RAM — where else?

In a number of articles we have talked about writing star commands for sideways RAM and ROM. When a star command is issued, the operating system offers it in turn to each sideways ROM present in the machine. Any ROM accepting the command can then perform the appropriate routine. But suppose you want to put a routine into sideways ROM or RAM which is called not by a star command, but by redirecting a vector. An entirely different technique is called for, and it is the purpose of this pair of articles to look into the way in which this can be achieved.

The routine which we will install is one which causes an audible "blip" each time that a key is pressed. Whether you are a two-finger typist, or a touch typist, you will probably find this routine extremely useful if you do any word processing work. The blip gives an invaluable indication of whether you have pressed each key hard enough to register, or if you have inadvertently pressed a pair of keys instead of one. In fact, I think you will find that once you have been using the "blip" for an hour or two, your keyboard will seem quite naked without it.

In the first of these two articles we will look at the routine itself, and get it going in user RAM. Once all is operational we will move on, next month, to the problem of getting it to work from sideways RAM and ROM, and will provide the requisite header code. Owners of sideways RAM will be able to load in the routine directly, and those with access to EPROM blowers will be able to blow the code into ROM if they so choose.

BLIP FROM USER RAM

The accompanying listing is a version of the key blip program suitable for running from main memory. To get it going, type it in, and save it away before running it. When it is run you will be prompted to save the assembled version by copying the string which appears on screen. This will save the machine code under the filename:

BLIP

so use some other name for the source code - SBLIP perhaps? To load and run the machine code routine, just type:

*BLIP

or *RUN BLIP   (cassette users)

You should now hear a gentle "blip" at the press of every key.

If Break or Ctrl-Break is pressed at any time, the effect will stop, not because the machine code has been corrupted or removed, but simply because the vector which directs the operating system to the "blip" routine has been reset. There are a number of ways to reinstate this. But these should not be used if the "blip" routine is already working - or unpredictable results will occur.

If you are in Basic, the quickest way is probably to type:

CALL &900

You could even set up the Break key to perform this automatically as follows:

*KEY10 CALL &900|M

This again will only work from Basic; and the Break key can be cleared by pressing Ctrl-Break. Another way to reinstate the routine is to re-load and run the machine code with *BLIP or *RUN BLIP as described above. This will, very conveniently, work from within a word processor such as View or Wordwise without the need to exit to Basic. Pressing Break can provide a useful way to cancel the "blip" effect from within Wordwise or View.

MEMORY USAGE

As presented, the program resides in user RAM from location &900 to &936. This is no good on a cassette system if you are writing data to the cassette port, as you would be if you were word processing, and saving text files to cassette. The simplest solution is to assemble to &D00, which is clear on cassette-only machines.

To do this, just set the variable "code" to &D00 in line 100; and remember to CALL to &D00 not &900 if you are using this method to restart the routine after a Break. If you are using a Master or Compact, you may like to use the specially designated "transient program area". To do this set "code" to &DD00, again at line 100.

HOW IT WORKS
When the machine code is first run, just three lines of code are executed – lines 150-170. All these do is to redirect the so-called "read character vector" at addresses &210 and &211. This is re-directed so as to point to the "blip" routine. At the end of the "blip" routine, the program is directed back to the original contents of these two locations. This is performed by the JMP instruction at line 320. And incidentally, the way in which this is performed means that if you want the blipper to work on a machine with a different operating system, you will need to re-assemble it on the new system first.

Once the vector has been redirected, every time that a key is pressed, the operating system is sent to the "blip" routine, before executing its normal resident routines. The "blip" routine itself is very simple. It first preserves the major registers (lines 200-220) then flushes the sound buffer (line 230), then performs OSWORD call number 7, which is the machine code equivalent of the Basic SOUND command. This is accomplished in lines 240 to 270, and uses a parameter block set up in line 400.

If you wish to make a different sound, then just alter line 400. As currently set it executes a SOUND 0,-9,2,1, to produce a very short burst of sound on channel zero. If you are going to change the sound channel used, remember to change the buffer number which is flushed by the OSBYTE call on line 350. The reason why it is essential to flush the buffer is that under certain circumstances, such as when executing an EXEC file, the sound buffer will fill up, and slow down the speed of execution of the file, effectively reducing execution speed to some twenty characters per second. The OSBYTE call on line 230 flushes the sound buffer before each new blip is executed, and thus

prevents a queue from building up. This method is more effective than setting the "flush" parameter on the SOUND command.

NEXT MONTH
Next month we will, as promised provide the code necessary to put this routine into sideways RAM or ROM. But we will begin with a general look at the system vectors, and a discussion of the method provided by Acorn for using them to direct the operating system to sideways-resident code.

```
 10 REM Program Keyboard Blip
 20 REM Version B 0.1B
 30 REM Author  Bernard Hill
 40 REM Beebug  April 1987
 50 REM Program subject to copyright
 60 :
100 code=&900
110 FOR opt=0 TO 3 STEP 3
120 P%=code
130 [ OPT opt
140 :              \ set new vector
150 LDA #blip MOD 256 : STA &210
160 LDA #blip DIV 256 : STA &211
170 RTS
180 :
190 .blip :        \ main routine start
200 PHA            \ save registers
210 TXA : PHA
220 TYA : PHA
230 JSR flush
240 LDA #7
250 LDX #parms MOD 256
260 LDY #parms DIV 256
270 JSR &FFF1      \ do SOUND 0,-9,2,1
280 PLA : TAY      \ restore registers
290 PLA : TAX
300 PLA :
310 :              \ jump to old vector
320 JMP !(&210) AND &FFFF
330 :
340 .flush :       \ flush sound queue
350 LDA #21 : LDX #4 : LDY #0
360 JSR &FFF4
370 RTS
380 :
390 .parms :       \ SOUND parameters
400 EQUW 0 : EQUW -9 : EQUW 2 : EQUW 1
410 ]
420 NEXT opt
430 PRINT'''"Code Assembled. Now Type:"
440 PRINT'" *SAVE BLIP ";~code;" "~P%
450 PRINT'"then use *RUN BLIP or *BLIP"
460 PRINT"to run the code"'
```

Auto-Run Basic

Dr. R. D. Bagnall of Edinburgh University describes a method for auto-running Basic programs using only *LOAD.

When a Basic program runs on the BBC micro, essential details such as PAGE, TOP, LOMEM, HIMEM etc. are stored at the beginning of page zero (i.e. from address 0000) together with the number of the line currently being executed and the place on the line that has been reached.

Suppose we somehow arrange to *SAVE the relevant chunk of page zero as a program runs, and then when the program ends we *LOAD the chunk into page zero again. The surprising effect is that the computer is fooled into thinking that the command RUN has been given, and the program will 'autorun' from the point reached previously at *SAVE.

An example will make things clearer. First run the following short program:
```
10 *SAVE AUTORUN 0 1E
20 MODE 1
30 GCOL 0,1
40 MOVE 1200,0
50 PLOT 85,600,1000
```
At line 10 the program *SAVEs a file called AUTORUN from page zero, carrying the important information that line 10 is being executed, and that the end of the line has been reached. Now press Break, type OLD and simply *LOAD AUTORUN. As if by magic the program will run from line 20 (remember the end of line 10 has been reached) even though no RUN command has been given.

Note too that since the value of PAGE is also stored, it will be reset by the *LOAD to the original value, and there may be situations where this could be useful. More generally, the method has possibilities for program protection, against meddlers rather than against illegal copying. Since a program can be auto-run from almost any line, and often with a choice of PAGE, there is ample opportunity to include lots of important-looking but entirely redundant program lines, while the real starting line is tucked safely away deep in the program. With auto-run, there would be no obvious indication of where the program starts unless you knew how to decipher page zero.

The technique also has the potential of allowing a program to be saved at any point in its execution (with some limitations), and later restarted from that point. In most cases you will need to *SAVE not only zero page, and the program, but the whole of memory from the value of PAGE up to &7FFF. This will ensure that the current screen display, and all the data in memory is saved as well. You would also need to include as appropriate any function key definitions, or user-defined characters. The idea will work with simple programs, but it appears that the state of a program can only be satisfactorily saved if it is not executing a REPEAT-UNTIL or FOR-NEXT loop, and is not within a function or procedure at the time.

If you want to experiment with auto-run, the following tips may prove useful:

1. Do not extend the program after *SAVE, because the computer has already stored the value of LOMEM in the saved file, and will use memory above this for variable storage as the program runs.

2. The *SAVE line must not be the last line, because the program tries to auto-run from the line after this.

3. To auto-run a SAVEd Basic program, it must be loaded back into its original place in memory, or PAGE will not match the value stored in the auto-run file. The easy way to do this is simply to *LOAD a Basic program rather than the normal LOAD, then *LOAD its auto-run file, and hey presto!

Finally an extra suprise. Type in the example program again and press Break. Now simply *LOAD AUTORUN. Again the program runs! Try listing it, however, and it isn't there, because without OLD it shouldn't exist!

Hmmmmmmm...............

1st course

Using the Function Keys

This month, C.P.Yu takes a closer look at the function keys. Are you using them to the best advantage?

INTRODUCTION

The implementation on the emergent BBC micro of a set of user dedicated keys was something of an innovation. A few years on, a set of function keys now seems to be almost de rigeur on any new micro; and as past pages of the magazine clearly show, their uses are legion. In this pair of articles, we will be looking at some of the many ways in which the keys can be used. But as with all of the First Course series, we will begin with basics.

PROGRAMMING BASIC COMMANDS

The most common use of the function keys is for holding various Basic commands, and this provides us with a useful starting point. Programming the keys is easy: for example, you could set up key f0 to LIST a Basic program as follows:

    *KEY0 LIST

Once you have entered this, pressing f0 will cause the word "LIST" to appear on the screen. If you then press Return, your program will be listed. It is boring having to press Return to execute the contents of a function key, and we can include the Return character inside the key definition itself, using a special convention. This uses the "|" character (found to the immediate left of the left cursor key). On this convention "|M" indicates a carriage return. So if you reprogram key f0 as follows:

    *KEY0 LIST|M

pressing f0 will now cause the listing to take place directly.

STRINGING IT TOGETHER

Commands are easily strung together within a single key definition. For example, if you wish to make key f0 not only LIST your program, but set the LISTO options as well, you could use the following:

    *KEY0 LISTO7|MLIST|MLISTO0|M

This sets LISTO7 (to give a neatly tabulated listing), lists the program using this setting, then resets the LISTO option to its default of zero.

We can usefully take this example a little further. If you wish to see the listing in so-called paged mode (invoked by VDU14), where the Shift key is used to display one screenful at a time, rather than have it flash before your eyes, then use the following:

    *KEY0 LISTO7|MVDU14|MLIST|MVDU15|MLISTO0|M

Note the way in which VDU15 is used to cancel the effect of VDU14 once the listing is complete.

In fact, you can string together considerable numbers of commands in a function key definition, and with a little care, it is possible to install complete Basic programs in a single key. There are, however, two major limiting factors which must always be borne in mind. No single input string on the BBC micro (and therefore no single function key definition) may exceed 238 characters in length. Moreover, the total space allocated to key definitions, in the so-called function key buffer, is also restricted. On the model B it is 256 characters in length, while on the Master and Compact the area is expanded to 1024 characters; though in both cases a few bytes are used by the operating system.

ABBREVIATIONS

Because of the severe space restriction, especially for model B users, it is usually a smart idea to abbreviate function key definitions wherever possible. The BBC micro very sensibly stores all Basic keywords as single characters, so that whether you enter "P." or the full command "PRINT" in a Basic program, the computer stores this as a single byte (of value &F1). This is extremely economical. But this does not happen within function key definitions. They remain exactly as they are entered. For this reason, abbreviations are well worth using. The example given earlier could be entered more economically as:

*KEY0 LIST07|MV.14|ML.|MLIST00|MV.15|M
This saving of just 4 bytes is small
compared to savings which can sometimes be
made. It all depends on the keywords used.
ENDPROC for example abbreviates to "E."
with a saving of 5 bytes on this single
word. Of course, as you will appreciate,
using abbreviations makes for poor
readability, but the sacrifice is usually
worth it. A full list of abbreviations is
given on panel 9 of the BEEBUG Giant
Reference card. Another way to save a
little space is to use a colon to link
Basic statements or commands, rather than
using "|M" as we have done in the above
examples. This works perfectly
satisfactorily with all commands except
those such as LIST and LISTO which may not
be used inside a Basic program, and which
therefore require full "|M" separators.

USING VARIABLES
    Variables may be used in function key
definitions in exactly the same way as in
a Basic program. As a practical example,
this excellent definition submitted by
J.P. Carnell automatically saves a program
with a new version number each time that
f0 is pressed:
    *KEY0 Z%=Z%+1|MSAVE"PROG"+STR$(Z%)|M
To use it, first set Z%=0 (in fact Z% is
automatically set to zero on power-up).
Then press f0 each time you want to save a
new version of your program. The first
time it will be saved as "PROG1", the
second as "PROG2", and so on.

```
┌─────────────────────────────────────────┐
│  EPSON CODES FROM WORDWISE PLUS          │
│  Key Definition          Effect          │
│                                          │
│ *KEY0 |!!US|!"          Underline Start   │
│ *KEY1 |!!UE|!"          UnderLine End     │
│ *KEY2 |!!US|!!FI|!!UE|!" Draw Line        │
│ *KEY3 |!!ES33,0|!"       Pica (normal)    │
│ *KEY4 |!!ES33,8|!"       Emphasised       │
│ *KEY5 |!!ES33,19|!"      Elite            │
│ *KEY6 |!!ES33,30|!"      Double Strike    │
│ *KEY7 |!!ES33,63|!"      Wide Spaced      │
│ *KEY8 |!!ES52|!"         Italics On       │
│ *KEY9 |!!ES53|!"         Italics Off      │
│              Table 1                      │
└─────────────────────────────────────────┘
```

SAVING AND LOADING FUNCTION KEYS
    On a model B, the easiest way to save
a set of key definitions is to use the
following:
    *SAVE filename B00 BFF
where "filename" is any suitable name. It
works by saving the entire contents of the

function key buffer. The definitions can
be loaded back in again without disturbing
any Basic program which happens to be
resident, by using:
    *LOAD filename
This method is extremely easy to use, and
makes it possible for one set of key
definitions to call another. For example,
if you saved one set called "SETONE", and
a second set called "SETTWO"; and if you
arranged for key 9 of SETONE to be defined
as:
    *KEY9 *LOAD SETTWO|M
and key 9 of SETTWO to be defined as:
    *KEY9 *LOAD SETONE|M
then key f9 would always swap between key
definitions, doubling up the number of
instantly accessible keys.

Teletext Control Characters

Keyboard	Decimal	Hex	Function
Shift-f1	129	&81	Red Alphanumeric
Shift-f2	130	&82	Green Alphanumeric
Shift-f3	131	&83	Yellow Alphanumeric
Shift-f4	132	&84	Blue Alphanumeric
Shift-f5	133	&85	Magenta Alphanumeric
Shift-f6	134	&86	Cyan Alphanumeric
Shift-f7	135	&87	White Alphanumeric
Shift-f8	136	&88	Flash On
Shift-f9	137	&89	Flash Off
	138	&8A	nothing
	139	&8B	nothing

Keyboard	Decimal	Hex	Function
	140	&8C	Normal Height
	141	&8D	Double Height
Control-f1	145	&91	Red Graphics
Control-f2	146	&92	Green Graphics
Control-f3	147	&93	Yellow Graphics
Control-f4	148	&94	Blue Graphics
Control-f5	149	&95	Magenta Graphics
Control-f6	150	&96	Cyan Graphics
Control-f7	151	&97	White Graphics
Control-f8	152	&98	Conceal Display until next colour code
Control-f9	153	&99	Join Graphics characters
	154	&9A	Separated Graphics characters
	155	&9B	nothing
	156	&9C	Black Background
	157	&9D	New Background (previous foreground colour)
	158	&9E	Hold Graphics
	159	&9F	Release Graphics

    Master and Compact users are, perhaps
unexpectedly, less fortunate in this
respect. The function key buffer on the
Master Series machines is hidden away in
private RAM, and cannot be directly
accessed for *SAVE and *LOAD operations.
The way around the problem is to store the
key definitions in an EXEC file, and EXEC
this in when required. Using EXEC files on
the Master Series machines is treated more
fully elsewhere (Vol.5 No.6 p.42 and No.8
p.41). But essentially, it involves
creating a file using View, Wordwise, or
the Editor with the following format:
    *KEY0 LIST|M
    *KEY1 etc
Then to load the definitions into the
machine, execute:

*EXEC filename

or, since "EXEC" may be omitted on the Master Series, just:

*filename

This method still allows one set of keys to call another in the manner described above, provided that *EXEC is substituted for *LOAD in the definitions held in key 9.

```
┌─────────────────────────────────────────────┐
│  SOME USEFUL SINGLE LINE DEFINITIONS         │
│                                              │
│ *KEYn RUN|M                                  │
│ *KEYn CHAIN""|M          (cassette only)     │
│ *KEYn *CAT|M                                 │
│ *KEYn "LDA "             (assembler)         │
│ *KEYn "STA "             (assembler)         │
│ *KEYn longvariablename                       │
│ *KEYn PROCfrequentlyusedprocedure            │
│ *KEYn *DRIVE 1|M                             │
│ *KEYn *DIR S.main.subdirec|M*.|M (ADFS)      │
│ *KEYn *WORDWISE|M                            │
│ *KEYn MODE128:*WORD|M                        │
│ *KEYn *FREE|M  (free disc space on 1770)     │
│ *KEYn DIM P%-1:P.HIMEM-P%|M  (free mem)      │
│ *KEYn MO.Ø|M|S|@|G|@|@|@|S|A|@|@|@|@          │
│       (Set mode Ø black on white)            │
│                                              │
│               Table 3                        │
└─────────────────────────────────────────────┘
```

TEXT ENTRY

Function keys can just as easily be programmed with strings of commands or of frequently used text for use in a wordprocessor. For writing this particular article, I have programmed f0 as follows:

*KEYØ function key

In this case "function key" is the actual text which the key enters, and for this particular article it saves a fair bit of typing. I am using Wordwise, and in Wordwise, the function keys are accessed by pressing the Ctrl key simultaneously with the Shift key and the desired function key. This is also the case with View, though before this will work in View, you need to enable the function keys by executing:

*FX228,1

from command mode.

If I had wanted to insert a space before the two words "function key", I could not have simply added a space to the definition, since leading spaces are stripped out by the operating system. The way around this is to bracket the definition with quotation marks, thus:

*KEYØ" function key"

The quotation marks do not appear when the key is pressed, they just tell the operating system precisely where the definition starts and ends.

The function keys may also be used in Wordwise to set up various embedded commands. The accompanying table gives a set of key definitions to be used from within Wordwise to invoke a variety of Epson print codes. Note particularly the use of two groups of characters:

|!!

and

|!"

These insert the Wordwise green (f1) and white (f2) codes. You may also like to use one key in Wordwise to set up the header codes for a particular page format. This could even include your name and address for the top of a letter, though this would take up a lot of precious space in the function key buffer. Similarly, function keys can be programmed with View commands - see BEEBUG Vol.4 No.10.

```
┌─────────────────────────────────────────────┐
│      OTHER USEFUL KEY DEFINITIONS            │
│                                              │
│ To list the names of all assigned Basic     │
│ variables                                    │
│ *KEYn F.F%=&482 TO &4F4 S.2:A%=!F%A.&FFF     │
│ F:IF A%>&FF REP. V.1Ø,13,-512+F%/2:M%=A%     │
│ +1:REP.M%=M%+1:V.?M%:UN.?M%=Ø:A%=!A%A.&F     │
│ FFF:UN.A%<&FF:N.EL.N.:P.|M                   │
│                                              │
│ To LIST Basic program to the printer,        │
│ followed by a form feed                      │
│ *KEYn LISTO7|MWIDTH7Ø|MV.2|MLIST|MV.12,3     │
│ |MLISTOØ|MWIDTHØ|M                           │
│                                              │
│ To insert Epson American (#) font 3          │
│ lines from the top of Wordwise Plus text     │
│ (not placed at the top in case the first     │
│ command is a printer reset command)          │
│ *KEYn :CURSOR TOP|M:CURSOR DOWN 3|M:TYPE     │
│ CHR$2+|"ES82,Ø|"+CHR$13|M                    │
│                                              │
│ The Break key may be programmed (using       │
│ *KEY1Ø) just like the other 1Ø function      │
│ keys. When pressed it will perform its       │
│ normal Break, then issue the programmed      │
│ string                                       │
│ *KEY1Ø OLD|MRUN|M                            │
│                                              │
│              Table 4                         │
└─────────────────────────────────────────────┘
```

AND WITH NO STRINGS

So far we have only looked at using the function keys to generate pre-programmed strings of text. There is

All good programmers make extensive use of pointers. Surac shows a variety of ways of using this technique in your Basic programs.

Much programming is devoted to the accessing and manipulating of data in memory, and the right choice of technique can be important. This month, I want to discuss some of the options available, particularly through the use of pointers. Often, much depends upon being able to fit the requirements into some kind of regular pattern.

A typical example might arise when we want to determine the name of the day of the week from the day number, or the month from the month number. One technique is to use the variable RESTORE function.

```
1000 DEF FNmonth(m%)
1010 LOCAL month$
1020 RESTORE 1050+10*m%
1030 READ month$
1040 =month$
1050 DATA January
1060 DATA February
1070 DATA . . . .
```

From the month number (m%) the function returns the name of the month. With this method, each month must appear as a separate data statement, though it has the advantage that the routine reads the required month directly. Alternatively, we could pack the month

```
1000 DEF FNmonth(m%)
1010 LOCAL month$
1020 RESTORE 1050
1030 FOR I%=1 TO m%
1040 READ month$
1050 NEXT I%
1060 DATA January,Febru
     ary,March,April
1070 DATA May,June, . .
```

in one or two DATA statements by rewriting the function with a loop. This will be slightly slower, as it reads each month in turn until the required one is reached.

An altogether different technique involves reading the month names into an array at the start of the program. Any month can then be read directly.

```
100 DIM month$(12)
110 FOR I%=1 TO 12
120 READ month$(I%)
130 NEXT I%
140 DATA January,February,March,April,
150 DATA . . . . .
```

Once the array month$() has been set up, a simple reference such as:
    PRINT month$(5)
will automatically pick out the required month. The disadvantage of this method is that the data, in the form of the names of the months, is stored twice, once in DATA statements, and once in an array. If there were much data, this doubling of memory storage could be avoided by setting the data up in a disc file, and then reading direct from disc to the array (but it does take more effort to set up).

```
100 DIM month$(12)
105 F%=OPENUP("MONTHS")
110 FOR I%=1 TO 12
120 INPUT#F%,month$(I%)
130 NEXT I%
140 CLOSE#F%
```

In the examples above everything is quite easy because the months of the year form a nice continuous numerical sequence from 1 to 12. But what if this is not the case? There are various solutions, but let's first consider a different example. Students at college are normally allocated registration numbers. Although these may be assigned in sequence, the registration numbers of students on any one course are likely to be discontinuous. We'll take a simplified case of ten students with registration numbers as shown.

1024 Jones	1026 Brown	1027 Green
1033 Black	1034 Stamp	1036 Smith
1037 Evans	1038 Baker	1042 Patel
1044 Lloyd		

Even if we reduce the sizes of the numbers by subtracting 1023 from each we still have a range from 1 to 21 for ten people. If we used the RESTORE method we would have to insert dummy names for all the missing numbers in the range; if we used an array it would have to be more than twice as large as necessary, again with empty spaces.

One approach is to have two arrays, one for registration numbers and one for names. We can then search through the list of numbers until the required one is found, and use its position in the list to pick off the student's name:

```
1000 DEF FNname(No)
1010 LOCAL p%:p%=0
1020 REPEAT
1030 p%=p%+1
1040 UNTIL Number(p%)=No
1050 =Name$(p%)
```

The function assumes that the relevant registration numbers and names have been read, probably from a disc file, into two arrays called Number and Name$.

In effect, we have created an index in which we look up the registration number and pick off the student's name in the same position. Often the situation is more complex. In a book index, the topics are listed alphabetically, but the corresponding page numbers are in quite a different order. In our example we might wish to have both numbers and names in appropriate order. Clearly this is impossible with the two arrays as described. What is required is a third 'pointer' array.

We can introduce an array PNa() which can be set to point to the corresponding names. Given a registration number in position n, the corresponding name can be found at PNa(n), and the name itself will be Name$(PNa(n)). Thus we could re-order the names alphabetically, and provided the pointers are correct, we can go through the student list in registration number order picking off the names, or we can go through the names in alphabetical order – yes, we have no way of picking off the corresponding registration numbers. What is required is a second pointer array PNu() which enables a name to point to the corresponding registration number. Thus, with a name in position m, the corresponding number will be in position PNu(m),

and the number itself is Number(PNu(m)).

Here are two routines based on this. The first extracts the name corresponding to a registration number, the second (very similar) extracts a registration number corresponding to a student's name.

```
1000 DEF FNname(No)
1010 LOCAL p%,found%:p%=1:found%=FALSE
1020 REPEAT
1030 IF Number(p%)=No found%=p%
1040 p%=p%+1
1050 UNTIL found% OR p%>N%
1060 IF found% THEN =Name$(PNa(found%))
     ELSE =""
1100 DEF FNnumber(Na$)
1110 LOCAL p%,found%:p%=1:found%=FALSE
1120 REPEAT
1130 IF Name$(p%)=Na$ found%=p%
1140 p%=p%+1
1150 UNTIL found% OR p%>N%
1160 IF found% THEN =Number(PNu(found%))
     ELSE =0
```

Now you may well feel that everything has all become quite complicated, and you would be right. In fact, there is probably a better solution, still using four arrays as before. This time we don't worry about the ordering of names and registration numbers, but simply ensure that names and corresponding numbers occupy the same positions. That is, the nth positions in the two arrays contain the name and number of the same (nth) student. This data never changes in its relative order. The array PNu() is a set of pointers held in membership number order, and PNa() is a list of pointers in name order. We can then rewrite just four lines as shown.

```
1030 IF Number(PNu(p%))=No THEN
     found%=PNu(p%)
1060 IF found% THEN =Name$(found%)
     ELSE =""
1130 IF Name$(PNa(p%))=Na$ THEN
     found%=PNa(p%)
1160 IF found% THEN =Number(found%)
     ELSE =0
```

Although you may not feel that there is much difference between the two sets of routines, you will find that the second method of storing the data and pointers is much easier to use and to manipulate in practice. Complete demonstrations of both sets of functions are included on the magazine cassette/disc for this month. B

The Pitfalls of Christmas

You've done it again. Your Christmas issues always contain an excellent game (I thoroughly enjoyed Chinese Chequers the previous year) and this year, Pitfall Pete is no exception. However, we almost enjoy editing the screens and setting the puzzles as much as the play. It didn't take my daughter long to cotton on to the idea that if you filled the screen with diamonds you got the highest score. However, I did manage to create some interesting screens, and I enclose the best so far.

Ian Macfarlane

Other members have also sent us some interesting screen designs for Pitfall Pete. It is impractical to print any of these here, but we hope to include the best of all those received on a future magazine cassette/disc.

Model B for Ever

In the list of points on why I should renew my membership to BEEBUG you don't seem to mention that a growing amount of your magazine is being devoted to the Master. On the front cover it says clearly 'BEEBUG for the BBC Micro'. Please keep it that way. The Master may be in the BBC range, but it is quite different to the original Beeb that I am proud to own. So please make BEEBUG purely for the BBC micro, as it was not so long ago.

Andrew Brown

We have likewise received several letters expressing approval for our coverage of the Master. Despite Mr Brown's understandable concern, the Master is not taking over BEEBUG. We do now have 6 pages in each issue devoted to the Master series, but by taking 4 pages away from the supplement, the loss to model B owners is small. Indeed, there is often information that is applicable to other Acorn machines, (the use of EXEC files for example). And there is no intention to expand this feature further.

In our view there is room for all; the similarities between Acorn's machines are just as important as the differences.

What about the Compact?

A cry from the heart of a Compact owner! When may we expect 3.5" discs for the monthly magazine programs etc.?

James Manfield

We have been reviewing this situation for a while now, and we shall be introducing a monthly 3.5" magazine disc from the start of Volume 6, that is from the May issue. These discs are somewhat more expensive than 5.25" discs, and this is reflected in the overall costs. These are normally shown on the inside back cover of the magazine.

To mark the introduction of 3.5" discs and the start of Volume 6, we have a special offer on all new disc and cassette subscriptions, and renewals. For six month subscriptions we are offering one free disc or cassette, and for 12 month subscriptions there are two free discs or cassettes. Full details of this offer appear on the back cover of this issue.

ViewSheet Problems

Having been an avid user of ViewSheet for a number of years, and extremely satisfied with the performance and facilities provided, I recently upgraded to a Master and was pleased to discover that ViewSheet was included with the package. I was also pleased to discover that the ADFS was provided as standard and that I could take advantage of this very versatile system.

My pleasure subsided slightly when I discovered that I could not use my spreadsheet data files (V.VS1 etc.) without having copied them to a sub-directory of the currently selected directory (CSD).

When using the DFS, all array files were accessible from any directory. With the ADFS, a 'V' directory has to be created in each CSD directory. This can become a bit tedious and I wonder if any other readers can come up with a patch for ViewSheet/ADFS with the facility of creating array files in say the root directory, or more logically in the library directory, that could be accessed by any ViewSheet file in any directory.

J.Reid

INTERWORD LINE

With Wordwise Plus (versions 1.4A, 1.4C and 1.4E), you can draw an unbroken horizontal line across the page using:

<G>US<G>FI<G>UE<W>

and this would automatically conform to any changes in margins and line length. A similar result can be obtained in Interword as follows. Make sure that the line to be used is blank, and that there is a blank line immediately above and below it. Press <f3> insert marker, <Tab>, <f6> align right, <f3> insert marker, <cursor left> one space, <Shift f4> underline. You can now remove the blank lines above and below if you wish.

R.Sterry

OK BRAINS!

Since this IS the April issue, you might like to type the following question directly into your computer, pressing Return only at the very end (after the '|' character). It will prove just how close is the era of artificial intelligence!

*K.Ø What is the most frequent cause of parcels coming undone in the post in the UK?|

Now press Return.   D.E.G.

INTERWORD TAB

The Interword manual tells you that you can delete TAB markers from a ruler (as opposed to the TAB codes in the text), using <Delete>, <Copy> or <Ctrl A>. What it does not tell you is that you can also add them with the <Tab> key after positioning the cursor at the desired place on the ruler.

R.Sterry

DATE-STAMPING

To keep a record of when a disc was last backed-up, you could save a dummy file of zero length, with the date as the filename, e.g.:

*SAVE !7thDec Ø Ø

The use of the ! prefix will ensure that the date will usually appear as the first file. Master users can automate this by using the Master's in-built clock. Simply extract the day and month by using MID$, and then save this as the filename, using the following lines:

T$=MID$(TIME$,5,2)+"-"+
MID$(TIME$,8,3)
OSCLI("SA."+T$+" Ø Ø")

Of course you could save on file space and put the date as the title of the disc.

R.Sterry

WORDWISE DEGREE

To obtain a 'degree' symbol with Wordwise on an Epson compatible printer, the following command may be used:

<f1>OC27,83,Ø<f2>o<f1>
OC27,84<f2>

This works by printing an 'o' in superscript mode, but only in draft, not NLQ mode. Wordwise Plus users can substitute 'ES' for 'OC27,'.

T.Boyd

MAGAZINE TAPE/DISC

Users may wish to note that each program on the monthly magazine cassette/disc contains a title screen not published in the magazine. This uses lines Ø to 9 and 32700 onwards. To remove it, simply delete these lines.

The monthly magazine disc also has a pause between displaying the loading page, and actually loading the required program. To speed up this process, simply press the Shift key.

M.Way

ROMAN CLOCK

How about the following for a Roman clock?

```
10 INPUT "H,M,S";h,m,s:T
IME=100*(s+60*(m+60*h))
20 VDU12,23,1,0;0;0;0;:c
=&70:d=&72:!c=&4C565849:REP
EAT:FOR i=0 TO 1
30 VDU 31,10,10+i,141:PR
OCsh(TIME DIV36E4 MOD 24)
40 VDU58:PROCsh(TIME DIV
6E3 MOD 60):VDU58:PROCsh(T
IME DIV 100 MOD 60)
50 VDU 8224,8224;32:NEXT
:UNTIL FALSE
60 DEFPROCsh(n):PROCd(n
DIV 10,1):PROCd(n MOD 10,0)
:IF n=0 VDU45
70 ENDPROC
80 DEFPROCd(v,p)
90 IF v=4 VDU c?p,d?p:EN
DPROC ELSE IF v=9 VDU 73,88
:ENDPROC ELSE IF v=5 VDU d?
p:ENDPROC
100 IF v>5 v=v-5:VDU d?p
110 IF v=0 ENDPROC
120 FOR j=1 TO v:VDU c?p:
NEXT:ENDPROC
```

B.Hill 🅱

```
3490 PROCClrBuffa
3500 FOR I%=16 TO 479 STEP 4:PBuff!I%=&
18181818:NEXT
3510 FOR I%=50 TO 450 STEP 40:PBuff?I%=
&F8:NEXT
3520 PROCPosChar(MID$(NxtTim$,6,1),10)
3530 ENDPROC
3540 :
3550 DEF PROCVertGrad
3560 LOCAL I%
3570 PBuff!15=&08080808
3580 FOR I%=25 TO 480 STEP 8:PBuff!I%=P
Buff!I% OR &0808:NEXT
3590 ENDPROC
3600 :
3610 DEF FNGetch(str$)
3620 LOCAL key$
3630 *FX15,1
3640 REPEAT
3650 key$=GET$
3660 IF key$>="a" AND key$<="z" THEN ke
y$=CHR$(ASC(key$) AND &DF)
3670 UNTIL INSTR(str$,key$)
3680 PRINT key$;
3690 =key$
3700 :
3710 DEF PROCTitle(str$)
3720 LOCAL I%
3730 CLS
3740 FOR I%=1 TO 2
3750 PRINT TAB((31-LEN(str$))DIV 2) CHR
$141 CHR$131 CHR$157 CHR$132 str$ " " "CH
R$156
3760 NEXT
3770 ENDPROC
3780 :
3790 DEF FNMin(a,b)
3800 LOCAL m
3810 m=a
3820 IF m>b THEN m=b
3830 =m
3840 :
3850 DEF FNMax(a,b)
3860 LOCAL m
3870 m=a
3880 IF m<b THEN m=b
3890 =m
```

```
3900 :
3910 DEF FNTime(tim)
3920 LOCAL Hr$,Min$,Sec$,Dec$
3930 Dec$=RIGHT$("00"+STR$(tim MOD 100)
,2)
3940 Sec$=RIGHT$("0"+STR$((tim DIV 100)
MOD 60),2)
3950 Min$=RIGHT$("0"+STR$((tim DIV 6000
)MOD 60),2)
3960 Hr$=RIGHT$(" "+STR$(tim DIV 360000
),2)
3970 =Hr$+":"+Min$+":"+Sec$+"."+Dec$
3980 :
3990 DEF FNYes
4000 =(FNGetch("YN")="Y")
4010 :
4020 DEF FNRunTime
4030 LOCAL Pos,T,T$
4040 INPUT "? "T$
4050 T=0
4060 REPEAT
4070 Pos=INSTR(T$,":")
4080 IF Pos THEN T=T*60+VAL(LEFT$(T$,Po
s-1)):T$=RIGHT$(T$,LEN(T$)-Pos)
4090 UNTIL Pos=0
4100 T=T*60+VA
4110 =T
(T$)
4110 =T
4120 :
4130 DEF FNtimecomp
4160 :
140 TM2$=RIGHT$(TIME$,8)
4180 VDU2:FORti=1 TO LENTitle$
150 =1 AND TM$=TM2$
4200 NEXT:VDU3
160 :
4170 DEF PROCtitle(Title$)
4180 VDU2:FORti=1 TO LENTitle$
4190 VDU1,ASC(MID$(Title$,ti,1))
4200 NEXT:VDU3
4210 ENDPROC
4220 DEF PROCStar:VDU 7:ENDPROC
4230 DEF PROCOld:VDU 7:ENDPROC
4240 DEF PROCPlot:VDU 7:ENDPROC
4250 DEF PROCSave:VDU 7:ENDPROC
```

← 53 **First Course**

another way in which they may be used. If you press any of the function keys when Shift or Ctrl are held down, the keys do not generate a pre-programmed string, but a single character. The ASCII values of characters generated in this way are given in table 2, taken from the BEEBUG Giant Reference Card. You will notice that they fall in the range 139 – 153. This is intentional, and permits them to be used for generating mode 7 teletext control codes. The effect of these codes on a mode

7 screen is also indicated in the table.

With this information, you can very easily produce coloured program listings. If you enter the following:

    10 REM" <Shift><f3> some text
you will see that when this one-line program is listed in mode 7, the "some text" will appear in yellow. You can do the same in a print statement. Try:

    20 P."<Shift><f8> flashing text"
When you LIST this line, or execute it, while in mode 7, the text will flash.

FLITTY FLEA

Flitty Flea may be short on graphics but it's a real buzz of a computer game once you start playing. David Fryett wrote the program, T.F. Jones the accompanying description.

Flitty Flea is trapped within the bowels of the Earth. To escape, he needs to jump through ten underground caverns, collecting precious dice(!) on his way.

Platforms are scattered around each cavern to assist Flitty to jump onto higher dice and to escape from the cavern through the opening in the roof.

However, Flitty, like most of us, has his limitations. Jumping saps his strength and if his strength runs out before he escapes from any cave then poor Flitty will die. If he hits his head on the roof, a wall, a platform or a dice, this dazes him and his strength will diminish even further. Fortunately, on escaping from a cavern, Flitty has a new lease of life with renewed strength but this is always a little less than he started with in the previous cavern.

While there is no time limitation for Flitty, time does tick away and is recorded on his final escape as a target to beat the next time he becomes trapped.

The controls are 'A' for jump left and 'L' for jump right. The longer the key is depressed the higher the jump. The trick is to release the key when Flitty is over the dice or platform that he wants to land on because although his ascents follow a diagonal path, his descents are always vertical. This is easier said than done! To go through the roof opening, simply keep the key depressed.

Anybody who has serious strength problems may like to increase the value of S% at line 1010.

Be careful when typing in the data at the end of the program as any mistake could ruin the characters or screen display. The data between lines 2290 and 2380 is for the co-ordinates of each platform and the data between lines 2440 and 2660 is for the user defined characters. Also, to change the colour of Flitty and the platforms (useful on a monochrome screen), simply change the parameters of the VDU 19 command at line 130.
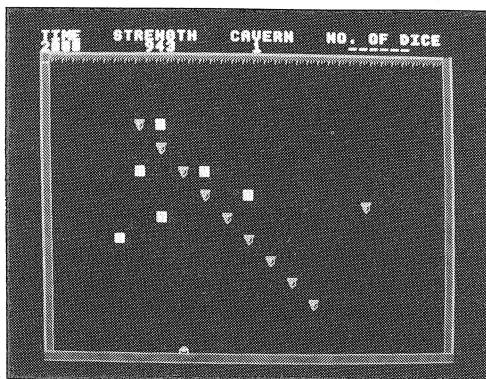
Get jumping, and good luck!

```
  10 REM Program Flitty Flea
  20 REM Version B0.1
  30 REM Author  D.Fryett
  40 REM BEEBUG  April 1987
  50 REM Program subject to copyright
  60 :
 100 ON ERROR GOTO 2690
 110 REPEAT
 120 MODE 1
 130 VDU19,1,5;0;19,2,2;0;19,3,7;0;
 140 PROCchars
 150 PROCscreen
 160 PROCcontrol
 170 UNTIL A%=1
 180 END
 190 :
1000 DEFPROCcontrol
1010 REPEAT
1020 C%=0:S%=999
1030 REPEAT:C%=C%+1
1040 COLOUR1:FOR F%=17 TO 23:VDU31,F%,3
,235:NEXT
1050 ST%=S%-C%*50:L%=608:H%=95:D%=0
1060 PROCgame
1070 UNTIL C%=10 OR A%<>0
1080 IF A%<>0 THEN 1100
1090 PROCwin
1100 UNTIL A%
1110 ENDPROC
```

```
1120 :
1130 DEFPROCdice
1140 VDU5:GCOL0,0:MOVEL%,H%-32:VDU242:V
DU4
1150 D%=D%+1
1160 PRINT TAB(28+D%,2);CHR$(235+D%)
1170 SOUND 1,-15,15,2
1180 IF D%=6 THEN COLOUR1:VDU31,17,3,24
4,32,32,32,32,245
1190 ENDPROC
1200 :
1210 DEFPROCplatoff
1220 FOR F%=6 TO 29
1230 PRINT TAB(1,F%);STRING$(37,CHR$32)
1240 NEXT
1250 ENDPROC
1260 :
1270 DEFPROCcave
1280 RESTORE C%*10+2280
1290 COLOUR 1:FOR N%=1 TO 10
1300 READ R%,U%:VDU31,R%,U%,229
1310 SOUND 1,-15,N%*50,1:NEXT
1320 COLOUR 3:FOR Q%=1 TO 6
1330 REPEAT:X%=1+RND(36)
1340 Y%=8+RND(19):J%=0
1350 J%=J%+POINT(X%*32,1019-Y%*32)
1360 J%=J%+POINT(X%*32,1019-Y%*32+32)
1370 J%=J%+POINT(X%*32+35,1019-Y%*32)
1380 J%=J%+POINT(X%*32-4,1019-Y%*32)
1390 UNTILJ%=0:VDU31,X%,Y%,242
1400 SOUND 1,-15,RND(150),1:NEXT
1410 PRINT TAB(10,2);ST%
1420 PRINT TAB(20,2);C%
1430 PRINT TAB(29,2);"------"
1440 ENDPROC
1450 :
1460 DEFPROCgame
1470 PROCcave
1480 VDU5
1490 IF ST%<1 THEN VDU4:COLOUR3:PRINT T
AB(10,2);"0   ":VDU5:GCOL0,2:MOVEL%,H%:VD
USH%:GCOL0,2:MOVEL%,H%:VDU243:VDU4:SOUND
3,-15,-1,9:PROCplatoff:PROCover
1500 IFA%>0 ENDPROC
1510 GCOL 0,2:MOVE L%,H%:VDU224
1520 REPEAT:COLOUR 3:VDU4:K%=0
1530 FOR F%=0 TO 150 STEP 10
1540 T%=T%+1:PRINT TAB(0,2);T%
1550 IF INKEY(-66) THEN VDU5:GCOL 0,0:M
OVE L%,H%:VDU224:GCOL 0,2:MOVE L%,H%:VDU
227:K%=1:F%=150:SH%=228:GOTO1570
1560 IF INKEY(-87) THEN VDU5:GCOL 0,0:M
OVE L%,H%:VDU224:GCOL 0,2:MOVE L%,H%:VDU
225:K%=2:F%=150:SH%=226:GOTO1570
1570 NEXT:UNTILK%<>0
1580 ON K% GOTO 1590,1690
1590 F%=1:VDU5:GCOL0,0:MOVE L%,H%:VDU22
7:VDU4
1600 F%=F%+1:ST%=ST%-1
1610 VDU5:GCOL0,0:MOVE L%,H%:VDUSH%:VDU
4
1620 IF POINT(L%-9,H%+32)<>0 OR POINT(L
%-1,H%+31)<>0 THEN ST%=ST%-50:T%=T%+30:V
DU5:GCOL0,2:MOVEL%,H%:VDU246:SOUND3,-15,
-30,5:GCOL0,0:MOVEL%,H%:VDU246:VDU4:GOTO
1780
1630 L%=L%-32:H%=H%+32
1640 IF D%=6 AND H%>924 THEN PROCplatof
f:SOUND 3,-15,-20,3:ENDPROC
1650 SOUND 1,-15,F%*5,1
1660 VDU5:GCOL0,2:MOVE L%,H%:VDU228:VDU
4
1670 IF INKEY(-66) GOTO 1600
1680 GOTO 1780
1690 F%=1:VDU5:GCOL0,0:MOVE L%,H%:VDU22
5:VDU4
1700 F%=F%+1:ST%=ST%-1
1710 VDU5:GCOL 0,0:MOVE L%,H%:VDUSH%:VD
U4
1720 IF POINT(L%+33,H%+32)<>0 OR POINT(
L%+33,H%+31)<>0 THEN ST%=ST%-50:T%=T%+30
:VDU5:GCOL 0,2:MOVEL%,H%:VDU246:SOUND3,-
15,-30,5:GCOL0,0:MOVEL%,H%:VDU246:VDU4:G
OTO 1780
1730 L%=L%+32:H%=H%+32
1740 IF D%=6 AND H%>924 THEN PROCplatof
f:SOUND 3,-15,-20,3:ENDPROC
1750 SOUND 1,-15,F%*5,1
1760 VDU5:GCOL0,2:MOVE L%,H%:VDUSH%:VDU
4
1770 IF INKEY(-87) GOTO 1700
1780 IF ST%>0 THEN COLOUR3:PRINT TAB(10
,2);ST%;"  ":PRINT TAB(0,2);T%
1790 VDU5:GCOL0,0:MOVE L%,H%:VDUSH%:VDU
4
1800 P%=POINT(L%+16,H%-33):IFP%=1 OR P%
=3 VDU5:GCOL0,0:MOVE L%,H%:VDU(SH%):VDU4
1810 IF P%=3 PROCdice ELSE IF P%=1 THEN
1480
1820 H%=H%-32
1830 VDU5:GCOL 0,2:MOVE L%,H%:VDUSH%:VD
U4
1840 GOTO 1790
1850 ENDPROC
1860 :
1870 DEFPROCover
1880 PRINT TAB(11,7);"G A M E   O V E R"
1890 COLOUR1:PRINT TAB(3,9);"Flitty has
lost all his ST":COLOUR2
1900 PRINT TAB(5,11);"You decoded ";C%-
1;" caverns"
1910 PRINT TAB(5,13);"and collected ";D
%;" dice."
1920 PRINT TAB(5,15);"It took you ";T%;
" light seconds."
1930 COLOUR3:PRINT TAB(5,20);"IS THAT T
HE BEST YOU CAN DO ?":A%=FNagain
1940 ENDPROC
```

```
 1950 :
 1960 DEFFNagain
 1970 COLOUR2:PRINT TAB(5,24);"Try again
y/n"
 1980 *FX15,0
 1990 G$=GET$:IFG$="N" OR G$="n" THEN VD
U22,7:PRINT TAB(19,12);"Bye."TAB(0,24);:
SOUND1,-15,99,5:=1
 2000 =2
 2010 ENDPROC
 2020 :
 2030 DEFPROCwin
 2040 PROCplatoff
 2050 COLOUR3:PRINT TAB(12,7);"WHHOOOPPP
PEEEEE!":COLOUR1
 2060 PRINT TAB(2,10);"All Caverns decod
ed. Well done."
 2070 PRINT TAB(2,12);"Flitty will remem
ber you for ever."
 2080 PRINT TAB(2,15);"It took you ";T%;
" light seconds."
 2090 IF T%<B% B%=T%
 2100 PRINT TAB(2,17);"Best time = ";B%;
" light seconds."
 2110 A%=FNagain
 2120 ENDPROC
 2130 :
 2140 DEFPROCscreen
 2150 VDU23,1,0;0;0;0;
 2160 T%=0:A%=0
 2170 COLOUR1
 2180 FOR P%=0 TO 38
 2190 VDU31,P%,30,231,31,P%,3,235
 2200 NEXT
 2210 FOR V%=3 TO 29
 2220 VDU31,38,V%,232,31,0,V%,233
 2230 NEXT
 2240 COLOUR 3
 2250 PRINT TAB(0,1);"TIME     STRENGTH
CAVERN   NO. OF DICE"
 2260 PRINT TAB(0,2);"0"SPC(9)"999"SPC(7
)"0"SPC(8)"------"
 2270 ENDPROC
 2280 :
 2290 DATA 9,9,11,11,13,13,15,15,17,17,1
9,19,21,21,23,23,25,25,30,16
 2300 DATA 11,11,13,13,15,15,17,17,19,19
,21,19,23,17,25,15,27,13,29,11
 2310 DATA 36,22,22,10,20,12,18,14,20,16
,22,18,20,20,18,22,20,24,22,26
 2320 DATA 20,15,21,16,22,17,21,18,20,19
,19,18,18,17,19,16,5,18,33,19
 2330 DATA 20,10,20,12,20,14,20,16,20,18
,18,14,16,14,22,14,24,14,2,23
 2340 DATA 14,10,24,10,14,15,24,15,14,20
,24,20,24,20,24,20,24,20,24,20
 2350 DATA 5,20,33,20,5,20,5,20,5,20,5,2
0,5,20,5,20,5,20,5,20
 2360 DATA 30,16,30,17,30,18,30,19,30,20
,30,22,29,17,28,18,31,17,32,18
 2370 DATA 10,10,12,10,14,10,16,10,10,12
,10,14,12,14,14,14,10,16,10,18
 2380 DATA 5,25,8,25,11,25,14,25,17,25,2
0,25,23,25,26,25,36,25,36,19
 2390 :
 2400 DEFPROCchars:RESTORE 2440
 2410 FORC%=224TO246:VDU23,C%
 2420 FORC2%=1TO8:READC3%:VDUC3%:NEXT,
 2430 B%=100000
 2440 DATA0,0,56,84,254,186,130
 2450 DATA0,0,16,40,120,240,192,112
 2460 DATA6,13,15,30,40,72,80,144
 2470 DATA0,0,8,20,30,15,3,14
 2480 DATA96,176,240,120,20,18,10,9
 2490 DATA255,255,254,122,122,54,60,24
 2500 DATA0,0,0,0,0,0,0,0
 2510 DATA255,85,170,85,170,85,170,255
 2520 DATA215,171,215,171,215,171,215,17
1
 2530 DATA235,213,235,213,235,213,235,21
3
 2540 DATA4,6,127,6,4,0,0,0
 2550 DATA255,255,170,85,170,106,72,64
 2560 DATA254,254,254,238,254,254,254,0
 2570 DATA254,190,254,254,254,250,254,0
 2580 DATA254,190,254,238,254,250,254,0
 2590 DATA254,186,254,254,254,186,254,0
 2600 DATA254,186,254,238,254,186,254,0
 2610 DATA254,186,254,186,254,186,254,0
 2620 DATA254,198,186,230,254,230,254,0
 2630 DATA0,0,0,130,186,254,84,56
 2640 DATA252,248,240,224,192,128,0,0
 2650 DATA63,31,15,7,3,1,0,0
 2660 DATA73,42,28,119,28,42,73,0
 2670 ENDPROC
 2680 :
 2690 ON ERROR OFF
 2700 MODE7:IF ERR=17 END
 2710 REPORT:PRINT" at line ";ERL
 2720 END
```

![B logo]

MULTI-COLUMN WORDWISE PLUS (BEEBUG Vol.5 No.6)

It has been drawn to our attention that there is a small correction needed to the Wordwise Plus utility program on page 20 of the above issue. The 19th line from the bottom of the left hand column should read:

    TYPE "|GBP"

and not as listed (the '|' character replaced with a space)

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.

HINTS

There are prizes of £5 and £10 for the best hints each month, plus one of £15 for a hint or tip deemed to be exceptionally good.

Please send all editorial material to the editorial address above. If you require a reply it is essential to quote your membership number and enclose an SAE.

Magazine Cassette/Disc

Fancy Lettering

Flitty Flea

Chart Recorder

All this for £3.00 (cass) £4.75 (5¼"disc) £5.75 (3½" disc)+50p p&p.
Back issues (5¼" disc since Vol. 3 No. 1, cass since Vol. 1 No. 10) available at the same prices.

	5¼" Disc	3½" Disc	Cass
Subscription rates (UK only)			
6 Months (5 issues)	£25.50	£29.50	£
12 Months (10 issues)	£50	£58	
Subscription rates (Overseas)	5¼" Disc	3½" Disc	
6 Months (5 issues)	£30	£34	
12 Months (10 issues)	£56	£64	£39

Prices are inclusive of VAT and postage as applicable. Sterli...
Cassette subscriptions can be commuted to 5¼" disc subscripti... £1.70 per issue of the subscription left to run. (3½ disc...
All subscriptions and individual o...

BEEBUG, Dolphin Place, Holywell Hill ... **AL1 1EX.**