AN ARGUS SPECIALIST PUBLICATION
SEPTEMBER 1988 £2.95

disk USER

1ST CHOICE FOR SOFTWARE

BBC MICRO
MODEL B
MODEL B+
MASTER 128
ALL DFS FILING SYSTEMS

WIMP IT – IKON SYSTEM
TELED – TELETEXT MADE EASY
MARVELLOUS MANDELBROTS
GRAPHCALC – STUNNING 3D
DYNAMIC DOODLES – GENERATOR
SPLASH – MODE 5 PIXEL EDITOR
WINDOW PLOTTER
BIRDS – TRACER DATAFILE

BEGINNER'S GUIDE
READERS' ROUTINES
MUSIC
ANIMATIONS
UTILITIES
NEWS
REVIEWS
DISCUSSION

NOW MONTHLY

By popular demand

disk USER

is

NOW MONTHLY

For INSTANT, ENTERTAINING and USEFUL software you only need to know about two keys

SHIFT and BREAK

- **Just BOOT for instant magazine action.**
- **Disk User – the unique interactive magazine on disk.**
- **FULL to brimming with BBC BASIC and machine code software**
- **AND graphics, demos and adverts on disk**

PLUS pages full of reviews, hints and tips and tutorial on how to use your disk system effectively.

In the 1987 Magazine Publishing Awards organised by The Publisher, Disk User for the BBC Micro won a second place certificate. The experienced panel of judges praised the "sheer value of the cover-mounted disk that formed part of the new publication's concept".

SUBSCRIBING MADE EASY
JUST TURN TO PAGE 25 . . .

SUBSCRIBE NOW!

We've made it easy to subscribe to Disk User. You can choose from a 6 monthly or 12 monthly subscription. Twelve disks full of quality software PLUS colourful magazine for just £35.40.

	6 months	12 months
UK	£17.70	£35.40
Europe	£21.60	£42.00
Middle East	£21.70	£43.40
Far East	£23.20	£46.40
Rest of the World	£22.00 or USA $32.00	£44.00 or USA $64.00

Airmail rates given on request

disk
USER

**Disk User
Number Eleven
September 1988**
Editor: Andrew Brown
Software Consultant: Matthew Fifield
Group Editor: Mark Webb
Advertisement Manager: Marcus
Advertisement Copy Control: Sonny Mehta

IN THE MAGAZINE

Disk Instructions 4
How to get the best out of Disk User – including 40 to 80 track conversion. Make sure you get the full picture by reading these pages before using your Disk User disk

Disk News 6
Latest releases and news for disk users

Discussion 7
Your letters column

Sector Zero 13
Two reader routines, and we announce colour printer winner

Beginners Guide to Disks 14
Francis Botto gives a guided tour to the BBC micro hardware

Transfer 21
Moving files to other disks

Birds Datafile 22
British Birds listed

Superfont clinic 23
Readers problems remedied

Starting Out – Assembler 28
The final installment of our guide to learning machine code, and building a powerful utility along the way

Services 32
Reader offers, binders and back issues

DISK DOCUMENTATION

ALL MODEL B, B+ and Master Series compatible

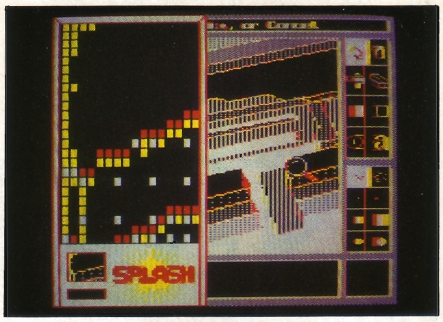
3D Graph 9
Amazing graphical display of mathematical formulae

Collectors Items 10
Animations for you to copy and keep. Build a collection from this entertaining A to Z of animations by computer artist Abbas

Splash 10
A mode 5 pixel editor with some neat touches

Dynamic Doodles 11
At last, we present this fast animated graphics generator

Extended Editing 13
Put Zip into your editing, and dash into your deleting



Star Scroller 13
An essential ingredient for that shoot-em-up you are writing

Windows made easy 16
Automate the creation of text and graphics windows for use in your own programs

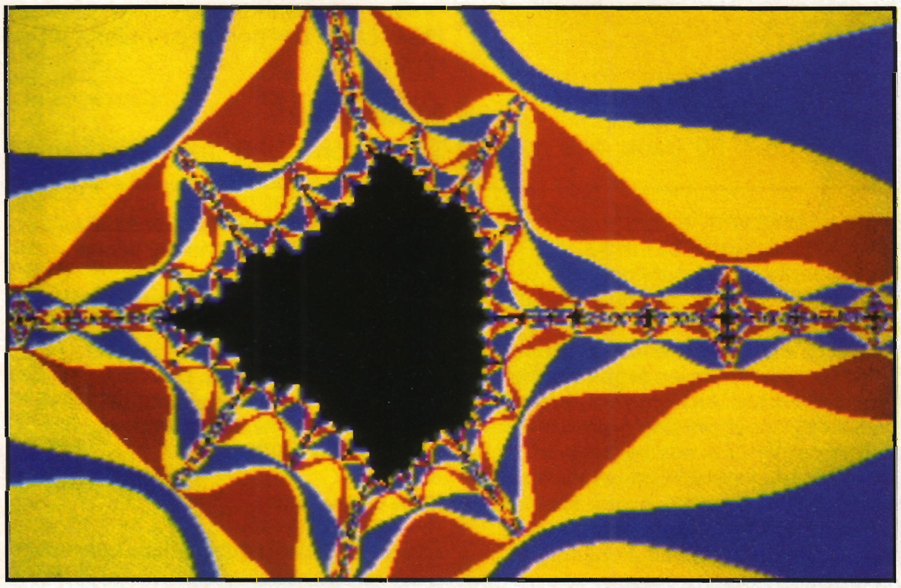
Ikon System 17
Our four page pull out this month is a full featured Ikon maker

Birds Datafile 22
A Tracer datafile on our feathered friends

Marvellous Mandelbrots 25
Explore the mysteries of imaginary numbers with this speedy routine

Teled 27
Painless, well almost, Mode 7 screen editing

**Disk User is supplied on a 40 track disk format and can be run without conversion on a 40 track drive.
If you have 40/80 switchable drives then make sure the drive is switched to the 40 option.
For 80 track only drive owners, a conversion program is provided – see Disk Instructions.**



ELECTRON COMPATIBLE:

**Dynamic Doodles
3D Graph
Splash
Windows Made Easy
Marvellous Mandelbrots
Ikon System
Star Scroller
Extended Editing
Birds Datafile**

DISK INSTRUCTIONS

Disk Instructions

To get the best from your copy of *Disk User*, please carefully read the instructions below. We have made *Disk User* able to run on a very wide range of systems.

All Users

Please make a **Backup copy** and keep the original in a safe place with a Write-Protect tab on. You should use this copy as your working copy, as many of the programs need to write to the disk, and doing this will diminish the usefulness of the original, and may not be possible anyway due to the 31 file limit imposed by many DFSs.

New Users

If you are a new user **Don't Panic!** ,first find out whether you have 40 or 80 track drive(s) attached to your computer (ask someone knowledgeable if you don't know). Then go to your User guide or Welcome Manual and read the chapter on filing systems. In particular find out how to use the *COPY command. Next re-read the section above **All Users** , and then go to the appropriate section dealing with your particular filing system and follow the instructions listed there.

Advanced Users

You do not need help to run Disk User, but do refer to the instructions for the filing system you are using, and **Don't forget to make a Backup copy.**

40 Track Drive Systems

Disk User is supplied on a 40 track disk so will work on any 40 track BBC Micro system (at least, any that we know of!) straight away. Remember to make a working copy before use.

40/80 Switchable Drives

If you have this sort of drive, you can use *Disk User* straight away with the drive switched to the 40 track setting; don't forget to make a copy for normal use. However, you may wish to copy the disk on to 80 track format, in which case, with a single drive, you should follow the instructions for 80 track systems.

With two switchable drives, or one switchable drive set to 40 track and an 80 track drive (or even a 40 track drive and an 80 track drive), you can easily copy *Disk User* on to 80 tracks; put *Disk User* into drive 0 (40 tracks) and a blank formatted 80 track disk into drive 1 (80 tracks) and type:

```
*COPY 0 1*.*<RETURN>
```

Here <RETURN> means hitting the return key. You can set the boot option to drive one by typing:

```
*DRIVE 1<RETURN>
*OPT 4 3<RETURN>
```

80 Track Drives

Because *Disk User* is supplied as a 40 track disk, 80 track disk drives have to double-step through the disk. Probably the most convenient thing to do is to copy *Disk User* on to 80 track format. This can be done in two ways.

If your filing system allows double-stepping, we recomend using the system's own command. As a general rule, built-in 40-to-80 track converters should be used where available; the documentation for your filing system or utility ROM will give full instructions, and we give suggestions for some better-known systems further on.

Not all filing systems have facilities for double-stepping; Acorn's DFS is one such system. To overcome this, a program called CHANGE is supplied on the *Disk User* disk in a section which can be accessed by 80 track drives.

Using CHANGE

Insert *Disk User* into an 80-track drive (or 40/80 switched to 80-track) and type:

```
*CHANGE <RETURN>
```

The program will prompt you to insert a pre-formatted blank 80 track disk when it is ready to write to it (you will have to swap back and forward between the two disks several times if you are using only one drive). Once this is completed, you can use the newly created 80-track version of *Disk User* and keep the original as the back-up.

Our suggestions on how to use *Disk User* on some popular DFSs now follow.

Master 128

This Acorn DFS has a software

**Disk User is published monthly on the third Friday of the month preceding cover date.
Advertising enquiries to: Disk User, Number One Golden Square, London W1R 3AB. ☎ 01 437 0626.
Editorial enquiries to Disk User, 6C Belgic Square, Off Padholme Road, Peterborough PE1 IXF ☎ 0733 53355.
Contributions should include full source code and instructions file on disk.
Payments are extremely competitive.
Distributed by SM Distribution Ltd., 6 Leigham Court Road, Streatham London SW16 2PG. ☎ 01 677 8111.
Telex 261643. Fax 01 677 0136**

**Typesetting and artwork by Island Graphics.
Cover design and design by the ASP Design Studio.
Colour by Croxons.
Printed by Chase Web, Devon.
Duplication by DisCopy Labs, 20 Osyth Close, Brackmills, Northampton, NN4 0DY ☎ 0604 760261.**

ISSN 0953-0606

double stepping mode for a 80 track drive. Set it with the command
***DRIVE 0 40** <RETURN> and then hit <BREAK>
Disk User will then work without any need for conversion. However this may not allow writing to the disk in 40 track mode; in any case, you should make a working copy, so copy to a 80 track disk.

DFS on Master Compact

The DFS is supplied as an image on some versions of the Master Compact Welcome disk (or is available from Acorn on disk) and this may be used in conjunction with a 5¼ inch 40 track disk drive to run Disk User. Please note that we **cannot** at present supply *Disk User* on a 3½ inch disk (if there is sufficient demand, we may be able to in the future).

Opus DDOS/Challenger 3

If you are using the Opus DDOS disk filing system or Challenger 1.0/DDOS then issue the command
***4080 AUTO** <RETURN>
or
***ENABLE 40/80** <RETURN>
and Disk User will work without any need for conversion.

Challenger 3

If you have the later ROM version Challenger 1.1 then issue the command
***OPT 8,1** <RETURN>
to achieve the same result. Disk User will work effectively from the RAM disk. Use
COPY 0 4 *.
***CONFIG 4=0 0=4**
***OPT 4 3**

Solidisk DFS

With the Solidisk DFS 2.1 and 2.0 you can set a software double stepping mode for a 80 track drive with the command
***ENABLE 80** <RETURN>
Disk User will then work without any need for conversion.

Watford DFS

The Watford DFSs also have a software double stepping mode for an 80 track drive. Consult your manual for the appropriate FX call or command. Disk User will then work without any need for conversion.

Disk failure

If for any reason your copy of Disk User will not work on your system then please carefully re-read the instructions given above.
If you still experience problems then:
1. If you are a subscriber, return it to:
INFONET LTD, 5 River Park Estate, Berkhamsted, Herts HP4 1HL.
2. If you bought it from a newsagent, return it to:
Disk User Replacements (BBC), Diskopy Labs, 20 Osyth Close, Brack Mills, Northampton NN4 0DY
☎ 0604 760261.

Please use appropriate packaging, cardboard stiffener at least, when returning a disk. Do not send back your copy of the magazine. Only the disk please.

ADFS Users

All files on this disk except IKON SYSYTEM work with the ADFS. If any problems should arise, examine the program listings and remove DFS only operating system commands (e.g. *DRIVE 0) and replace with the ADFS equivalent.
Note:– Disk User 11 fills a 40 track disk. Any software that may need extra disk space to save information must be copied onto a blank disk. ie Ikon System, Teled, Splash and Dynamic Doodles.

Editorial/Technical Enquiries

You can make telephone enquiries about *Disk User* on Wednesday and Thursday afternoons on 0733 53355 (please ask for *Disk User* Editorial). Enquiries in writing to the following address:
Disk User, 6C Belgic Square, Off Padholme Road, Peterborough PE1 IXF.

Disk User SEPTEMBER '88

Electron Compatible Files:–
Dynamic doodle demo.
Doodles Generator.
Wave Generator.
3D Graphs.
Splash.
Plotter.
Mandelbrot.
The Ikon System.
Star Scroller.
Extended Editing.

All change – 40 track to 80 track convertor. Files:–
CHANGE – Machine code file.
To use type:
*RUN CHANGE <RETURN>

Disk User – Disk magazine title page animation (yes we know it goes in backwards!).
Author: Abbas Files:–
P.RUNDISC – BASIC program
A.DISC – Machine code file

Disk Menu – Easy selection of the software.
Author: Matthew Fifield Files:–
DUMENU – BASIC program

Dynamic Doodles – Generate fast and colourful graphics displays.
Author – D.F. Catlin Files:–
DRIVER – Machine code file
SOAP – Data file
B.PATTERN – BASIC program
B.WAVE – BASIC program

Theme Music – Groovey tune to get you in the mood.
Author: Ian Waugh Files:–
LOADER – BASIC program
Theme – Data file

Ikon System – A picture speaks a thousand words (or functions).
Author: J. Bullock Files:–
IKONLD – BASIC program
EXPAND – BASIC program
X.IKONSYS – Compressed data file

TELED – Puts full teletext features at your fingertips.
Author: Neil Craven Files:–
TELED – BASIC file

3D Graphs – Mathematic equations that make interesting pictures.
Author: Ian Scott Brandon Files:–
GRPHBSC – BASIC program

Splash – Add a touch of fine detail to MODE 5 screens.
Author:David Ingleby-Oddy Files:–
SPLASH – Machine code file

New animation – Improved menu with the latest amimation.
Author: Abbas/J.C. Kenney Files:–
ALFABET – BASIC program
I.MENU – Data file
L.ALPHA – Data file

Plotter – Automates the fiddly setting of graphic/text windows.
Author: G.D. & G.P. Hawkins Files:–
PLOTTER – BASIC program

The Complete Mandelbrot – All the features needed to explore and examine.
Author: Paul Sparks Files:–
Mandel – BASIC program
Manmain – BASIC program

Star Scroller – Special effects to rival those of the movies.
Author: N.J. Fairbank Files:–
STDEMO – BASIC program
STARS – BASIC program

Extended Editing – SHIFT the cursors that little bit faster.
Author: Tim Campen Files:–
EDITING – BASIC and Assembler

Birds – A great help to Omithologists who use Tracer.
Author: M. Wilson Files:–
I.BIRDS – Tracer data file

DISK NEWS

Acorn News

Disk User is glad to see that Acorn are shedding their habitual cloak of security and becoming much more open to the real world. The editorial staff met *Peter Rennison* of Hi-Tech Public relations, Acorns latest PR men, at the recent Micro User show. We found him to be friendly, open to criticism, and aware of the real-world problems that still face Acorn. For example this month we have had a veritable avalanche of material concerning the various doings of Acorn Computers. Of course not all of it is relevant to Disk User readers, but we feel that too much information is infinitely better than too little. This month therefore we are presenting a brief roundup of news from Acorn.

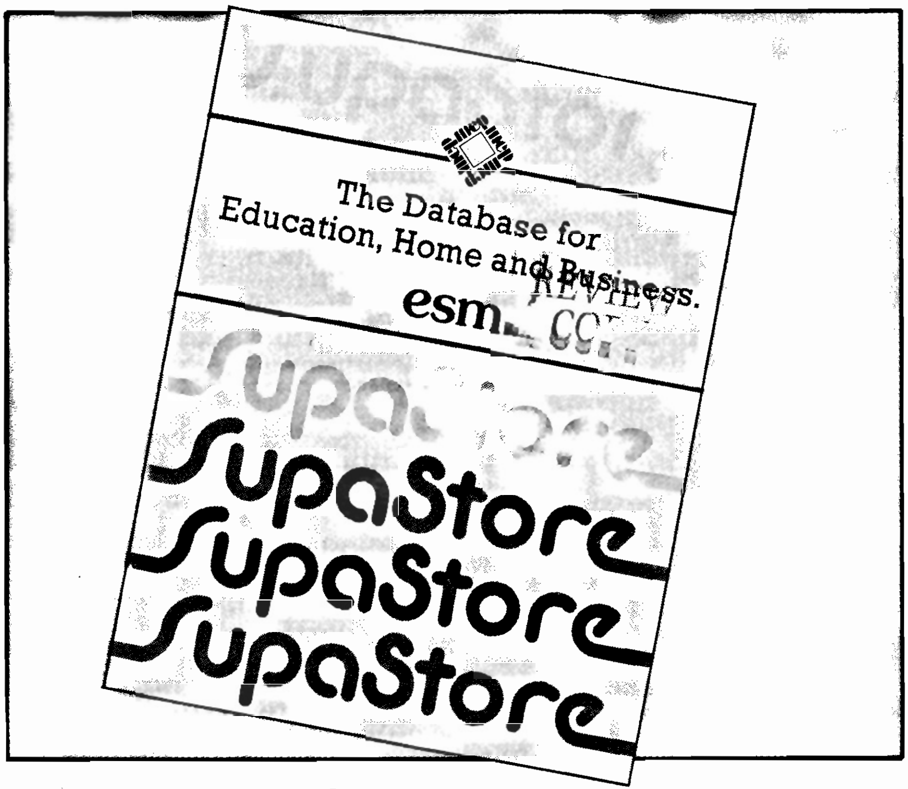
Nurses are to be given training in information technology using Master 128 computers, in a project estimated to cost £180,000. From September nursing and midwife teachers will attend CAL centres for three day courses in basic skills, such as spreadsheeting, databasing and wordprocessing. The centres themselves are new, and partly funded by the NHSS at a cost of £500,000.

Acorn are again holding a networking conference called, not surprisingly ECONET '88 on the 12th and 13th of September. Held at *Newman college* in the West Midlands the conference is set to cover many topics, but particularly the influence of RISC technology on network systems. On display will be Acorn's new ARM based fileservers.

Cambridgeshire Local Education Authority is putting its hand in not inconsiderable pockets to provide almost £300,000 worth of Archimedes computers to school children. 330 Archimedes computers will be installed in secondary schools throughout the county.

Acorn are linking up with Olivetti to put ARM based controller cards in ordinary PC's. The cards make laser printing possible using a PC plus *dumb* laser printer. The initial order is said to be worth £500,000 to Acorn, and MD Harvey Coleman anticipates the market volume for this kind of product to run into the hundreds of thousands.

Acorn Computers Ltd, Fulbourn Road, Cherry Hinton, Cambridge, CB1 4JN ☎ 0223 245200



Supastore can now do much more

ESM have produced an updated version of their popular database *Supastore*. A whole range of facilities, such as report generation, editing and deletion of records are updated for easier use. It is now much easier to combine records from other databases such as View, or Quest. Hopefully ESM are sending Disk User a copy of *Supastore Plus* for review, so watch this space for an in depth look at the new product.

Also from ESM is a package for early learning called *Numbercopter*. A follow up to the successful *Colourcopter*, Numbercopter contains eight programs that help kids to memorise,remember sequences and think logically.

ESM, 32 Bridge Street, Cambridge, CB2 1UJ ☎ 0223 65445

The Italian Connection

Our man in the cement overcoat reports.

The murky waters of Pasta manufacturer, violin case maker and sometime purveyor of computers to the mob, Olivetti has been stirred this week. Stories of an upheaval near to the top of this huge organisation have been confirmed with the appointment of a new chairman for British subsidiary Acorn Computers. The new chairman, *Elserino Piol* or Da Boss to his friends takes over from Mr Bruno (I love my Mama) Soggiu. *Harvey Coleman,* Acorn's MD welcomed the appointment, saying "It is extremely valuable for us to have a man with the vision of Elserino Piol as Chairman."

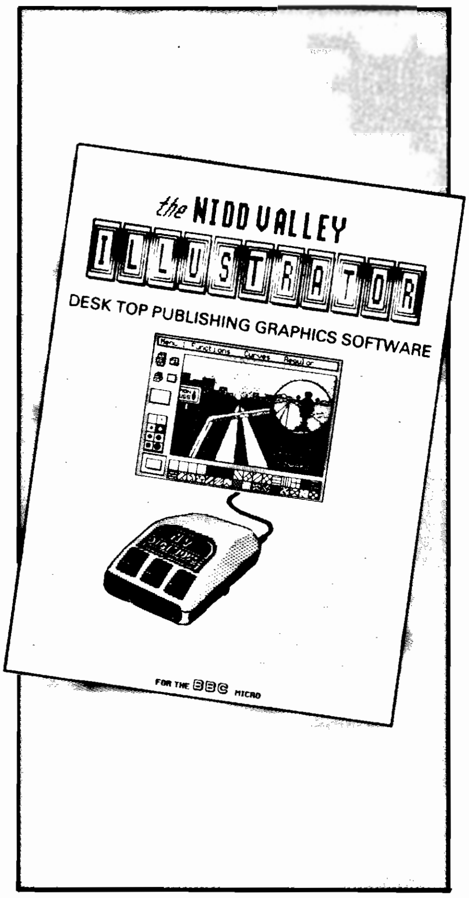
DISK NEWS...DISK NEWS...

The new Chairman has been a director of Acorn since 1985, and is Olivetti's Vice-Chairman for Strategy and Development.

Illustrator and Paintbox

We recently received review copies of the *Nidd Valley Micro Products* graphical software, Illustrator, and Paintbox. As opposed to the trend of combined ROM/disk approach Illustrator and Paintbox are completely disk based. I like this approach because it does not tie the user down absolutely to one machine. With a fast and powerful disk system like that on the BBC Micro it also is not necessary to ROM code just for speed alone.

We were unfortunately not able to fully test the software, as the mouse, which is really the heart of the product was not supplied. In their letter Nidd valley suggested that we might be able to borrow the one supplied to A&B Computing. I would like to point out at this time that Disk User is not in any way a subsidiary of A&B, and so companies who wish to have their software reviewed should not economise on review copies.

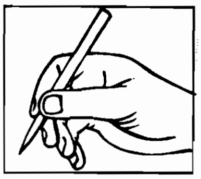


The software itself uses the standard WIMP interface for this type of product, with icons to the left on the screen representing functions such as filing, and selected disk drive. At the top of the screen are a list of options which, upon clicking the mouse, produce drop down menus. The bottom of the screen shows patterns available. These can be edited, as can icons, and brushes. Features such as a pixel editor, with variable zoom, flip, and rotate are included in the package, and the overall impression is one of power. The Epson print dump program included is versatile and sophisticated, while an Integrex colour printer can be used with the paintbox colouring software. The Paintbox program has a simpler feel to it, but has some very welcome features, including automatic save facility and save and load to a master file, thus eliminating the chance of overwriting existing art.

The cost of the Illustrator package together with the mouse is £49.90, alone it costs £19.90

**Nidd Valley Micro Products Ltd,
49 Thorp Arch Trading Estate,
Wetherby, West Yorkshire, LS23
7BJ ☎ 0937 844661**

DISCUSSION



£5 every month for our most informative letter

Electron variation

I know that Disc User is aimed at the BBC user, although you do say that some programs are Electron compatible, but how can we get rid of the Mode 7 codes? This month I expanded TRACER and found that it printed out as a double line of characters in the MENU, I know that on the BBC this prints out in double height characters and looks good, but on the Electron screen it looks terrible. Wouldn't it be possible to give us Electron users with disc drives either a short conversion on the disc or print out a list of variables.

There is another problem I am not sure about. What about the colour codes used to mark the field separators? Can the function keys on the Electron be used in the same way? Do all the other codes used on a BBC computer have to be changed on an Electron.

You have printed lists of variables on a few occasions the best one being Andrew Heptonstall's program QUICK COPY. Using this program on an Electron it comes to a halt after two or so passes as the RAM used overwrites the screen memory in Mode 6 and corrupts it. I worked out that the bottom of Mode 6 screens came at &6000 and the BASIC program came to &2000 so this left &4000 for the program to use on an Electron, so I changed the following lines in the program.

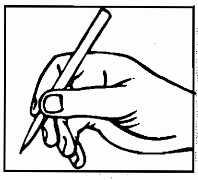
```
80 MODE 6
810 IF T% ‹=&4000 THEN PROC
load
820 IF (E%=C%+1 AND T%
‹&4000) OR T%=&4000 THEN
PROCsave
830 IF T% › &4000 AND D%=1
THEN big%=TRUE: D%=0: T%=0
840 IF T%›&4000 AND D%›1 THEN
T%=T%-T3% (E%-1): D%=
D%-1: E%=E%-1: PROCsave
```

This program is intended to copy whole discs in a few passes, but the one I use most is Dov Rosner's "SELCOPY" (Selective file copier (DFS)) which is one of the programs that works straight away on an Electron and is useful for copying just the programs that will work. There is one thing though, make sure that the disc you want to copy from is in the drive before you enter the final "copying from" command otherwise it will run through the disc you loaded it from first.

There is one thing I will add about DISC USER, it is great value for money, even though there are only a few programs that I can use, keep up the good work.

Yours faithfully
R.A. Wood
Essex

Electron compatibility gets better on Disk User all the time but in the case of something like Tracer, Mode 7 is pretty important. Getting rid of the CHR$ codes which create the Mode 7 displays is a laborious task without a BASIC editor, with which you can search and replace the CHR$ with a command of your choice (COLOUR perhaps) or just nothing at all. We'll give lists of variables and other useful information where we think it is practical for readers to make their own changes to the program.

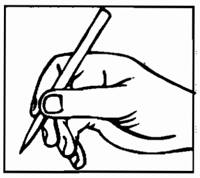


Tracer help

I am writing about the TRACER program. The programs are all now expanded on disc but I am having trouble with the CREATE option. I find that upon answering the prompt for a FILENAME the computer activates the disc drive and a 'DISC FULL' message is displayed. I found that by deleting line 785 (PROCsv) then it is possible to CREATE without trouble. Previously the only way to use this option was to load in an existing file eg. 'HELP-ME'. Upon then selecting the CREATE option and specifying a filename, it was possible to make a new file. I did find that due to line 785 (I presume) the HELP file had been saved under the new filename selected for the intended new file. I wonder if others have found this. Please find time and space to answer some of the queries made by those of us who while not claiming the title 'Computer Buff' do wish to progress in this field!!!

Yours Hopefully
Michael J. Cowie
Norbury

Thanks very much for this solution to what appears to be a *glitch* in the program. Anyone getting the "Disk Full" problem should give this a try. Just load up a copy of the BASIC program, CREATOR, delete 785 and resave. Copying just the program files to an empty 80 track disk should leave space free for the CREATE option to operate effectively.



In general

Before I continue with this letter, may I first congratulate you on an impressive magazine which encourages all that there still remains to be plenty of support for the old Beeb (though I myself own a Master 128!). Presently I do not purchase Disk User regularly, though I am seriously considering a subscription.

In the May 1988 issue of Disk User, on page 6 in the "Disk News" section you mentioned a new disk head cleaner made by *Parrot*. I am very interested in this product, but unfortunately you didn't seem to supply the address to obtain it! If you could pass it on it would be much appreciated. Also details are not supplied about the advertised book on page 15 – the "Creative Assembler".

Changing the subject a little, I remain a little concerned over the cover price of your magazine and it is mainly for this reason that I do not purchase DU regularly. I am aware that other magazines, such as Acorn User, A&B Computing cost £1.50, and that is without the disk and software supplied. However, disk prices have fallen drastically over the past few years and can now be bulk purchased for as little as 60p each. That means, if my calculations are correct, DU is about 70p overpriced (allowing for fluctuations in disk prices and the extra cost involved in placing the software on the disk).

Finally, I wish to get petty and request that ABBAS gets himself coordinated, and alters the Disk User title page so that the disk enters the drive the correct way around! I realise that this has been drawn to your attention by many readers, but if it is shown to those who are less experienced with the Beeb, surely this graphic will only confuse them.

Also, what about encouraging readers to send in their own tunes to be put on at the beginning of DU each month. That way quite a nice little library could be built up, retaining interest on what sounds are going to hit you this time you !BOOT up!

While talking about the DU disk in general I wish to praise the expansion program as this effectively lifts many of the problems experienced in supplying many files on DFS disks. Will a program be published in the future to turn readers' files into coded forms to enable them to distribute their programs efficiently? If not, **why** not?!

I would also suggest that you offer some of your programs for "downloading" over Prestel, if only to encourage more people to buy the magazine and whole disk full of software.

Yours formattingly
Matthew Ridd
Devon

P.S Keep the Master compatibility up!

Some interesting points here. Acornsoft's *Creative Assembler* is available from your local Acorn dealer or mail order from Vector Services. Call 0933 228953. Parrot are primarily a disk duplicator and do not sell their products mail order as far as we can tell. Pop into your local computer shop for the low-down on head cleaners.

When considering the price of Disk User you seem to forget that we have to duplicate the disks and package them for safe delivery. We carry vastly more software than our competitors in the most convenient form. Think how much a trained typist would take home for typing in the listings in some of our rivals! Disk User also delivers programs and data which no othr format could hope to match. It is, in a word, unique.

We've found in general that readers appreciate the user-friendly approach combined with a selection of original, exciting and downright useful programs. Thanks to enthusiastic and competent BBC programmers the standard continues to rise.

Quickies

Was there a Micronet demo on issue 4? Yes but because the original has sold out, the back issue is only available as software and manual. See our SERVICES section this month.

Can you do more to exploit the disk for games cheats, maps and so on? I appreciated the extra Repton screens on issue 4.

We can try! We have provided a comprehensive set of cheats but the main problem with maps and extra screens is that their size on disk is usually disproportionate to their value to readers. Watch out for our two **double-siders** in October and November when we will be using up every single K. It may be an opportunity for some of the things you mention.

Ian Scott Brandon

MENU PAGE ONE. OPTION THREE.

3D GRAPH

Help for all budding Einsteins everywhere as we reveal the beauty locked up in dry-as-dust mathematical equations

This program has been designed to calculate and display graphs of *two* variables (X and Y with result Z), rather than the usual *one* variable, (X with result Y). The results are plotted on screen in simulated 3D, and give a very impressive display. The program automatically scales the graph for the user. so all the user is required to do is input the equation, minimum and maximum values for the variables, step sizes, and the computer will calculate the rest for you.

A good example is the equation:

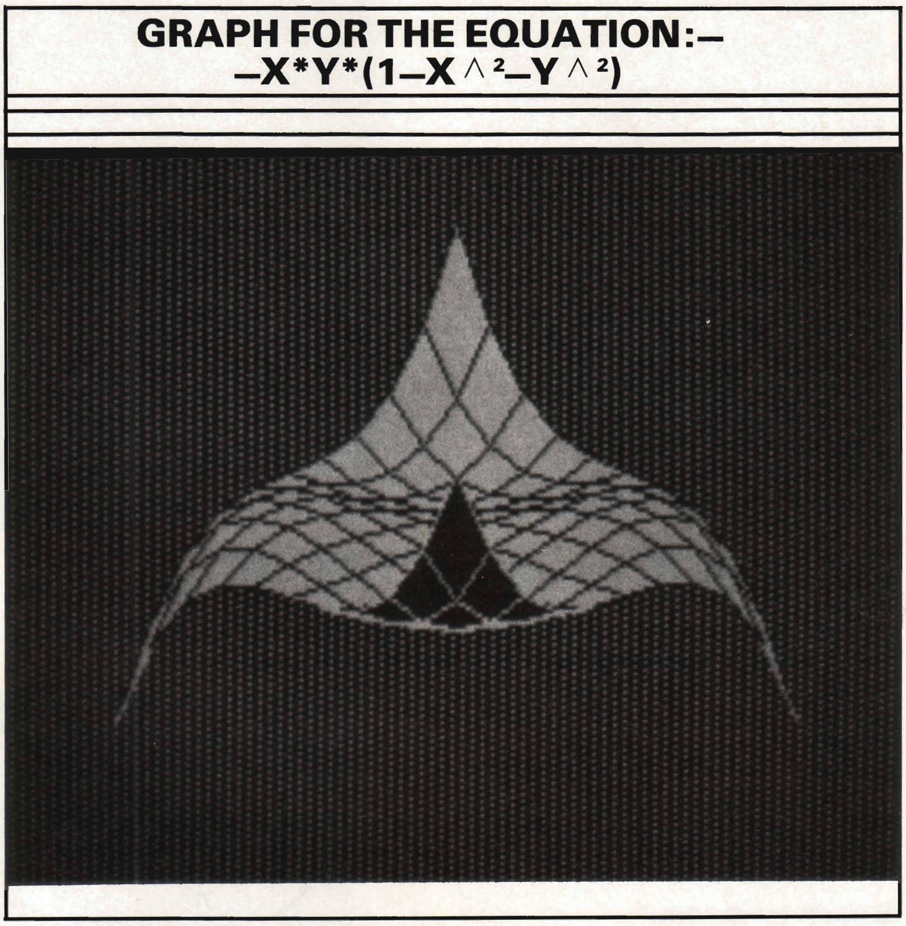
X*Y*(1-X∧^{2}-Y∧^{2})

with minimum and maximum values for all the variables being −1 and 1 with 0.2 as the step size.

Once the computer has finished calculating the co-ordinates, a basic graph of the equation will be displayed. It can be enhanced, by first pressing <H> to remove the hidden lines,and then <F> to display a filled/shaded graph. <N> returns to a normal display.

A screen dump can be sent to any standard Epson, or closely compatible printer, capable of double density graphics. It is started by pressing <P>. *Editors note:* Most dot matrix printers are Epson compatible, but the degree of that compatibility may vary. therefore perfect results cannot be guaranteed on all printers.

The program can be restarted by hitting <ESCAPE>.



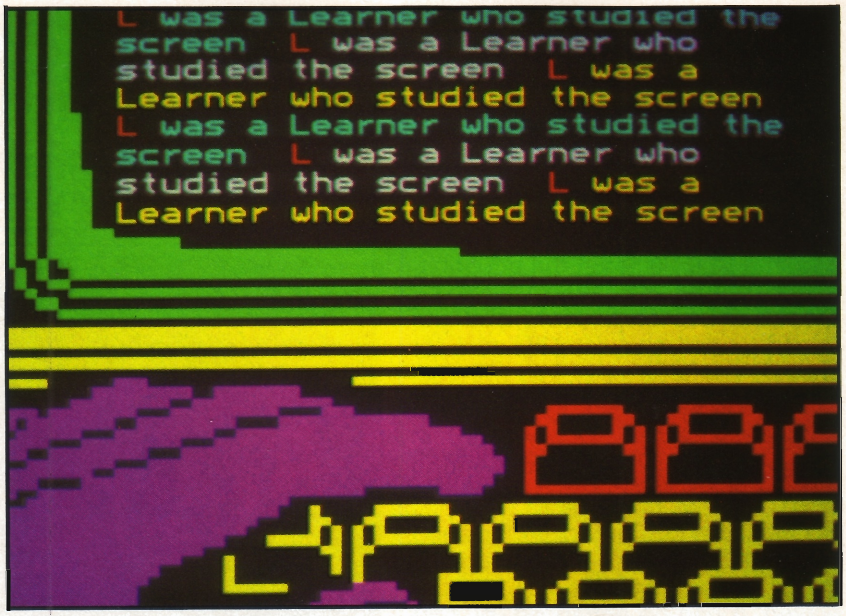
GRAPH FOR THE EQUATION:–
–X*Y*(1–X∧^{2}–Y∧^{2})

MENU PAGE TWO. OPTION FOUR.

Abbas

COLLECTOR'S ITEMS

Animate the alphabet with computer artist Abbas. Every month he animates a letter. Just choose this month's letter from the alphabet menu to see the action.



```
L was a Learner who studied the
screen L was a Learner who
studied the screen L was a
Learner who studied the screen
L was a Learner who studied the
screen L was a Learner who
studied the screen L was a
Learner who studied the screen
```

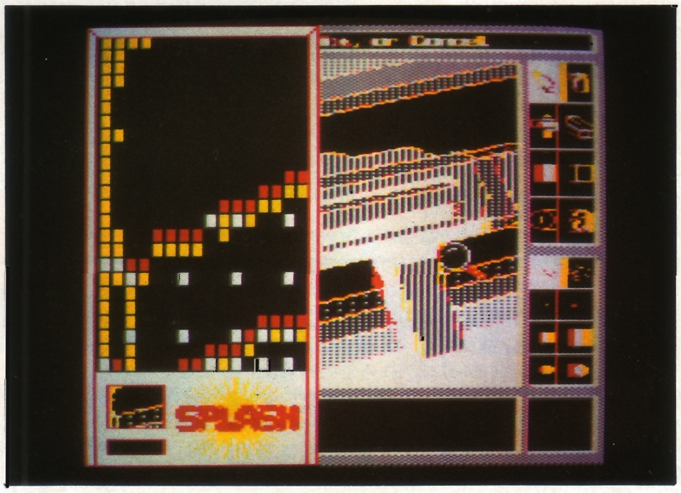
Instructions on how to use the menu for other letters were given in Disk User number seven. Letters A to K are available on back issues (see SERVICES this issue).

David Ingleby-Oddy

MENU PAGE ONE. OPTION SIX.

SPLASH

Make the most of Mode 5 with this pixel editing utility



You can use Splash by accessing it via the menu or by typing *SPLASH from BASIC.

Splash controls:

KEY DESCRIPTION
1 LOAD ANY MODE 5 SCREEN
2 SAVE CURRENT SCREEN
3 ENTER SCREEN EDITOR
4 EXIT PROGRAM
5 ENTER * COMMAND

In edit mode (magnify), a magnifying glass appears on screen. Control with:

Z – LEFT
X – RIGHT
: – UP
/ – DOWN
M – RETURN TO MAIN MENU
K – LIST KEYS
RETURN – ENTER PIXEL EDIT MODE

In pixel edit mode a square cursor appears at the bottom left of the window. Control with:

Z – RIGHT
X – RIGHT
: – UP
/ – DOWN
SPACE BAR – SET PIXEL

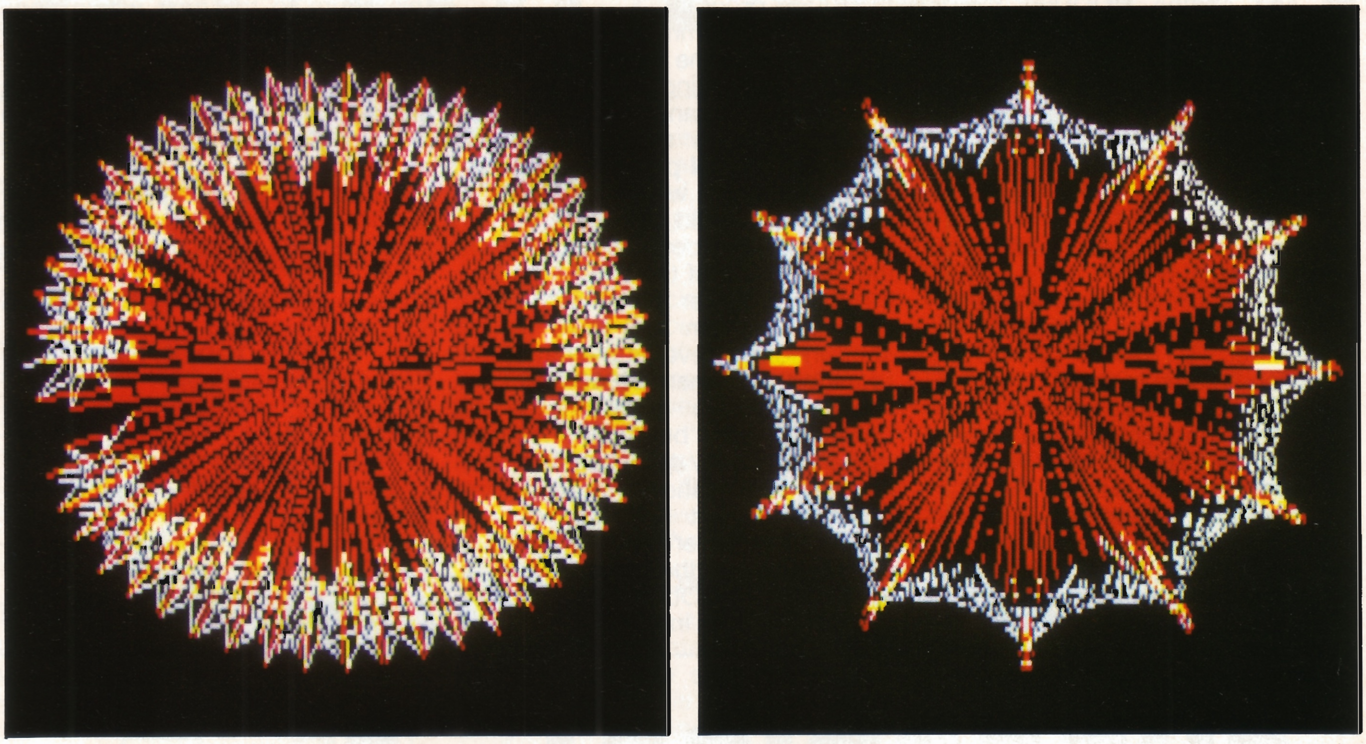
K – KEY MENU
M – MAIN MENU
1 – COLOUR 1
2 – COLOUR 2
3 – COLOUR 3
0 – COLOUR 0

The currently selected colour appears in the bottom left-hand box

V – CHANGE CURRENTLY SELECTED COLOUR EG DOES A VDU 19
D – DEFAULT MODE 5 COLOURS
V – UNDO PIXEL CHANGES
SHIFT – RETURN TO MAGNIFY MODE

Des Catlin

DYNAMIC DOODLES



Stunning psychedelic action with Dynamic Doodles

What is it?

Dynamic doodles can best be described as a simple method of displaying animated graphics smoothly and quickly. When a BASIC program draws patterns even on the BBC they can be painfully slow. So here is a simple and very effective method of speeding them up. Instead of BASIC controlling the graphics output this program simply records the output on disk. Then a machine code graphics driver replays the recorded pattern at a much greater speed. The results are stunning.

Workspace

The DYNAMIC DOODLES files are simply **EXEC** files **SPOOLed** from BASIC programs. If you are not sure what is meant by this try the following...

MODE 5 <RETURN> ***EXEC SOAP** <RETURN>

Each of the DOODLE files are loaded into memory as a whole and then processed by a machine code **DRIVER** called from BASIC.

In order to make room for the DOODLE files (the largest can be up to 12k) a certain amount of memory shuffling had to be employed.

The memory map alterations are as follows..

A00-AFF DOODLE DRIVER 1900-19FF LOADER prog 2000-4FFF DOODLE file 5000-5800 THEME file

The operation of the system is quite simple. Prior to the start of the tune the DOODLE DRIVER and DOODLE file are loaded. The system is initiated by a call of the form **CALL &A00, VAR%.** Where the value passed in VAR% is used to, **(a)** set the length of the DOODLE file, and **(b)** set the colour change flag. There after the DRIVER is called at **&A1D** at regular intervals. Every time it is called 64 bytes from the DOODLE file are passed to OSWRCH (Operating System Write Character Routine). The DRIVER then checks to see if it has reached the end of the file. If it hasn't then control is returned to BASIC. If it has reached the end of the DOODLE file then it firstly resets the pointers to the beginning of the file and then checks the colour change flag. If the flag contains a negative number then no colour colour change takes place and control is returned to BASIC. However if the flag is not negative then the values at &AFE and &AFF are placed in logical colours 1 and 3 respectively. Control is then passed to BASIC.

The alterations to the LOADER program are minimal though at first glance they seem quite extensive.

Line **70** *LOAD DRIVER puts the DOODLE DRIVER at &A00 Lines **80** and **90** have been added for the sake of demonstration only and can be removed when a 'one file' system is required. Lines **100-140** select the required pattern and place the file max values in D%. In a 'one file' program only one of these lines would be required. Line **150** note the new THEME load address and M% adjustment. Line **160** initiates the DOODLE DRIVER. Line **180-230** note the use of INKEY to give the double SPACE key effect. Line **200** this loop is included to ensure that the DOODLE continues if the first SPACE key press is held down. Line **210** this loop is included to ensure DOODLE continues until second SPACE press. Lines **270-280** note B%(1)-B%(3) are increased by &1000 Line **290** PROCD is called after all sound channels have been serviced Line **380** PROCD changes colours and calls DRIVER. Although colours are changed every time DRIVER is called a change will not be affected until the end of the DOODLE file is reached. The **B.PATTERN** file on disk is a BASIC program used as a source for DOODLE files. The **B. WAVE** file is the BASIC program used for RIBBON.

To create a new DYNAMIC DOODLE the following procedure should be adopted...

(a) The first step, *and probably the most difficult,* is to write a BASIC program to produce a pattern. (Look at B.PATTERN as an example)

Remember, DYNAMIC DOODLES has been created to work in *mode 5!* The pattern should normally use more than one colour and should move the cursor in such a way that the end position is also the start position, so that a smooth progression between patterns is obtained.

The use of PLOTs in the range **80-87** should be avoided ie Plot and fill a triangle, as these tend to take too long if used repeatedly.

Use **GCOL 3,n** wherever possible as this leads to the undrawing of any repeated patterns. Logical colours 1 and 3 are used by the DOODLE DRIVER and should be used inside any pattern for changing colours. The term 'pattern' should be employed loosely at this point. (After all look at the SOAP 'pattern' on disk.)

(b) Next, produce a SPOOL file from the BASIC program.

The SPOOL commands should be placed around the main body of the pattern generating commands in such a way that any mode changes, CLS commands etc are avoided. The **B.PATTERN** program shows in REM statements where the *SPOOL commands should go.

Use the ***INFO** command to check the final length of the SPOOLed file. If the file is longer than **&3000 (12k)** then the pattern will have to be shortened. However, if the pattern is too short, less than **&1800 (6k)** say, it will play too quickly and will quickly become 'boring', in this instance it is usually quite simple to cut the step rate of the BASIC program down so that it is drawn at half speed. The nearer to &3000 (12k) a file can be made the better.

(c) Extend SPOOL file to an exact number of pages in length.

Generally, SPOOL files will finish somewhere in the middle of a page of memory, this means that when they are loaded into memory that 'garbage' left in memory could be displayed as part of the DOODLE. (*The DOODLE DRIVER works to next whole page.*) To prevent this happening carry out the following...

(1) Make a note of the length of the DOODLE file using INFO. Increase this figure to the next whole page eg. if length = &2B37 then new length = &2C00

(2) *Switch off the computer!* This ensures that all memory is then cleared. Switch the computer back on again.

(3) Load SPOOL file into memory at &2000

e.g. ***LOAD DOODLE 2000** <RETURN>

(4) Save DOODLE file from &2000 to &2000+new length

e.g. ***SAVE DOODLE 2000+2C00** for the example above.

(d) Amend LOADER program so that line 100 reads...

e.g. **100 D%=&4C:*LOAD DOODLE** (assuming LOADER is now a 'one file' program).

It is extremely easy, for anyone with a BASIC program already written, to produce a DOODLE file from that program.

SECTOR ZERO

Colour printer winner, and two ingenious reader routines

The biggest prizes are in Disk User

No not the EEC butter mountain, or Big Daddy's Jockstrap. Yes it's time to announce the winner of a superb **Integrex Colourjet 132** printer in our great *Disk User Reader's survey* competition. Complete with colour print dump software, the Integrex is widely recognised as the standard in colour printing, and as such is supported by packages like *Illustrator Paintbox* from Nidd Valley, (see news section).

And the lucky winner of this £550 pound prize is: **Richard Brydges,** of Braintree, in Essex. Congratulations to him, the prize will be winging it's way to deepest Essex for his enjoyment.

Star Routine

This month we bring you two quite different routines. The first is literally a *star routine.* While the other is a very useful addition to the editing facilities on your micro's keyboard.

Star Scroller was written by N.J. Fairbanks of Liverpool. The program shows how text and pictures can be placed on screen while a 3D star field scrolls happily in the background. An excellent way to attract peoples attention to your latest program. The code for Star Scroller occupies two pages of memory and is self-contained. To run the demonstration select **Star Scroller** from the disk menu.

Speedy Keys

Extended Editing is the brain child of Tim Campen a relative newcomer to machine code programming. His routine once run will speed up the auto-repeat rate of the keyboard when **SHIFT** is pressed in conjunction with any key. This means that in View and Wordwise editing speed can be greatly improved.

For those with technical minds this is roughly how it works. The routine, which is machine code, is called every time a key is pressed. It checks to see if the key pressed is the **SHIFT** key if it is a ***FX12,2** command will be executed. This command sets the delay between each repeat of the key being read and printed. The routine once set up will stay in memory until the computer is switched off.

To run the routine simply select **Extended Editing** from the disk menu. To check that it has worked try pressing a letter key and while it is repeating press **SHIFT**. There should be a visible difference in speed. The real worth of this feature is seen when using it in conjunction with the cursor, copy and delete keys.

COMING SOON

TWO FOR THE PRICE OF ONE

In the November and December issues of Disk User you can look forward to TWICE as many programs for the USUAL price.
The November TWO FOR THE PRICE OF ONE disk features **educational software**, including a Computer Aided Learning development program. Inside the magazine we talk to the major educational software houses.

Adventures are the name of the game in December with full blown adventures to suit all tastes crammed onto our second **200K** issue.

Francis Botto

BEGINNER'S GUIDE TO DISKS

Beginning with the BBC? Don't be intimidated, it's just a box of RAMs and processors and buses and...we'll let Francis Botto explain

Hardware angles

The BBC Micro still boasts the sort of hardware which makes those critical – design – connoiseurs gesture with unbridled approval. You know the type of thing I mean, design intricacies that lead experts to stare into the circuit diagram as if it were some sort of crystal ball showing tomorrow's share prices.

Naturally, we can't all be design whiz kids – and it's a good job too – but the BBC Micro can be viewed as a little more than a 6502 processor running at 2MHz with "X" amount of memory and you don't have to be an engineer or an egg head to look beyond this. It's like most experts will tell you, computer hardware is as difficult as the level at which you wish to approach it, and for this discussion nothing more than a practical exploration is in order. Just think of your BBC Micro's hardware as an adventure – and by today's standards it's a pretty simple one.

Thanks for the memory

Discounting the almost compulsory upgrade of sideways RAM, the normal BBC Micro supports its notoriously meagre 32K – the memory size that launched that well-repeated and boring phrase, "...easy to fill!"

And just in case you didn't already know, RAM which stands for Random Access Memory is one of the great misnomers that's haunted this industry for years. The term was used to reflect the fact that it took the same amount of time to read any single piece of data. The same holds true for ROM or Read Only Memory, a quite different type of memory whose name actually infers its function – strangely enough! Obviously, RAM should have been called WARM – Write And Read Memory.

I propose to start a useless movement for bored computer people obsessed with lost and frivolous causes, to make up a petition to be handed to someone of no real importance in the computer industry, who can't do anything to change matters.

Setting aside memory capacity, the organisation of such RAM is also important, and varies considerably from one machine to another. Not forgetting the sort of RAM – as there are in fact two very distinct types. Additionally, the rate at which memory is accessed is also significant, and serves as a guide to the machine's overall performance.

The BBC Micro's memory is composed of sixteen 4816 dynamic RAM chips – unlike static RAM, dynamic RAM has to be constantly refreshed. Each chip has 128 two-bit memory locations – deriving the 32K is a little too involved for our discussion.

The RAM is quite fast being accessed four times every microsecond – 1/1000 of a second in other words – twice by the processor and twice by the 6845 CRTC (the Cathode Ray Tube Controller, which we'll talk about later). The CRTC also obliges in refreshing RAM.

The processor and friends

At the heart of every micro is a processor, which in a computer-sense determines it's very personality, particularly in terms of the machine's indigenous assembly language. The BBC Micro's 6502 processor like any other includes a number of registers, such as the program counter, the accumulator as well as an Arithmetic and Logic Unit – or ALU – which negotiates the all-important machine arithmetic and logic operations like AND and OR – just like they're initiated in BBC BASIC in fact.

If you're not sure as to what a register is, it's basically a tiny piece of memory capable of storing a small number of bits – and in this case the processor's registers are eight-bit. With the aid of an address bus, the program counter is able to step through memory, permitting the processor to access instructions or data in a step-by-step fashion. In this sense the program counter is very much a master of order. The 6502's program counter comprises two eight-bit registers, where sixteen-bit addresses are split into two bytes.

You may have realised, that the program counter and stack work pretty closely together when subroutines are being executed – the stack being the area of memory in which return addresses from subroutines are stored. Return addresses are "stacked" in a LIFO (Last In First Out) sequence, in which the last return address to be stored is the first to be retrieved. I suppose the stack of plates analogy best explains its operation – the last plate put on the stack will be the first to be taken off.

The addresses are naturally transmitted via the address bus, which consists of sixteen individual address lines labelled A0 to A15. All eight-bit processors like the 6502 incorporate sixteen lines – which are unidirectional permitting only the transmission of data from the processor.

Not long ago, the number of address lines was considered to hold the key with regard to the maximum amount of memory that could be incorporated. For instance when memory is connected directly to an eight-bit processor, it's found that 2△16 Kbytes – in other words, 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2, which is 64K – can be united. It's for this very reason that most eight-bit micros have a maximum of 64K memory.

The other bus is the eight-bit data bus, which is bidirectional premitting data to pass both to and from the processor. And if you're wondering, "Why then don't the data signals interfere?" This is where the clock comes in, for each cycle the processor like the whole BBC Micro does something precise. For instance, over one clock cycle, all the data lines could be programmed to output data, in the very next cycle, the data bus could be input lines.

With the processor running at 2 MHz, each cycle is a fleetingly short moment in time – for example, the data bus may output for 1/2,000,000 of a second, and then input for 1/2,000,000 of a second. So have a heart the next time you grumble about the speed of your BBC Micro, because it's working quite fast!

Video and things

Originally a monochrome device and designed to generate character information, the 6845 CRTC is pretty much stretched to its limit. The CRTC's specification is almost completely defied, with the device being used to provide both colour information as well as graphics. The colour information – after being shifted here and there – is achieved from four bits – or a nibble – of static RAM or palette RAM as it is chicly called. All colours, are encoded using only four bits. And just like address lines govern "a" maximum memory size, the greatest number of colours that can be achieved from a nibble is 2△4 – or 2 * 2 * 2 * 2 – which is 16. That's where the 16 colours in Mode 2 come from.

The 6845 CRTC works hand-in-hand with the video ULA - or Uncommitted Logic Array. The ULA is a custom chip in that it isn't a standard device which can be found in a number of machines, or bought as an off-the-shelf component. The BBC Micro's ULA, like any other, started life as a chip with a number of logic gates on it, the manufacturer – Ferranti in this case – then layed down the aluminium connections according to Acorn's requirements – "committing" the gates so to speak. Finally, the ULA became a video device. Incidently, the BBC Micro was one of the first machines to incorporate this sort of custom chip technology.

Among other vital operations, the ULA serialises data received from the CRTC. In this capacity it works as a parallel to serial convertor – in other words data applied a byte at a time is fed out one bit after another.

Looking at hardware is a bit like painting, you never know when to finish – if you *can* finish that is. And with most micros, to really get to grips with the inner-labyrinth a little more than a magazine article is required. Well it's a start!

NEXT MONTH

October Issue Out- September 16th

Comms crazy

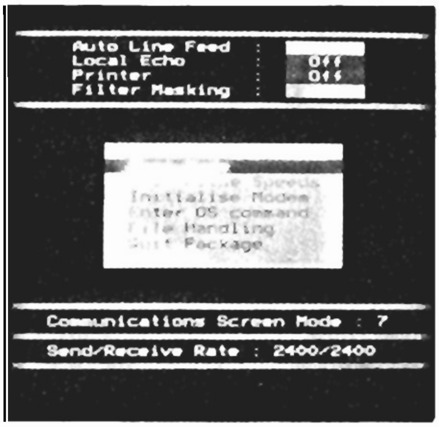
You don't have to be, but with British Telecom it helps. We provide an easy entry point with RACS, a pop-up menu driven comms program.

Graphic genius

Ever spent hours playing with one of those Spirograph things (what do you mean you're too young to remember). Well now we present **Spironet** a graphic drawing tool with AI interface, and command language.

Better Backup

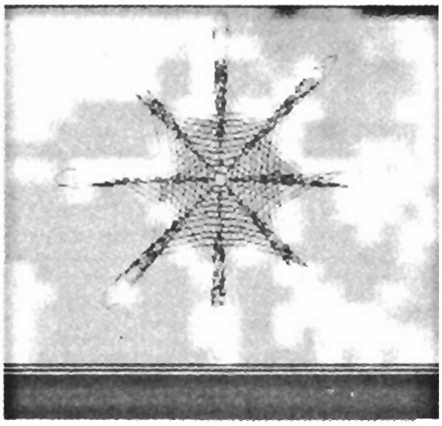
Superior by far to the Acorn Backup command this powerful menu driven program is the latest in a series of utilities from Disk User.



```
Auto Line Feed   :
Local Echo       :   Off
Printer          :   Off
Filter Masking   :

Communications Screen Mode : 7

Send/Receive Rate : 2400/2400
```



Gameplan

Peter Scott gives you the lowdown, and the highdown; everything in fact you need to become a successful games writer. Series starting now.

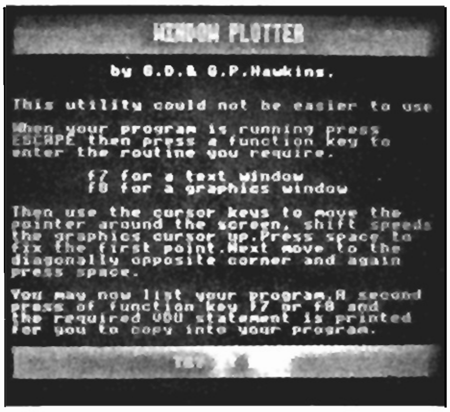
Plus

Logic Problems. the first of many. **Reader Routines** they keep coming in. **Beginners guide,** and much more.

G.D.Hawkins

WINDOWS

An ingenious utility for everyone who needs to create a screen window in BASIC



```
             WINDOW PLOTTER

       by G.D.& G.P.Hawkins.

This utility could not be easier to use

When your program is running press
ESCAPE then press a function key to
enter the routine you require.

        f7 for a text window
        f8 for a graphics window

Then use the cursor keys to move the
pointer around the screen, shift speeds
the graphics cursor up.Press space to
fix the first point.Next move to the
diagonally opposite corner and again
press space.

You may now list your program.A second
press of function key f7 or f8 and
the required VDU statement is printed
for you to copy into your program.
```

This program is designed to help those of you who are having co-ordination problems!

If you have ever needed a graphics or text window and have had trouble in finding the parameters this program will do it for you.

All *you* have to do is to start writing your program on the end of this one. Later you can delete this utility and renumber. Programmers should put the MODE statement at the beginning of their section of the program (line 600). The utility knows which screen Mode it is in and if graphics are possible.

Get your windows right first time!

When your program is running press <ESCAPE> then press a function key to enter the routine you require.

f7 for a text window f8 for a graphics window

Then use the cursor keys to move the pointer around the screen. The <SHIFT> key speeds up the graphics cursor. Press <SPACE> to fix the first point. Next move to the diagonally opposite corner and press <SPACE> again.

You may now list your program. A second press of function keys f7 or f8 and the required VDU statement is printed for you to copy into your program.

---

and <RETURN> must be pressed to return to the menu.

(c) Split File: Files are stored in two sections of the memory. The ikon size data is held from &A00-&AFF, and the ikon data is held higher in the memory. When a file is saved, all of page &A00-&AFF is placed just below the ikon data, and the whole lot is saved as one block. This means that the file is larger than it needs to be, and unused size bytes are included, since 256 size bytes are saved, even if less than 256 ikons have been defined.

Split file forms two files. If the current filename is £$, files S.£$ and I.£$ will be saved. S.£$ contains the correct number of size bytes ie if the file contained 16 ikons, S.£$ would be 16 bytes long. I.£$ contains the ikon data alone. Thus, the user is free to load the ikon size data anywhere in the memory. Note that the files created by Split File cannot be used in the editing program.

(d) Leave This Section: This moves to the second menu, which has three options : Return to main program, which re-enters the editing program, leave the ikon system, which returns to BASIC, and return to main menu, which returns to the main utilities menu.

(i) The first ikon in the file is ikon 0, so the number of ikons in the file, as shown on the help page is (last ikon)+1. X co-ordinates and Y co-ordinates both start at 0.

(ii) A blank filename is a filename containing no characters ie just press <RETURN> when the program asks for a filename to be entered. Filenames can consist of any characters except "*#:. and space. Also characters above 127, ie those obtained by pressing SHIFT/CTRL + f0-9 will not be accepted. Filenames are automatically placed in the correct directory on the disc.

(iii) If the program encounters an error eg disc full, this is reported and the user is given the option of continuing. If this is confirmed, the program re-starts; if not the program returns to BASIC.

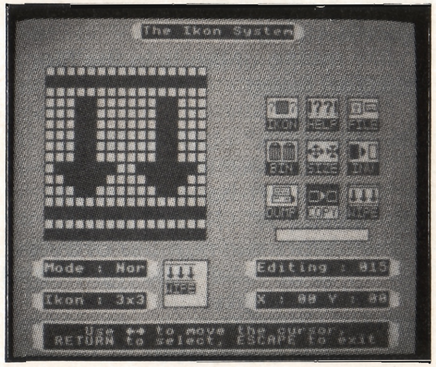
(iv) The program automatically unlocks all files on the disc, and also clears any function key definitions, since the catalogue data is held in the area of memory usually used for storing them.

(v) Always leave the program using the STOP or Leave the Ikon System menu options, since this resets the function keys and cursor keys etc. To re-enter the program, use CHAIN"IKON", do not type RUN.

(vi) The program may allow old file to be selected even if there is no file in memory. The program only checks the size data memory for valid bytes. If there happen to be some, the program assumes that they refer to a file, and will allow it to be edited. It is thus sometimes possible to edit a "file" containing whatever happens to be in the memory at the time.

Size bytes are valid if the byte=&xy where x and y lie between 1 and 4.

(vii) Although the program is fairly complex, instruction are shown at all stages, and the program traps any invalid inputs.



Use ↔ to move the cursor,
RETURN to select, ESCAPE to exit

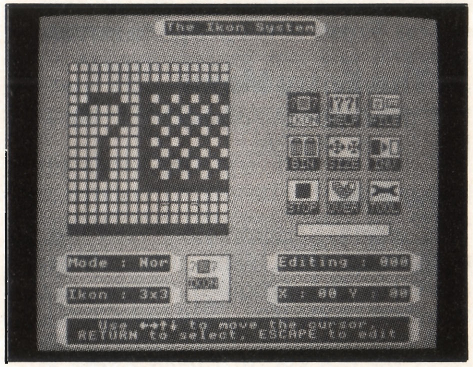
J.C.Bullock

IKON SYSTEM

A professional user interface is the most important part of any program. Use this powerful editor and ikon generator to add user-friendly control to any BBC Micro software

The Ikon System allows the user to design ikons, in effect very large user-defined characters. They may be designed in any size from one character by one character up to four characters by four characters. They can then be printed on the screen from any program using a simple machine code CALL command. The data for the ikons can also be printed as a hard copy and spooled as a text file which can be converted into a BASIC program using a *EXEC command.

The program has various facilities to allow ikons to be defined quickly and easily. It has been written to be very user-friendly, and hopefully totally "idiot-proof". It was written using, and is hence totally compatible with the 6502 second processor, and indeed is more powerful with it, allowing considerably more user memory space. The program as it stands does not use the extra shadow RAM on a Master or B+, though there is no reason why ikon files could not be loaded into the shadow RAM for use in other programs. N.B. The computer has to have a DFS fitted.



Use ↔↕ to move the cursor,
RETURN to select, ESCAPE to edit

On this month's Disk User all the system files are in the $. directory. D.IKONS is an example ikon file, P.OVERLAY is an example overlay file.

You can activate the program through the Disk User menu or by typing CHAIN "IKON". This will load the first program, which sets up the screen, and then loads the rest of the system from disk. A welcome message is printed in the window at the bottom of the screen, the information window. This is where all instructions for using the various program sections appear at the appropriate time, and most user input is carried out.

When the program has loaded, a three ikon mini-menu, the entry menu, is presented to the user. Instructions for using the menu are displayed in the information window. All mini-menus operate as follows.

Three ikons are displayed in neighbouring boxes, the left-hand ikon initially being highlighted by the cursor.

Use the left and right cursor keys to move the cursor to the correct option. Press <RETURN> to select.

A. The Entry Menu

The three options offered are : New File, Load, Old File.

New File

This starts a completely new file, containing one blank ikon. When selected, the program enters size selection mode (see "Size Selection"), and then asks for the name to be used for the file. If a valid name is entered, the program proceeds to the main menu. To return to the entry menu from New File, press <ESCAPE> during size selection, or enter a blank filename.

Load

This carries out the procedure for loading a file (see "Loading") and proceeds to the main menu. To return to the entry menu, press <ESCAPE> when in the load section.

Old File

This is only selectable when valid ikon size data is held in the memory ie a file has been previously created. This option is used after pressing BREAK or leaving the system. If no filename can be found in memory (it should be unless it has been corrupted) a new filename will be requested. As with New File, entering a blank filename returns to the entry menu.

The program now displays the first ikon of the file. In the case of a new file this will be blank. The top left-hand corner of the ikon is shown (16 squares x 16 squares), or all of a 1x1, 1x2, 2x1 or 2x2 ikon. The ikon is shown in real size in the box below the grid display.

B. The Main Menu

This contains nine options in a 3x3 grid. Use the cursor keys to move the cursor to the correct option, use <RETURN> to select. Pressing <ESCAPE> whilst in the main menu enters ikon edit mode. Whenever the menu is entered, the cursor is in the top left-hand square.

Edit Mode

This allows the current ikon to be edited. Note that the screen display shows several pieces of information. The main grid shows all of, or part of the ikon being edited. The bar below the menu grid shows the overlays (see "Define Overlays") that are available. The windows under the grid and menu show:

Editing Mode
Current Ikon Number
Ikon Size X x Y
Cursor X & Y position

Initially, the grid display shows the top left-hand corner of the ikon, with the cursor in the top left-hand square. Otherwise, when editing is resumed from the menu, the cursor will be where it was prior to entering the

ikons are shifted one down. ie ikon 16 becomes ikon 15, the current ikon. If ikon 29 (the last ikon) is deleted, ikon 28 becomes the current ikon.

C. The Utilities Section

A four option menu is presented giving the options Create Text File, Hard Copy, Split File, Leave this Section.

(a) Create Text File: If confirmed, a text file containing the data contained in the current file is spooled under the name T.£$ where £$ is the current filename. To turn this into a BASIC program enter:

***EXEC T.£$ (where £$ is the filename) END**

This will form a program consisting of several DATA statements in the form:

10 DATA nnss,xx,xx,xx ...

nnss,xx are in hex. *nn* is the ikon number, *ss* is the number of bytes contained in the ikon. *nnss* is followed by the appropriate number of data bytes to make up the ikon. The ikon data may take up more than one line; these are easily distinguished because they start with a single byte xx rather than the doyble byte nnss.

The data is in hex format, but must be read as a string and then converted to decimal as in the following algorithm:

```
FOR I%=0 TO n
        n=no of ikons in the file.
READ A$ Read
        nnss
N%=EVAL("&"+LEFT$(A$,2))
        Work out ikon number -
        the left-hand two digits.
S%=EVAL("&"+RIGHT$(A$,2))
        Work out number of bytes
        to be read - the right-hand
        two digits.
FOR L%=0 TO S%
        NB. Starts at 0
READ D$:D%=EVAL("&"+D$)
        Convert the hex to decimal.
        NB. read as a string.
Rest of Program...
        eg write the byte to
        memory
NEXT L%:NEXT I%
```

The character data is stored character row by character row. ie the characters, each needing eight data bytes are stored in the following order after nnss.

0, 1	
0, 1, 2	4, 5
0, 1, 2	8, 9, 10, 11
0, 1	6, 7
0, 1, 2, 3	12, 13, 14, 15
2, 3	2x2
3, 4, 5	3x3
3, 4, 5	3x2
2, 3	2x4
4, 5, 6, 7	4x4
6, 7, 8	etc etc

Thus, in the group of data refering to a 4x4 ikon, the first eight bytes would refer to the top left-hand character, the next eight bytes to the next character on the right and so on as shown above. It is thus easy to define user defined characters from ikon data, since the data is in user defined character format. The spooling of the text can be stopped midway by holding down <ESCAPE>. Note this may take a little time to respond.

(b) Hard Copy: This will print the ikon data on a printer. The data is output in hex format, and is printed in the form:

xx : &yy,&yy,&yy,&yy,&yy,&yy,&yy, &yy, &yy

where xx is the character number, and &yy corresponds to the various data bytes making up that character.

The printout can be stopped by holding down <ESCAPE>, though this may take a little time to respond. If the printer jams ie no more data can be sent out, the output stops, Pressing <ESCAPE> returns to the main menu. Pressing L allows a previously saved set of overlays to be loaded from disc. This is done in a similar way to loading an ikon file (see "Loading"), except that when a file is loaded, the overlays are displayed below the menu. If <C> is pressed at this stage, the disc is catalogued, and a new file may be loaded; if <ESCAPE> is pressed, the program returns to overlay editing.

Pressing <S> allows the current file of overlays to be saved in exactly the same way as an ikon file except that a filename must always be supplied – there is no current overlay filename.

Pressing <RETURN> allows the selected overlay to be edited. An overlay is edited in exactly the same way as an ordinary ikon (see "Edit Mode"). An overlay can be recalled in a different overlay, allowing variations of one pattern to be easily created eg to make overlay 1 the same as overlay 0, select overlay 1 and enter edit mode. Place the cursor in the top left-hand corner and press f7 then f0.

Overlays are retained even if BREAK is pressed or the utilities section is entered. They are also retained when a new ikon file is loaded. Any previously defined overlays are shown in the overlay window when the program is loaded.

(i) Utilities – TOOL Ikon This presents a three ikon mini-menu, giving the options Utilities (UTILS ikon), Copy ikon (COPY Ikon) and Delete ikon (WIPE Ikon). Copy and Delete can only be selected when there is more than one ikon in the file. Use <ESCAPE> to return to the main menu.

(1) Utilities: This checks whether the current filename is to be used. If this is confirmed, the utilities program is loaded. If not, a filename is requested. If a blank filenmame is entered, the program returns to the mini-menu, and the old filename is assumed, otherwise the utilities program is loaded, and this filename becomes the current filename.

(2) Copy Ikon: This allows an entire ikon pattern to be copied from one ikon to another to allow variations of one ikon. When selected, instructions are printed in the information window, and the user is prompted to press <RETURN>. The display will show Copy From : 000 and Copy To : xxx, where xxx is the current ikon. Copy From is highlighted by the cursor.

Use <SPACE> to move the cursor between the "to" and "from" values. Use up and down to change the highlighted value. Press <RETURN>, and the selected ikon will be displayed. If the value is changed but <RETURN> is not pressed, the old value is assumed, and this is shown when the cursor is moved to the other value. When the <COPY> key is pressed, the program shows the "from" ikon and asks for confirmation to copy "from" to "to". If this is given, the program performs the copy and "to" becomes the current ikon; if not, the mini-menu is returned to. Copy can be left by pressing <ESCAPE>.

The "from" ikon must be smaller than, or the same size as the "to" ikon. If it is not, the program will not perform a copy, and reports that the ikons are not compatible.

(3) Delete Ikon: If confirmed, the current ikon is removed from the file, and the next ikon in the file becomes the current ikon. If the deleted ikon was the last in the file, the previous ikon becomes the current ikon. eg if a file has 30 ikons, and ikon 15 is the current ikon, when delete is selected, ikon 15 is removed and the other

---

menu, providing the grid display has not been corrupted. The cursor is moved by pressing the cursor keys. The keys auto repeat so holding the key down keeps the cursor moving.

If the ikon is larger than two characters wide or two characters high, the grid display will scroll to reveal the parts of the ikon not shown, ie the grid display acts as a window onto the ikon. Simply move the cursor to the edge of the grid, and press the appropriate direction key. If the cursor is not at the ikon edge, the display will scroll to reveal more of the ikon.

The editor is controlled by the function keys as follows:

f0 : Normal Mode
Whenever edit mode is entered from the menu or f0 is pressed, normal mode is selected. This allows squares to be filled or cleared manually. To change the state of a square, move the cursor to the correct place and press <SPACE>.

f1 : Fill Mode
Pressing f1 selects fill mode. In this mode, each square that the cursor visits is automatically filled.

f2 : Delete Mode
Pressing f2 selects delete mode. In this mode, each square that the cursor visits is automatically cleared.

f3 : Inverse Mode
Pressing f3 selects inverse mode. In this mode, each square that the cursor visits has its state inverted, ie filled squares change to empty squares and vice-versa.

f4 : Line Operation
Pressing f4 causes the current editing mode to affect the whole of the cursor line ie if pressed whilst in fill mode, the whole of the cursor line would be filled etc. This operation has no effect in normal mode.

f5 : Store Line
Pressing f5 followed by fx where x lies between 0 and 8 stores the the current cursor line in store x. The previous contents of the store are automatically overwritten. The computer "beeps" upon completion of this operation.

f6 : Restore Line
Pressing f6 followed by fx where x lies between 0 and 8 fetches the line stored in store x, and places it on the cursor line. If no line has been placed in store x, this function has no effect. f6 followed by f9 has a special purpose. Whenever a line is retrieved from store, or a line operation is performed , the previous contents of the cursor line are placed in store 9.

Thus if a line is inadvertantly corrupted, it can be restored by pressing f6 then f9. However, note the following :

(i) Always restore the line immediately after it has been overwritten, since other functions overwrite store 9.

(ii) Only the line before the last operation is stored. Thus if a line operation is pressed twice, the original contents of the line will be lost.

(iii) Shifting a line/column (see below) will corrupt the line in store 9

f7 : Recall Overlay
Pressing f7 followed by fx where x lies between 0 and 7 places overlay x (See "Define Overlays") with its top left-hand dot at the cursor position. Note that overlays cannot be positioned anywhere that is less than eight dots from the right-hand side of the ikon. They can however be placed less than 8 dots from the base. If overlay x has not been defined, this function has no effect.

Pressing the <SHIFT> key with the left/right cursor keys shifts the cursor line left/right by one dot. Similarly, pressing <SHIFT> with the up/down cursor keys shifts the cursor column up/down. The state of the dot shifted in depends upon the editing mode; in fill or inverse mode it is filled, in normal or delete mode, it is blank. Column shift does not work with an ikon one character high, line shift does not work with an ikon one character wide.

Press <ESCAPE> to leave edit mode and return to the menu.

The Menu Options

(a) Select Ikon – IKON ikon This option allows a new ikon to be selected. The current ikon number is shown in the bottom right hand corner of the information window. Use the up and down cursor keys to change its value. If <RETURN> is pressed, this value is displayed in the Editing : xxx window, and the corresponding ikon is displayed. If <ESCAPE> is pressed at this stage, this ikon will become the current ikon ie the ikon upon which all editing functions etc will be carried out.

If the ikon value is changed to the value greater than the last ikon in the file, and <RETURN> is pressed, size selection mode (see "Size Selection") is entered, and if this is completed, a new ikon is added to the file. This becomes the current ikon. Pressing <ESCAPE> in size selection mode returns to ikon selection, and no new ikon is added to the file.

Note: when select ikon is left, the current ikon value is that which is shown in the *Editing : xxx window*, regardless of the value shown in the information window. <RETURN> must be pressed to select the current ikon.

Pressing the left arrow key changes the ikon value to 000. Pressing the right arrow key changes the ikon value to the last ikon in the file.

(b) Help – HELP ikon Selecting this option provides a page of information about the current file. Press <RETURN> to return to the main menu.

(c) File – FILE ikon This presents a three ikon mini-menu, giving the options load, save or new file. If an operation is carried out completely, the main menu is returned to; if not, this mini-menu is returned to. If a load operation results in invalid data being loaded, this mini-menu cannot be left. A new file must be started, or a valid file loaded.

(1) Loading : File selection mode (see below) is entered and if this is completed, the program tries to load the selected file. If this is successful, the program leaves the loading section.

There may be files shown in the catalogue that are not Ikon System files. These can be loaded, but should be rejected when the program checks the size data; if the size data contained in the file contains any invalid bytes, the program will report that the file is unsuitable. Pressing <RETURN> returns to the catalogue. However, any previous file in memory, will have been overwritten, and the program will not allow this file to be edited; a suitable file must be loaded or a new file started.

(2) Saving : The catalogue display is shown. Press <C> to recatalogue a disk, <ESCAPE> to leave the save section. If <RETURN> is pressed the program asks whether the data is to be saved to the current filename. If this is confirmed, the program saves the data, and returns to the main menu. If not, the program asks for a file name. If a blank filename is entered, the program leaves the save section, and the old filename is retained. If a valid name is entered, the data is saved under this name, and this new filename becomes the current filename.

(3) New File : This is operated in the same way as the New File option from the entry menu.

File Selection: Whenever a load/save operation is performed, the grid display changes to show the available files on the disk. In the case of ikon files, these are in the D. directory; overlay files are in the P. directory. If there are no files, "No file" is shown. If there are files, file selection mode is entered. A bar highlights the top filename in the left-hand column. Use the up and down cursor keys to move the bar to highlight the correct file. If there are two columns ie 17 or more files, the columns can be moved between either by (i) pressing down at the bottom of the left-hand column, which moves the cursor to the top of the right-hand column; pressing up at the top of the right-hand column has the opposite effect or (ii) pressing left/right as appropriate.

If <C> is pressed, the program catalogues the disk ie different discs can be tested. If <ESCAPE> is pressed the load section is left. If <RETURN> is pressed, the highlighted file is selected. If it is of unsuitable length, this is shown; pressing <RETURN> returns to the catalogue display.

(d) Clear Ikon – BIN Ikon If this option is confirmed, the current ikon is cleared. This option cannot be reversed ie the old ikon cannot be retrieved.

(e) New Size – SIZE Ikon This option allows the size of the current ikon to be changed. The program enters size selection mode, and if this is completed, the ikon is redisplayed in its new size. The size can be changed repeatedly, without affecting the original data held, allowing the size to be changed many times if necessary. eg a 4x4 ikon can be changed to 3x3, then back to 4x4 without losing the original data, providing a different ikon has not been edited between the size changes.

Size Selection: The program displays the maximum number of characters, *not* bytes, that the ikon can occupy. If this is confirmed, the left hand digit in the size display is highlighted. The <TAB> key moves the cursor between the left and right hand digits. Up and down change the value of the highlighted digit. <RETURN> enters the size shown and exits size selection unless the *x-size x y-size* is larger than the maximum allowed number of characters, in which case the program returns to size editing mode, and a valid size must be selected. When entered from New Size, the initial size shown is that of the current ikon. When entered from New File, the initial size shown is 2x2. <ESCAPE> exits from the section.

(f) Inverse – INV Ikon This option reverses the state of the dots in the ikon ie clears filled dots and fills cleared dots.

(g) Exit Program – STOP Ikon Leaves the program. To rerun, type "CHAIN"IKON", not RUN.

(h) Define Overlays – OVER Ikon Overlays are 1x1 patterns that can be placed anywhere in any ikon being edited. This allows a shape to be defined once and then used many times in different ikons. When the section is entered, a cursor highlights the left hand overlay in the overlay display below the main menu. The number of the highlighted overlay is shown at the right-hand end of this window.

Use left and right to select the overlay to be defined. The cursor will not move beyond the first undefined overlay ie if no overlays are defined, the cursor cannot be moved from position 0.

IKON SYSTEM

E. Using Ikons in Other Programs

Create two files using split file. Load the print routine (which writes the ikons on the screen) by typing:

```
*LOAD ICODE1
```

Now load the two data files. For example, suppose the size data was to be loaded to &A60, the ikon data to &4000. If the files were called I.£$ and S.£$, type:

```
*LOAD S.£$ 0A60 *LOAD I.£$ 4000
```

Now type:

```
!&900=&0A604000
```

ie the first two bytes (four digits) of the ! data correspond to the address of the size data (ie 0A60), the last two bytes to the address of the ikon data (ie 4000). These commands can, of course be put as lines in a program. The !&900 etc. must be correct for the print routine to work correctly. ~ Ikons can be printed in text mode (VDU4) at the text cursor, or in graphics mode (VDU5) at the graphics cursor. To print an ikon, four INTEGER variables ie followed by a % sign must be used. The command syntax is as follows:

CALL &90A,I%,X%,Y%,M%. Note that CALL &90A,1,100,100,1
Note that CALL &90A,1,100,100,1 ie using numbers directly will not work.

I% is the number of the ikon to be printed; X% is the X position; Y% is the Y position; M% is the print mode (0 – text or 1 – graphics).

To print ikon 1 in text mode at 10,10:

```
I%=0:X%=10:Y%=10:M%=0:
CALL &90A,I%,X%,Y%,M%
```

to print ikon 5 in graphics mode at 500,500:

```
I%=5:X%=500:Y%=500:M%=1:
CALL &90A,I%,X%,Y%,M%
```

General Notes

(i) the program will only print an ikon if there is a valid size byte ie. it should report an error if an attempt is made to print ikon 17 of a 10 ikon file.

(ii) ikons can be printed in any graphics screen mode. No check is made for valid X and Y positions ie. ikons can be printed off the screen.

(iii) the machine code is relocateable. It is placed at the memory location set by B% in line 10 of program "PRINT". The ! location is equal to B%, the call address is B%+10. eg. if the code was assembled at &2000 by setting B% to &2000, use:

```
!&2000=&xxxxyyyy
```

(as described above).

```
CALL &200A,I%,X%,Y%,M%
```

The machine code has to be assembled at the correct location ie. machine code assembled at &900 will not work if loaded to &2000; it must be assembled at and saved from &2000. When PRINT is run, it automatically saves the machine code to the file ICODE1.

(iv) since the print routine does not directly address the screen, it and the data can be loaded into the 6502 second processor, and will still function correctly.

TRANSFER

Taking your disk files one step further

Disk User programs can be so useful that you'll often want to transfer them to their own disks and use them separately, without title page or menu. To do this successfully you'll need to learn a little about BBC BASIC and the DFS (Disk Filing System). In Disk User we don't believe in referring you to the manuals to here's an explanation of how such a transfer can be achieved.

Let's take the *IKON SYSTEM* program as an example this month. The relevant file on Disk User is:

X.IKONSYS

Make sure you have a blank data disk ready to receive the file. Insert Disk User and type

```
*COPY 0 0 X.IKONSYS
```

and press the RETURN key. Follow the keypress prompts on the screen until the > prompt returns. Now

```
*COPY 0 0 EXPAND
```

and press the RETURN key. Follow the prompts.

Because X.IKONSYS is a compressed file special to Disk User the first thing to do with your new disk is to type

```
CHAIN "EXPAND"
```

and to choose the option for IKON SYSTEM. The program will now expand out the separate programs which make up the package (be patient!).

Give it a title

You can now type

```
*TITLE IKON SYSTEM
```

and press the RETURN key. Now type

```
*CAT
```

and press the RETURN key and the screen will display the title plus the !BOOT, ICODE, ICODE1, IKON, IKON1, IKONS, UTILS, D.IKONS, P.LOVERLAY lined up below along with EXPAND and X.IKONSYS – which you can now delete.

The !BOOT file works with option 3 so type:

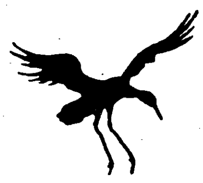
```
*OPT 4 3
```

and wait for the disk to stop whirring. You can now activate the IKON SYSTEM with the SHIFT/BREAK combination.

Details on transfer are also given on screen when you select IKON SYSTEM from the Disk User menu.

M Wilson

TRACER–BIRDS DATAFILE



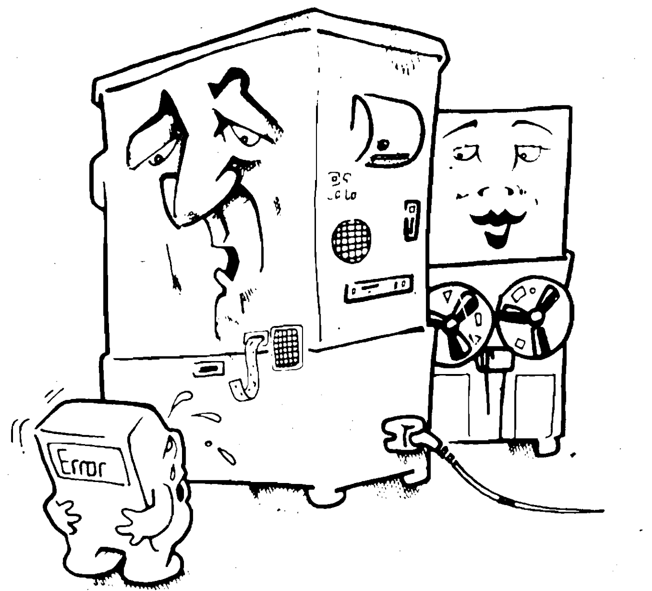
Another useful Tracer Datafile for all bird-lovers everywhere

Judging by the response from our readership many of you have found a use for our *Tracer* database program. And so, from reader M Wilson, for all ornithologists, twitchers, and naturalists everywhere we have a bonus this month. A short datafile, listing many of the more common British birds, and some of the not so common ones.

You will see that he has stuck to a relatively simple format, with just four fields on the screen giving the basic information for each bird. Whilst this might seem restrictive, or even arbitrary you should remember that Tracer has the unique ability to search for information over a number of files. This means that you could set up another file, giving different information about the bird, eg Latin name plus a short description, or set up a file recording your own observations.

Remember that a little time and thought setting up your datafiles will pay dividends later in ease of use and spped of recovery. Bear in mind the sort of application that Tracer is suitable for, ie one line records, and free text entry, not tables, and numeric data, and you will soon be able to build up a very powerful reference system with the minimum of fuss.

Readers with 40 track disk drives who may have been having trouble *creating* files in *Tracer* should refer to Sector Zero this month for a little help.


Error
Listen Son, You may be a 2nd Generation Computer but just remember that compared to a human you've only got the brain-power of a HADDOCK.—

SUPERFONT CLINIC

Superfont problems solved in our in depth look at readers requirements from this popular package

Dov Rosner in your issue No.7 has offered a splendid set of programs in Superfont. They come nearest to something I've been looking for, but will inevitably join the other font-builders that I've collected and do not use. This is because:
1. The printer driver insists on printing immediately. I need to save it for printing.
2. It cannot be used with a WW+ text with control codes in green. When I use OC commands (in white?) they are ignored
Would it be possible to have a font and printer driver which would add to their usefulness by letting me see the characters in Preview, and would allow me to prepare the text in the usual way for WW+.
Dr W.A Bennett Cambridge
Of the several points raised by Dr Bennett the first is perhaps the trickiest, and the most interesting. As I read his letter he wishes to use *Superfont* to **preview** his text, before actually printing out. However he may also wish to prepare a file which he can later print out without having to load in Superfont. In the second case the Superfont printer driver program (PRINTER) will have to be modified so that the output is redirected to a file. However if all that is wished is to see your file on screen, as it will be printed, then a simple ***FX5,0** will send all printer output to a sink, ie it will just disappear, but the characters will still appear on screen.

Superfont *can* be used with WW+ control codes in the text, however you must be careful, and follow a set procedure every time, as such things as line length are critical. What you do is:
1. Prepare your file in the normal way using whatever text formatting codes are necessary.
2. Then add the Superfont codes, which are all standard text, but preceeded by an @ symbol.
3. Using option 8 **SPOOL** the text onto disk.
Superfont can then be run in the normal way. It may not be possible to use some of the WW+ control codes in this way, but it is always worth remembering that the modern way of laying out text does not make use of lots of special effects. Instead text is spaced well apart, and the emphasis is on readability. If something different is required, then use Superfont, but sparingly, otherwise your text will resemble a typesetters nightmare.

Lastly to send commands to the printer; use the **OC** command, but put a @ symbol in front so that Superfont will recognise it and act accordingly.
*EXPAND and SUPFONT were copied without difficulty to a a new formatted disk. When EXPAND was chained on this second disk it appeared to run properly, but the expanded files are as shown on the attached *CAT printout*

```
*CAT
 (34) FM
Drive 0     Option 0 (off)
Dir. :0.$   Lib. :0.$

    DESIGN      EXPAND    L
    FONT        HIFONT
    HIPAGE      INTRO
    PRINTER     SUPFONT   L
    TEST        TEXT
```

These are not the files as described in the text on Page 21 & I am unable to run the program.
V J Cox Grantham
The names of the programs had to be changed, as originally there would have been copyright problems with an IBM compatible Desktop Publishing program called *Fontasy.* Unfortunately we did not quite get the programs on disk to agree with the text, hence the confusion. We can confirm that files on disk are correct.

PRINTER is the Superfont printer driver program (SPD) *DESIGN* is the Superfont editor *FONT* is an example font datafile *HIPAGE* is the assembly language source code for *HIFONT* *HIFONT* object code to raise *PAGE* on model B machines *INTRO* BASIC program prints page of info about Superfont *TEST* prints out Superfont characters *TEXT* example text file
I was very impressed by the professional nature of the May Disk User, and enjoyed System Wadgebury, but was disappointed to find I could not get the Superfont utility to work, for no apparent reason. However while reading the letters page I saw Mr. Gregory's problem with his lack of EQUB etc on his BASIC I machine, and investigated if the problem had arisen on my own BASIC I BBC. Indeed it had. I attempted to get round the problem, but to no avail. I have been looking for a decent printer driver for a long time after a disappointing effort from another BBC magazine, and I would appreciate it if you published the necessary changes, or to be more worthwhile provided a general purpose program to change ***any*** *EQUB ridden program to its BASIC I equivalent.*
Barnaby Willitts London

I think we have already sorted Mr Willitts problems with a new version of HIPAGE in issue nine, and corrections to Superfont itself published in issue ten. Also in issue nine we published help for BASIC I users. We hope that he, and other BASIC I user are now using Superfont to it's full potential.
Thanks for a very informative, and useful letter. We are sure that it will inspire Disk User readers to great things with Superfont. A Disk User T-Shirt is winging it's way to Eddie's home even as I write this.
I have recently purchased the May edition of Disk User. I have transferred the excellent Superfont program to another disk, and it is working perfectly – except that I am unable to call up the font within WW+. I have redesigned the font to print the Greek alphabet, and want

to use it to write documents in Greek. I would be most grateful if you could supply me with precise instructions on how to do this.

You don't say whether it is modern Greek, or Ancient Greek you wish to write in, but in general I would refer you to the above letters for information, in particular to Mr Climo, as he has overcome a very similar problem to yours. Of course where an alphabet is not based on the Arabic /Roman alphabet we are all used to then problems will arise, whether it be Greek, Cyrillic, or Chinese. this is due to the fact that there is no easy correlation between characters entered, and what is required. For example the Chinese *alphabet* contains over 10,000 characters. However there is a well established *shorthand* in use, with only a fraction of the characters required for understandability. Linguists, translators, (and typsetters) have encountered these problems many times before, and it is a good idea to find out how these problems have been solved before. Your local reference Library is a good source of information. We cannot supply ready made answers, as for every problem there will be a different solution.

How do I load your Printer driver into ***View*** *to use your special symbols. You have given a good example of how the effects can be used, but have said* ***nothing*** *on* ***how*** *to use them. Please show a little more sympathy and understanding to the less computer able, after all we all have to start from the bottom, or thereabouts.*

Superfont is **not** a View Printer Driver, but a general Font generator, and printer driver, and is therefore compatible with **all** wordprocessors. To use Superfont within View follows the same general procedure as for other word processors. Superfont commands are inserted within the text prefixed by an @ symbol. When the *PRINTER* program is chained it takes this text file and converts the Superfont *commands* into effects on the printer. However it was found that Superfont files, and View files were not fully compatible, so in issue ten a program, Adjust was provided to ensure full compatibility. For more detailed information I refer you to the above letters, and their answers.

One final point; we omitted to mention that to make Superfont work correctly then **Epson FX80** compatibility is required. This does not mean that if your printer is not an FX80 it will not work, but that some features supported by the FX80 may not be supported by your printer, or vice versa. We have noted that the problems people have had with printers is more often due to one of the specific printer commands, such as subscript, rather than a problem with Superfont. Please remember that in the long run Superfont is a general tool, and so is what **you** make of it.

Congratulations on the contents of Issue 7 – the Procedure Library, the Wadgebury game and even the short Background Print Utility were all very impressive!

But for me, top marks go to your Superfont suite (or is it 'Fontasy' – slight confusion in the article!). It performs very well: the font Designer program is simple but effective, and the printer-driver and Test font-dump are superb.

This brings me to the two main points of this letter:

(1) Firstly, I have designed 2 fonts to be used by Superfont – print-outs of them are attached. As you can see, one of them is a Celtic Uncial script, and the other is an emphasised script called Block.

(2) Secondly, I have some tips on how to use the "@CHmm, nn..." embedded command in Superfont, particularly when trying to access characters that are not on the keyboard. The problem seems to be that although you can, of course, access as many exotic characters as you care to re-define, remembering them can be a problem, and trying to read a document with the codes inserted is quite a strain.

My answer is to leave the document easily legible for as long as possible, by using some form of mnemonic for the exotic characters when creating and editing the document. Then just prior to printing it out, use the 'Search and Replace' facility on your Word Processor to substitute the appropriate embedded command + control code for each mnemonic. Indeed, a quick Basic or Wordwise+ program would be worth writing if you were intending to do this often enough.

Perhaps a quick example might make this clearer. In the French language, accents occur only on vowels, and are of 4 types: acute, grave, circumflex and diaresis.

I think you'll agree the first draft is much more readable than the second. Can you imagine what the second draft of a piece of Japanese would look like – a solid mass of embedded commands! With the above system, however, you could type it in using the standard system of Romanised transcription, and read and edit it quite easily.

I hope you find these ideas of interest.

Eddie Climo
Somerset

P.S. I had to use 'Search and Replace' to get the Printer-driver to print the '@' symbol in this letter.

```
<accented letter> = <letter><accent>

        <letter> = <vowel ! c>

        <accent> =
<acute!grave!circumflex!diaresis!cedilla>

                 = </ ! \ ! ^ ! .. ! 5 >


       'e acute' = e/  = @CH 132
       'e grave' = e\  = @CH 133
    'e circumflex' = e^  = @CH 134
     'e diaresis' = e.. = @CH 135
      'c cedilla' = c5  = @CH 150
               ............
               ... etc ...
               ............


First Draft of Document:

Re/ne/ est a une de/po^t, en e/tendant ses e/tudes dans une
cole\re nai..f.

Second Draft after Search & Replace:

R@CH 132 n@CH 132  est @CH 129 une d@CH 132 p@CH 142 t, en @CH 132
tendant ses @CH 132 tudes dans une col@CH 1133 re na@CH 139 f.

Final Hard-copy using Superfont Printer-driver:


René est à  une dépôt, en étendant ses études dans une colère naïf.
```

Step by step Superfont from Eddie Climo.

Paul Sparks

MARVELLOUS MANDELBROTS

See the beauty of the mandelbrot set in hours, not days with this efficient machine code program

In the past year or so, there has been a lot of fuss about the Mandelbrot set, and much on the subject has been published in scientific and computing magazines. It is undeniably a fascinating object, and what I present here is a program for serious investigation of it.

THE MANDELBROT SET

For those who have been in Outer Mongolia, the Mandelbrot set is a picture drawn on what is known as the *complex plane*. You all know about real numbers – 1, 1.4,- 4 etc. but there are other *imaginary* numbers which have no physical existence, like the square root of -1 (called **i**) for example. Imaginary numbers are often represented as a *real* number + a multiple of **i** for the imaginary part, eg (4 + 5i). These can be represented on a graph (the complex plane) with the y-axis giving the imaginary part and the x-axis giving the real.

If you take one of these complex numbers, then repeatedly square it and each time add on its original value, some complex numbers *diverge* . That is; the more you apply the function, the distance of the number plotted on the complex plane from **0** + **0i** (the origin) moves towards infinity and some *converge*. In other words, they never get further away from the origin than a certain value. A number is considered to have diverged when it is more than two units from the origin.

If you plot all converging numbers on the plane in black, you get an interesting shape called the Mandelbrot set. If you represent how fast diverging numbers diverge by plotting other colours, you get a beautiful colour picture. The Mandelbrot set only occurs in the region (-2 – 2i) to (2 + 2i). What makes it so fascinating is that there is not just one beautiful picture to be found, but an **infinite** number. If you magnify areas around the edge of the set and look more closely, you can see an infinite variety of other scintillating figures. *Editors note; if you think that it is impossible for such things to exist, just take a very close look at a leaf. You will find that the closer you look, the more irregular the edges will appear, but in a way that reflects the overall structure. Such similarities provide a link between the world of mathematics, and that of living things.*

PROGRAM CONCEPT

Back to the serious stuff. What you need to investigate the set, as a mathematician, or simply as an interested tourist, is a user friendly program to enable you to magnify and plot certain areas of the set, then save the pictures for future enjoyment and further magnification.

The problem is with programs published up to now is slow speed and lack of user friendliness. The slow speed is due to writing them in BASIC because implementing complex number arithmetic is difficult in machine code. However, it is possible and this program offers a much faster plot than the *leave it on all night* approach of conventional programs. However, machine code is not magic and it still takes up to several hours. Therefore, you can plot a special fifth size version to see what a picture will look like before committing yourself.

On running you are given a blank screen with 4 numbers at the top. The screen is like a window on a section of the complex plane. The first 2 numbers are the present co-ordinates of the bottom left corner of the screen (x first then y, ie *real* then *imaginary*). The second 2 are the co-ordinates of top right. To get started, type <H> which gives you a brief reminder of the functions, which are:

1. MODE. Select mode 1 or 2 for the next plot.

3. MAGNIFY. Used before each plot to select the area of the screen for magnification (ie the area to be plotted). This option has associated keys for selecting the area. Cursors : move magnification box. > , < : expand, shrink box. F : make cursor move faster. S : slow cursor to 1 pixel movement for fine adjustment. COPY : fix box and return to main program. The co-ordinates of the box are shown as you move it.

4. LIMIT. Allows you to set how many iterations the program will try before deciding that a number is converging. Default = 200, giving accurate pictures, but use 10 or 20 for quicker plots, 20 probably giving good enough accuracy.

2. PLOT. This allows you to plot the area of the screen within the magnification box. It gives a short menu: 0 large : full-screen picture 1 small : fifth size test picture (much quicker) 2 menu : return to main program During plotting, pressing Q allows you to abandon a plot.

5. SAVE. Allows you to save the present screen picture on tape or disk for later recall. If your finger slips and you don't really want the SAVE option, just type in a filename longer than seven characters to return to the main program. Reloaded pitures can continue to be magnified and worked on as normal.

6. LOAD. Allows you to load a pre-saved picture. If you don't want this option, return to the main program as in option five above. When it has loaded, type in the four numbers as prompted separated by commas, so the program can tell what part of the set it is.

Typing * enables you to access O.S. commands like *TAPE, *DISC and *CAT.

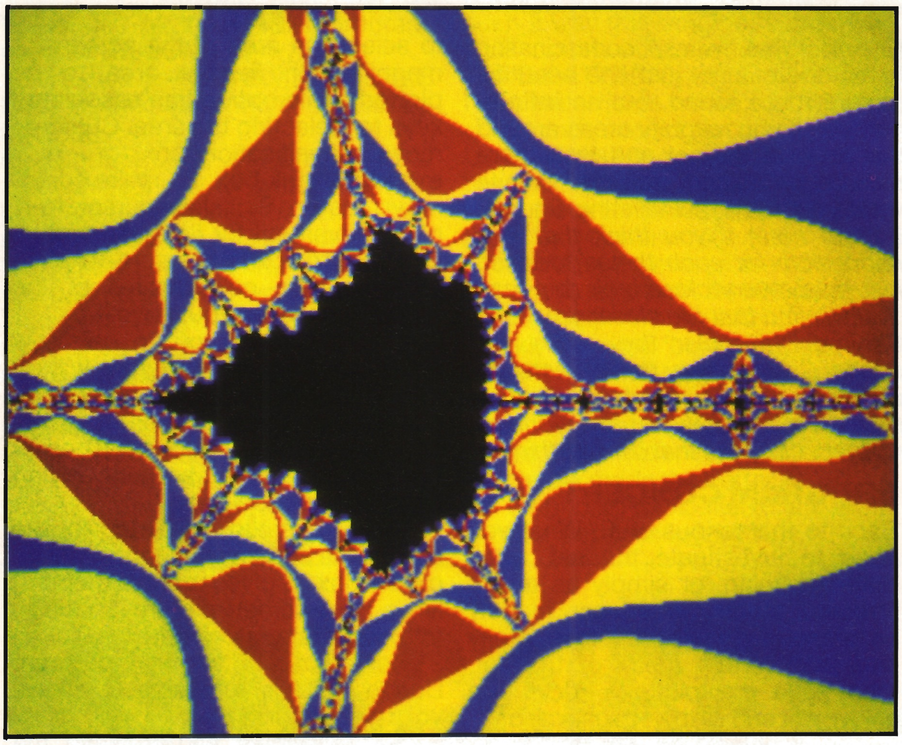
Your first action should be to plot a full Mandelbrot: -2,-2 to 2,2 so you can see what you're magnifying in future. Then you can try magnifying some areas, and explore yourself.

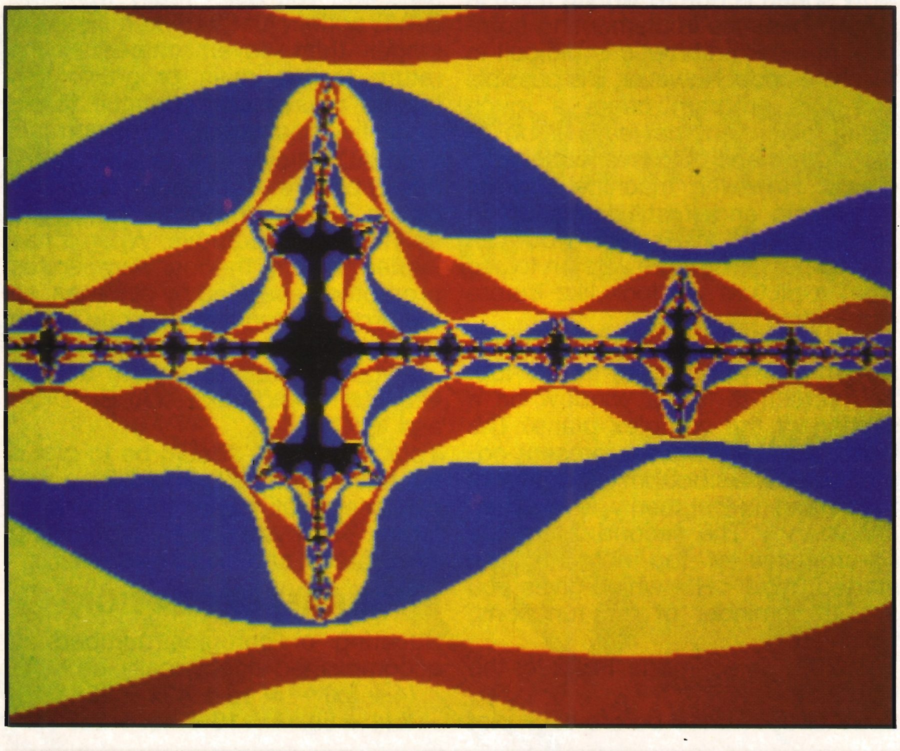
PROGRAM OPERATION

Squaring of complex numbers is accomplished algebraically, eg:

$(4 + 5x)^{\wedge 2} = 16 + 40x + 25x^{\wedge 2}$

& $(4 + 5i)^{\wedge 2} = 16 + 40i + 25i^{\wedge 2}$





But as $i^{\wedge 2} = -1$, this = 40i – 9
Thus, if you represent each number by 2 decimals, representing real and imaginary parts, calling the point being tested c and the current value of the function z, then you can say:

$$z = (zr) + i(zi)$$
$$c = (cr) + i(ci)$$

then the equation: $z = z^{\wedge 2} + c$
becomes:

$$zr = zr^{\wedge 2} - zi^{\wedge 2} + cr$$
$$zi = 2 * zr * zi + ci$$

The check for divergence becomes:

$$\mathrm{IF\ SQR}(zr^{\wedge 2} + zi^{\wedge 2}) > 2$$

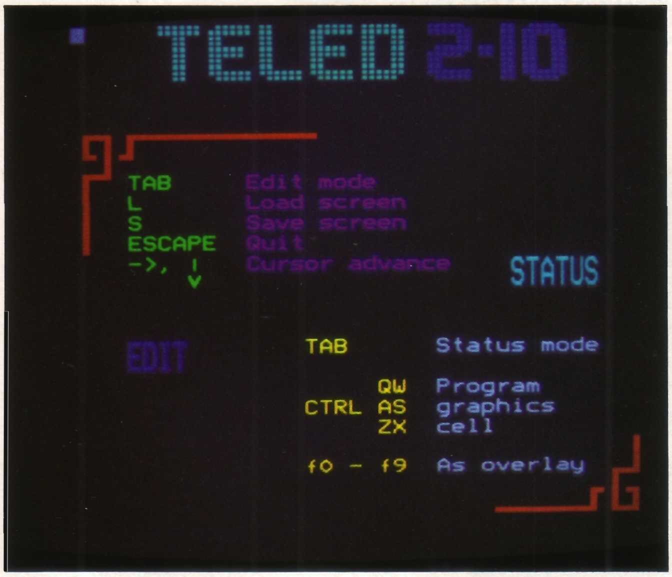
(Pythagoras), or: $zr^{\wedge 2} + zi^{\wedge 2} > 4$
This can be accomplished by simple arithmetic. The program uses 5 byte fixed-point, 2's complement numbers, the first 4 bytes giving the fractional part, the last the whole number part. The program uses an optimized machine code multiplication routine. Negative 2's complement numbers can be multiplied if they are turned into their positive counterparts first by negating routines.The sign of the answer is worked out by EORing the sign bits, then putting the number into 2's complement form if negative. The rest of the machine code is straightforward, mainly implementing the above functions. Notice how zr ^ 2 and zi ^ 2 are worked out only once, saving time. It must also keep track of the co-ordinates of the point being checked and the corresponding screen pixel position, continually incrementing them by a fixed amount. Points found to diverge before *limit* iterations have been performed are plotted in a colour depending on **co (no. of iterations) MOD mde (no. of colours)** and numbers which do not diverge before *limit* (ie converge) are plotted black. Other colour systems may be substituted to give better results.

Notice the use of EQUS FN to give macro-like functions in assembler. The machine code has been quite severely crammed in – PAGE is lowered to &1200 and some M/C resides in &B00 – &CFF: UDG and function key memory.
The photographs show some of the beautiful results, all computer generated and untouched by human hand.

Neil Craven

TEL-ED

To remark that teletext is awkward is to palter with the truth! *TELED* is a full screen editor which does its best to elevate teletext page creation from the impossible to the merely difficult



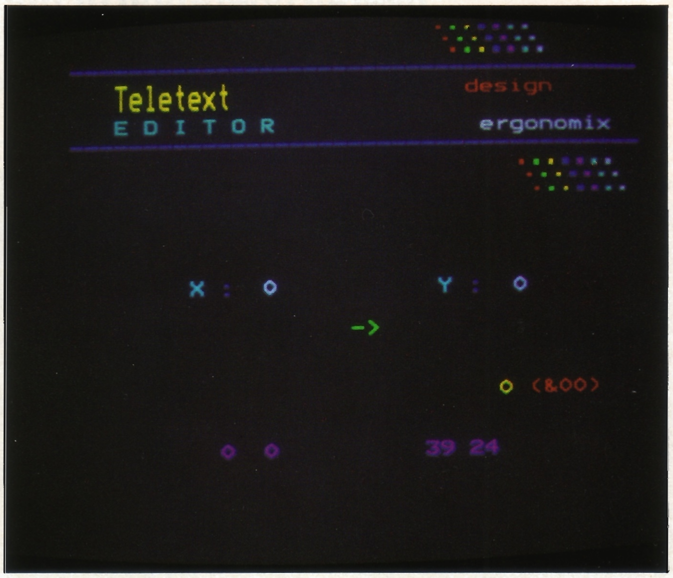
```
TELED 2·10

TAB        Edit mode
L          Load screen
S          Save screen
ESCAPE     Quit
->, |      Cursor advance
    v
                                 STATUS

EDIT         TAB        Status mode

                  QW    Program
             CTRL AS    graphics
                  ZX    cell

             f0 - f9    As overlay
```



```
Teletext                    design
E D I T O R                 ergonomix


      X : 0              Y : 0

                 ->

                           0 (&00)

      0 0              39 24
```

Unlike other screen Modes, the teletext based Mode 7 requires little memory yet supports eight colours and a number of unique effects. It is not by accident that impressively large programs including commercial adventure games choose Mode 7.

Lured by its promise to supply full technicolour on the head of a pin we should perhaps expect some devastating proviso in the small print; we are not to be disappointed. Teletext is not at all easy to use. Hence our utility this month.

Double screen

The TELED editor maintains two screens denoted: **status** and **edit**.

Movement between the two is accomplished by pressing the <TAB> key. Initially status mode is active and contains the following editing information:

Cursor coordinates
Cursor advance
Character at cursor
Markers #1 and #2 coordinates

The cursor coordinates reflect the position of the editing cursor. Cursor advance determines the direction in which the editing cursor moves following input.

The character at the editing cursor is displayed with its ASCII code in decimal and hex format. If it is a teletext control code its name, as it appears on the key overlay, is displayed.

The final pair of numbers are respectively the coordinates of marker #1 and marker #2.

Pressing L or S from status mode will produce LOAD and SAVE prompts at which point a screen filename should be supplied. Both L and S are idiot proofed. Errors occuring during load and save operations will result in a reprompt. A save request will result in a CONFIRM prompt if a file of that name exists – press Y to overwrite. Use <ESCAPE> to exit TELED.

Editing

In edit mode the function keys perform the action detailed on the key overlay. HOME moves the cursor to top left or top right depending on the setting of the cursor advance. MARKER #1 uses the current editing cursor coordinates to define the position of the top left corner of a window. MARKER #2 similarly uses the cursor position to define the bottom right of the window. CLEAR deletes the defined window.

COPY (CTRL f0) shifts the contents of the window to the current cursor position. The actual <COPY> key represents the teletext block character.

Graphics

Teletext supports primitive graphics in which lower case characters are displayed as a *graphics cell*. The dimensions of a cell are two wide by three tall. It is possible with the editor to program a cell directly without having to remember the corresponding lower case character.

By pressing CTRL and one of the letters Q,W,A,S,Z or X, corresponding to the position required to be toggled on or off within a cell, each pixel can be defined. The cursor does not advance during this operation.

A demonstration teletext page TEXTPIC is provided on disk and contains a further summary of commands.

Dave Stiles

STARTING OUT ASSEMBLER

The second part of this series looked at the overall operation of the search program. This final part looks in more detail at the operation of the opcodes used in it

This article is intended to help the novice in understanding assembler programs. To do this we will disect the search program code given in the previous article of the series. First, though, some definitions of terms used in the discussion.

I will use the abbreviation *hex* in place of the word *hexadecimal* and prefix hex values with an ampersand (&) as per BBC computer notation. A *mnemonic* is merely something to help one to remember; the mnemonic LDA is easier to remember than &A9, which is it's hex machine code equivalent, and is easier and quicker to write than LOAD ACCUMULATOR, which is what it means. LDA itself is an *opcode* (short for OPeration CODE), and requires an *operand* to tell it what to load. Not all opcodes require an operand, but if it is needed it can be either one or two bytes long, depending on the opcode itself.

Assembler mnemonics perform similar functions to BASIC keywords; they tell the microprocessor what to do and how to do it. Where the keyword is tokenised into a single byte in BASIC, the assembler mnemonic is converted to an opcode. The difference between them is that each assembler opcode can only perform a limited action, since it works directly on the microprocessor chip itself, whereas a BASIC keyword is interpreted to make use of several of these opcodes in order to perform a much more complex function. As the assembler opcode can be likened to the BASIC keyword so the assembler operand can be likened to the BASIC argument.

For reference a listing of the search program produced by the assembler (as opposed to the BASIC listing) is given. This was obtained by running the full program using 3 instead of 2 as the argument for the second call to PROC ASMBL (to give OPT 3). The left column of the listing gives the address (in hex) at which each line of code resides in memory. The next column, comprising between one and three subcolumns, gives the opcode residing at that address, followed by one or two bytes of operand as appropriate.

Load up

We will start at the beginning of the program listing with the code produced by the macro function SET–ADDR, which assembles to the code shown at address &0900. The purpose of this macro call is to put the 16-bit (two byte) value of address ADDR% into two bytes of memory at location STOR%, and comprises four lines of code at addresses &0900 to &0906 inclusive.

The first opcode encountered is LDA (LoaD Accumulator). The accumulator (sometimes known as register A) is the most important register in the microprocessor; it is the place where all the sums and data manipulation are carried out. Obviously one register alone cannot accomplish much, nor even with the other two registers X and Y to help it. In order for real work to be done the register has to be loaded with data, and after manipulation it often has to be stored into memory for later use. In the example, each of the LDA mnemonic lines is followed by the line STA (STore Accumulator).

The operand to the LDA opcode in the current example is preceded by the symbol # which is interpreted by the assembler to mean *immediate address mode* since the data to be loaded is to be found in memory immediately following the opcode byte. The operand is a constant value known at assembly time, and can never by varied at run time. In conjunction with opcode LDA it means that the operand byte immediately following the opcode should be loaded into the accumulator. The BASIC equivalent of the code at &0900 would be **LET A=&27.** The other two registers, X and Y, can also be loaded in this fashion with the mnemonics LDX, LDY. Note that STA, STX, STY can never have an immediate operand type. Think about it.

In the SET–ADDR example the value to be loaded has already been assigned to ADDR% for convenience, and is a 16-bit long (two byte) address ranging in value from zero to 65535 (&0 to &FFFF). Since the registers in the 6502 can only take a single byte value in the range of 0-255 (&0 to &FF), the value of ADDR% must be transferred in two stages; first the lower byte, then the higher byte (or the other way around; it matters not). The lower byte is obtained by loading A with the remainder of ADDR% after it has been divided by 256 (&100) (the action of BASIC function MOD), and the upper byte is obtained by ADDR% being divided (DIV) by 256.

Into store

The STA opcodes in this part of the

program show another type of memory addressing operand. This type of operand works directly on the contents of a memory location; the operand itself (STOR%) is the address of that location. The high byte of an address must be stored one location higher than the low byte in the 6502 memory in order for it to be used by certain addressing modes (discussed later). In some processors this order is the other way around, and is one reason for putting the routine into a macro.

To put things where they belong, an offset is added to the basic memory address of STOR%. In the first case this is zero, in the second it is one. Thus the low byte goes into location STOR%+0, whilst the high byte goes into the next location up. The +**0** is not really necessary; it merely clarifies the situation.

This mode of memory addressing is usually called *absolute* mode, and may be used to load or store any of the three registers A,X,Y. The complete code for this instruction normally takes three bytes; one opcode, two operand bytes (one for the high address byte, one for the low). However the code shown in this example is a slightly modified version which addresses page zero of memory (&0 to &FF) directly, and takes only one address byte. It is called *zero page* addressing, and if used correctly can save not only memory space but also program execution time.

This is one of the reasons why memory page zero is valuable, although there are even more compelling reasons for its use. Zero page addressing still addresses absolute memory, as in the longer form, but the upper byte is implied as being zero rather than being included in the operand. The program writer need not worry about specifying the correct opcode for this operation other than to specify the address; the assembler automatically deduces whether or not to use zero page or absolute addressing from the value of the address.

A glance at address &0928 will show the code for full length absolute addressing. On the way down to this address you will notice several examples of immediate and zero page addressing. Note also the cute way the assembler has of placing the macro function caller OPT before and after the code which it generates. The 'before' bit is what is returned from the function, whilst the 'after' bit is the actual function call!

Another address

The code at &0900 stores the address of the message strings block in zero page location &70 and &71. Taking a look at the sub-routine PRINT (at &0A20), which uses this stored address, will uncover yet another addressing mode. The PRINT routine is called after register Y is loaded with the offset value of the required message by the code at &0908; in this case Y is zero, since the message required is the first one in the message block.

PRINT starts by loading the accumulator with the first character of the message which it is to print. Unlike earlier LDA's this one does not have a #-symbol preceding the operand. Instead part of the operand is enclosed in parentheses.

If the notation used were **LDA msg** then the accumulator would be loaded with the byte contained in location MSG; a reverse operation to the STA operation already examined. If the notation were **LDA msg, Y** (without the parentheses) then the value contained in register Y is added to the address of MSG to form an *indexed* address. Registers X,Y are known as *index* registers for this reason, and either can be used in this method of addressing, both in absolute and zero page addressing modes.

This is very useful for fetching values from tables. An example of this can be seen at &094B, where Y contains a calculated table offset value which, when added to the base address of the table ELKTAB will allow the accumulator to be loaded with the required byte from the table. Thus if Y contains 2, then the accumulator will be loaded with the third byte (the first byte is at table +0) of the table, namely the value &CD.

Returning to PRINT, the notation **LDA (msg),Y** indicates *post-indexed indirect* addressing. The memory address at which the required data byte is to be found is itself contained in locations MSG (lower byte) and MSG+1 (upper byte); this address is then added to the contents of register Y to form the actual address. As an example consider the following suppositions.

The address of MSG is &0070.

&0070 contains &27. &0071 contains &0A. Register Y contains &0D (decimal 13).

The actual address of the data is &0A27.

&0A27+&0D=&0234 &0A34 contains &46.

So the accumulator is loaded with the contents of location &0A34; ie &46.

This makes it very quick and convenient for sending a string of data to the screen, since all that needs to be done after sending each character is to increment register Y, when the next character location becomes immediately available. Note, though, that if register Y were to be incremented past the value of 255 then it would start again from zero. Only register Y can be used in this form of addressing.

A further mode of addressing (not used in this program) is *pre-indexed indirect* addressing, and this may be used only with register X. It is used for obtaining data from one of several addresses, the address of each data byte being stored in an address table. The notation is LDA (ADDR,X), where ADDR is the address of the address table. As an example consider the following.

ADDR is &0080 &0084 contains &12 &0085 contains &08 &0812 contains &F3

If X contains 4 then the instruction **LDA (ADDR,X)** will load the accumulator with the contents of address &0812; ie &F3. This operation is doubtless useful on occassion, but I have not yet found a use for it.

Sub-routines

We came to PRINT by way of a sub-routine call. The mnemonic JSR (address &090A) is a sub-routine call similar in action to a BASIC sub-routine or procedure call. It switches the program to execute a piece of code which may be anywhere in memory, and then returns to the instruction immediately following the sub-routine call in the program.

The opcode is followed by a two byte operand which is the the address of the sub-routine to which it must go, &0A20; note the way in which the address is stored in memory, with the higher byte in the higher memory location. JSR has a companion called JMP (JuMP), which is equivalent to the BASIC GOTO. The operand format is the same for both, but there is no coming back from the jump, since it does not remember whence it came.

The flow of the program is controlled by the *program counter* (PC register). This contains the address of the next opcode to be executed. In action, the JSR opcode pushes the program counter (ie the address of the opcode which follows the JSR, in this case the address &090D of the LDA opcode) onto the stack, so that the processor can find its way home at the end of the sub-routine. In the case of the JMP, the program counter is not pushed to the stack. In both cases, however, the program counter is loaded with the address to which it must jump for its next ocode; it is, in fact, analogous to a LOAD PC instruction.

The *stack* is just a pile of memory locations in page one (&0100 to &01FF) of the 6502's memory which can be used for temporary storage by sub-routine calls and PUSH/PULL operations (more later). It is termed a *stack* because each item is added onto the top of the last one as if it were a stack of dishes being readied for washing, and then removed in the inverse order as they are washed; ie last one on is first one off. The odd thing about stacks, though, is that they are up side down. New items are not actually placed at a higher address, but at a lower one. Do not ask why.

Every time a byte is pushed onto the stack a *stack pointer* register is decremented to point to the next free location (remember it is growing up side down), and incremented back again when the byte is removed from the stack. In the case of the sub-routine call, the two address bytes of the PC which are pushed onto the stack by JSR will cause the stack pointer to go from, say, &01E6 to &01E4. These bytes are pulled off and returned to the PC by the RTS (ReTurn from Sub-routine) instruction at the end of the sub-routine, and the stack pointer returned to &0186.

Every sub-routine call must end with an RTS. If you try to JMP back from a sub-routine then the stack will gradually fill up. Not only that, but if you jump out of, say, the third of a group of nested sub-routines, then the second and first sub-routines will not be able to find their way home. The 6502 is rather more lenient than most microprocessors in that the stack pointer is only an 8-bit register; when it reaches the bottom (or top) of the stack, it just wraps around on itself, instead of pushing data into the program code space.

A sub-routine which is entered from an interrupt rather than from a JSR must be terminated with an RTI (ReTurn from Interrupt) opcode rather than with RTS. This is because the interrupt not only pushes the PC to the stack, but also pushes the status flag register. RTI replaces both of these registers. An interrupt does *not* save any other registers; you must save all registers before you use them in an interrupt routine.

Push and pull

The other use of the stack is to save the contents of the accumulator. This is done by way of the PHA and PLA (PusH Accummulator, PuLl Accumulator) instructions. PHA pushes the one byte accumulator onto the stack, whilst PLA recovers it (some processors call this POP rather than PULL). This is very useful in the middle of a calculation; for example when the register needs to be temporarily saved whilst another part of the sum is performed. Another pair, PHP/PLP, push and pull the status flags (see later).

The stack pointer may be read or changed only by copying it to and from register X with TSX and TXS instructions (Transfer Stack pointer to X register and vice versa). As a slight digression from the stack, there are two further pairs of transfer instructions; TAX/TXA and TAY/TYA. The first of these transfers the contents of the accumulator to register X (TAX) and back again (TXA), whilst the second pair do the same for register Y.

An example of TYA may be seen at address &09B5. Here is illustrated a drawback on the 6502; there is no PUSH instruction for the X and Y registers! I have defined macros in the source code which simulate PHY and PLY (PusH and PuLl register Y respectively), although they do need to be used with care since they destroy the contents of the accumulator.

In the case of the PHY code, the contents of register Y are transfered to the accumulator (TYA) and then pushed to the stack (PHA), whilst its opposite number PLY at &0A06 reverses the action (PLA, TAY). Should the need arise, similar pieces of code could be used for pushing and pulling register X.

As with the JSR opcode, anything pushed to the stack *MUST* be pulled off again. In fact more so. One of the major causes of new code failing is pushing more code to the stack (generally inside a loop) than is being pulled off it. Or vice versa. If a byte goes onto the stack, it and only it must come off again.

A piece of code which demonstrates this is at &09CD etc. This loop converts a two byte binary number into its decimal equivalent by repeatedly subtracting thousands, hundreds etc from it. The result of the subtraction of the first (low) binary byte is saved on the stack by the PHA code whilst the second (higher) byte is being dealt with (do not worry as yet over the SBC opcodes). After the BCC mnemonic (again do not panic) the result of the second calculation (the high byte) is stored and the first (lower) byte is recovered from the stack by the PLA opcode ready for storing.

Now the BCC (Branch if Carry flag is Clear) opcode is an instruction which can, under some circumstances, cause the program to jump out of the loop to the part of the code labelled NOSUM before the PLA can be carried out; very much like jumping out of a FOR/NEXT loop in BASIC. In order to balance the stack, a dummy PLA has been added at NOSUM to lose the extraneous byte from the stack (the accumulator is filled with the contents of register Y immediately afterwards).

BRANCH FLAGS

In order to react to various results produced from calculations or tests, the microprocessor must keep a set of status flags. *Status flags* are bits in the microprocessor's one byte status register which hold the results of various operations such as add, subtract and so forth, or in some way control the microprocessor. The various bits are given single letter mnemonics which represent the SET state of the bit. The flags are as follows.

Negative (N): set if the sign bit (bit 7) is set.

Zero (Z): set if the operation result is zero.

Carry (C): set if a maths operation resulted in an overflow of the accumulator.

Overflow (V): set depending on bit 6 and bit 7 overflows.

Break (B): set if a BRK interrupt occurs.

Decimal (D): when set maths operations are in binary coded decimal rather than in binary.

Interrupt disable (I): when set interupts are disabled.

It is possible to set and reset the carry, decimal and interrupt flags directly using SEC/CLC, SED/CLD and SEI/CLI respectively. It is also possible to clear the overflow flag (CLV).

In order to react to these flags there must be a decision instruction, analagous to the IF-THEN statement of BASIC. This is done using a *conditional branch* opcode. The branch is a kind of jump, except that instead of going to an absolute address (as the BASIC GOTO would go to an exact line number) it jumps by a *relative* amount from its current address.

It does this by adding the single byte operand of the branch opcode to the program counter. This byte is a signed twos complement one; if the top bit (bit 7) is set then the value is negative. Thus the program counter can end up with either a bigger or smaller value in it, and the program will branch to an instruction which is either further on or further back in the program, up to a limit of plus or minus 127 bytes.

Let us take the branch example already mentioned above. If the carry flag is set (remember this one is Branch if Carry Clear) then the next operation will be taken from

&09DC. However, if the carry flag is set the code at &09DA adds &0B to the current value of the program counter (&09DC). The program counter will now contain &09E7, and so the next operation to be performed will be taken from that address; ie at the label NOSUM.

Further back in the program is an example of a backwards branch. The BNE (Branch if Not Equal.. to zero) opcode at &09AB branches back to &099F (the label NXT) by adding the negative equivalent value &F2 to the current program counter value of &09AD. &F2 is the twos complement value -&0E (&100-&F2=&0E), thus &09AD-&0E=&099F.

The sums

The word *computer* means *one who calculates.* Therefore the computer must be able to do sums. The 6502 has very limited but adequate calculation abilities. The simple ones, INC and DEC (INCrement, DECrement) merely add one to or subtract one from a value held in memory. They have a pair of relatives, INX/DEX and INY/DEY, which do the same for the contents of index registers X,Y.

The 6502 also has the basic logical functions AND, OR and EOR (Exclusive OR), together with a non-destructive version of AND called BIT, which does a pretend AND which changes flags without changing the actual value.

AND is very useful for resetting bits in a flag byte. If a bit in the flag byte is set high and the bit AND'ed with it is also high, then the resulting bit in the accumulator is high; if either one of the originals is low then the result is also low. An example of this is given at &0940, where bit 5 of the accumulator is reset to force a possible lower case letter into an upper case one. Assume the letter is 'a' (binary code is 01100001) then this AND &DF (binary 11011111) results in the character 'A' (binary 01000001).

An example of the OR instruction is to be found at &09EE, where it is used to set bits in the accumulator instead of resetting them, which the AND would do. In this case a binary number of from zero to nine is OR'ed with &30 to form an ASCII number. If the accumulator held binary 4 (00000100) then the result would be &34 (00110100) or ASCII '4'.

EOR is an EXCLUSIVE OR. If either of the original bits is set, then the resulting bit is set. If both bits are set, however, the resulting bit is reset to zero. As an example consider the following. The accumulator holds binary 01100110. After the instruction **EOR #&A7** (binary 10100111) the accumulator would hold binary 11000001. This is a convenient way of toggling a flag bit without worrying which way up it is to start with.

ASL and LSR (Arithmetic Shift Left, Logical Shift Right) move all the bits in the accumulator or in a memory location one bit to the left or right respectively. In ASL, bit 7 is shifted into the carry flag and bit 0 is replaced with a zero, whilst in LSR the opposite occurs to bits 0 and 7. The effect of shifting left is to multiply the value by two; four shifted left becomes eight. A right shift has the effect of dividing by two.

Two similar instructions are ROL and ROR (ROtate Left, ROtate Right). The difference is that instead of being replaced by zero, the vacant bit of the shift is filled from the old carry flag; a nine place rotation would return the register and carry flag to their original value.

In addition

The only addition instruction on the 6502 is ADC (ADd with Carry). This adds the contents of the accumulator to the contents in the memory location addressed by the instruction's operand, putting the result into the accumulator. It will, however, also add the carry flag into the sum, so that if you do not want the carry flag taken into the calculation then it must first be cleared by the CLC instruction. Thus if the carry flag were set and the accumulator already held the value 6 then after **ADC #2** the accumulator would hold 9.

Any overflow of the accumulator is placed into the carry flag. On a multibyte sum the carry flag would first be cleared; successive additions automatically set or clear the carry flag as necessary and include it in the next byte addition.

An example of ADC can be seen at &0946, where the carry flag is cleared by default as a result of the compare (CMP; see later) above it; if the carry flag were set the code after the BCC would not be executed. This is fine in a final application, but if the code could be modified then watch out for carry errors; put a comment in the code to remind you to check for carry malfunction.

There are no real multibyte adds in the program, but there is one which cheats slightly. At &0A0F the lower address byte of a binary line number is added to the length of the line to get the address of the next line. Since the value to be added is never above 255 (one byte) there is no point in loading the second byte and doing a full ADC, so a test is made and the higher byte merely incremented instead.

Subtract and compare

SBC (SuBtract with Carry) is the obvious complement to ADC, but has a slight difference in the carry operation. Because the carry is now a borrow, the flag is first subtracted from one to form the complement before it is subtracted from the accumulator. The actual calculation carried out is **A=(A-M)-(1-C)** where A is the contents of the accumulator, M is the memory contents added to it, and C is the value of the carry flag.

An example of two-byte subtraction can be seen at label SUMLOOP2 (&09CD), where the lower byte, after calculation, is saved on the stack whilst the higher byte is subtracted. The decimal flag is cleared to binary mode at &09C9 before the loop is entered, and the carry flag is set at &09CA. If SUMLOOP2 is repeated it does not directly alter the carry flag again. The flag is, however, altered to the correct condition by the sum just before the BCC branch; if the branch is not taken then the flag must be correct.

ADC and SBC can both be used for normal binary calculation in the usual way, but can also be used to calculate in BCD (Binary Coded Decimal) by setting the Decimal flag in the status register with the SED opcode. In this mode the contents of the accumulator are adjusted after the calculation to convert the resulting binary value into two decimal values of 0-9 each (with carry if relevant). Thus if the accumulator contains &47 (representing the two decimal digits 4 and 7) then after adding five to this with the Decimal flag on the accumulator would contain &52 (representing decimal 52), not the normally expected binary value &4C.

The compare instruction occurs in several places in the program. It is actually a subtraction of the operand from the accumulator, similar to SBC, but where a subtraction would alter the contents of the accumulator the non-destructive compare only sets the processor status flags instead. Nor is the carry flag used in the calculation. The sum is a direct **A=A-M** where A is the contents of the accumulator and M is the contents of the operand. The decimal flag is irrelevant to the compare operation. Two other compares are available; CPX and CPY (ComPare X, ComPareY), used to compare the contents of the index registers X,Y.

So much for the description of the assembly code mnemonics. Whilst it obviously cannot compare with a book dedicated to the subject, I trust it has been of interest and benefit.

SERVICES

ORDERING INSTRUCTIONS

Order back issues and subscription from Infonet Ltd. Order software and binders from ASP Reader Services. Please make out separate cheques to speed the progress of your order.

BACK ISSUES

Issues two, three and five to nine are available in full colour magazine and floppy disk.

Disk User Two:

Ants!
Expert investigator
World Wars
Automatic Disk Catalogue Utilities
Mode 7 machine code
Software Manager
Guide to essential games on disk
Operating with OSWORD
Which RAM disk?

Disk User Five:

How your disk drive works
Program transfer
Arkenoid/Thrust cheats
Ravenskull screen maps
Magic Wall puzzle
Save Your Bacon
Disk User theme tune
Disk routine library
Overlay techniques
Spock's Logical disk menu
Toolkit character editor

Disk User Six:

Clip art
Psychebrot mandelbrots
Graphics from BASIC
Bobber arcade game
Martian Nim strategy
Pixel Perfect – zoom editor
Schizoscreenia – double screen
Collectors Items
Zoom Lens – enlarge or reduce screen areas
Hot Key – instant access to the OS

Disk User Seven

System Wadgebury arcade game
Background print utility
Procedure library manager
How to write a hit game (routines)
Superfont printer driver and character generator
Collector's Items
*CLOSE utility
*FORMAT for personalised disk formatting

Disk User Eight

Tracer database
Ecosoft superlife game
Extended disk editor
Autorun for BASIC programs
Hyperdriver NLQ font demo
Easypoke advanced cheating for games players!

Disk User Nine

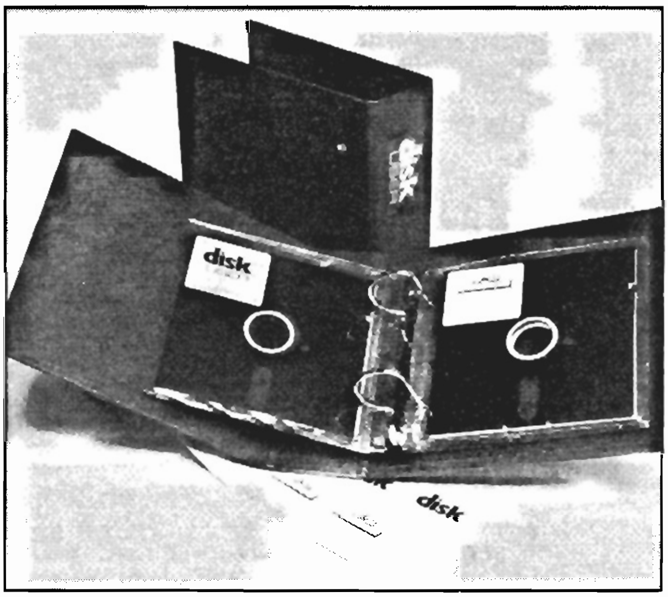
A.C.E. – ultimate character designer
ORCREST – arcade adventure
Flash Fonts – new screen fonts
One on One – advice for BASIC 1 owners
Starting out – programming in assembler
Kings & Queens – Tracer datafile
Welcome to A1?
Memory lister
Interrupt driven music
Collectors Items

DISK USER NO.3

Cholo cheat editor
Auto Save
Tazman
Dual Catalogue
Quick Break
Squash program
Count with Teddy
Quick Copy
Scientific calculator
Collectors Items
Order product number DU3

Back issues of Disk User cost £2.95 and £0.50 postage and packing. Disk User back issues and subscriptions are sent in special packaging to protect your software.

Send your remittance to:
INFONET LTD
5 River Park Estate
Berkhamsted
Herts HP4 1HL.
Tel: 04427 76661-4 for VISA and ACCESS credit card orders.



BINDERS

Organise and protect your disks with Disk User disk binders and data disks. Why not keep your Disk User program collection alongside your magazines in a stylish Disk User disk binder? The binder contains 10 disk sleeves to organise, and protect, your program disks.

And we've produced special Disk User data disks to match. The Disk User logo immediately identifies your Disk User program disks and there's room to title them and document the relevant magazine details.

Prices are as follows:

Disk User disk binder £4.95, which includes all postage and packaging
Order product number BDAU1
Disk User disk binder including 10 Disk User data disks £9.95, which includes all postage and packaging
Order product number BDAU2

PRODUCT CODE NUMBERS

PROGRAM TITLE	BBC B/B+	MSTR 128	COMPACT	ELECTRON
Global View	DB01	DB16	DB33	DE06
Graphics Pack 1	DB03	DB17	DB37	DE01
Musician	DB06			
Mode 7 Utilities	DB12			
Adventurescape III	DB20	DB28	DB35	
Combat Zone	DB21	DB30	DB32	
Easy Font	DB26	DB39	DB44	
ADFS Menu		DB27	DB31	
Graphics pack 2	DB29	DB34	DB38	DE29
Colour Ikon	DB36			
A&B Bibliography	DB40	DB53		
Bibliography Upgrade	DB55	DB54		
Statistics	DB41	DB42	DB43	
Home Office	DB46		DE46	
Graphics Constr. Set	DB49	DB50	DB51	
Master Only		DB56	DB57	
Getting Into Assembler	DB61	DB62	DB63	
Wordwise Plus	DB64	DB65	DB66	
Sideways RAM	DB67	DB68	DB69	
Pascal	DB70	DB71	DB72	
Adventure Special	DB73	DB74	DB75	
Graphics Special	DB76	DB77	DB78	
Skywatcher	DB79	DB80	DB81	
Online!	DB82			
Disk User One	DU1			
Disk User Three	DU3			

AVAILABILITY AND PRICE

ALL SOFTWARE AVAILABLE ON 80 TRACK (ADFS LARGE) FORMAT. ADFS OWNERS ENQUIRE FOR 40 TRACK AVAILABILITY.

Name	Product	Price
Global View	DB01	£10.00
ADFS	DB16	£10.00
Graphics Pack 1	DB03	£10.00
ADFS	DB17	£10.00
Musician	DB06	£8.00
Mode 7 Utilities	DB12	£8.00
Adventurescape III	DB20	£15.00
ADFS	DB28	£15.00
Combat Zone	DB21	£8.00
ADFS	DB30	£8.00
Easy Font	DB26	£10.00
ADFS	DB39	£10.00
ADFS Menu	DB27	£12.00
Graphics pack 2	DB29	£10.00
ADFS	DB34	£10.00
Colour Ikon	DB36	£8.00
A&B Bibliography	DB40	£15.00
ADFS	DB53	£15.00
Statistics	DB41	£10.00
ADFS	DB42	£10.00
Home Office	DB46	£10.00
Graphics Construction Set	DB49	£10.00
ADFS	DB50	£10.00
Bibliography Upgrade	DB55	£6.00
Master Only ADFS	DB56	£8.00
A&B '87	DB58	£8.00
ADFS	DB59	£8.00
Getting Into Assembler	DB61	£10.00
ADFS	DB62	£10.00
Getting Into Wordwise Plus	DB64	£10.00
ADFS	DB65	£10.00
Getting Into Sideways RAM	DB67	£10.00
ADFS	DB68	£10.00
Getting Into Pascal	DB70	£10.00
ADFS	DB71	£10.00
Adventure Special	DB73	£22.00
ADFS	DB74	£22.00
Graphics Special	DB76	£22.00
ADFS	DB77	£22.00
Skywatcher	DB79	£8.00
ADFS	DB80	£8.00
Online!	DB82	£10.00
Disk User One	DU1	£4.00
Disk User Three	DU3	£4.00

DISK USER NO.1

Available only as a software product, floppy disk and laserset manual. Programs are:

- Collectors Items
- Barmy Butterflies
- Cube of Zoth
- Auto Cataloguer
- Disk Examiner
- Karate Kid

Order product number DU1

DISK USER NO.4

SIDE 1:
- Introduction to ADFS
- Blockade
- Collectors Items
- Three dimensional graphics
- Random Access Tutorial
- Music Tutor
- Interceptor utilities
- Disk repair
- Repton screen printer
- New Repton 3 screens

SIDE 2:
- Full Micronet demo

SOFTWARE

9 Hall Road, Maylands Wood
Estate, Hemel Hempstead,
Herts HP2 7BH Tel: 0442 41221

CHECK WITH THE TABLES TO SEE IF OUR SOFTWARE IS AVAILABLE FOR YOUR COMPUTER. PLEASE INDICATE YOUR REQUIRED PRODUCT NAME, NUMBER AND TRACK FORMAT IN THE ORDER FORM

Software is only available on the formats recorded in the tables.

All prices include VAT and postage and packing. Please make cheques/postal orders payable to Argus Specialist Publications Ltd and print your name and address clearly on the back of your cheque. Telephone orders are taken for Visa and Access. All software is sent first class. Prices include postage and packing for European destinations. Please include £1.00 postage and packing (airmail) for other overseas order. Please allow 28 days for delivery.

ORDER FORM

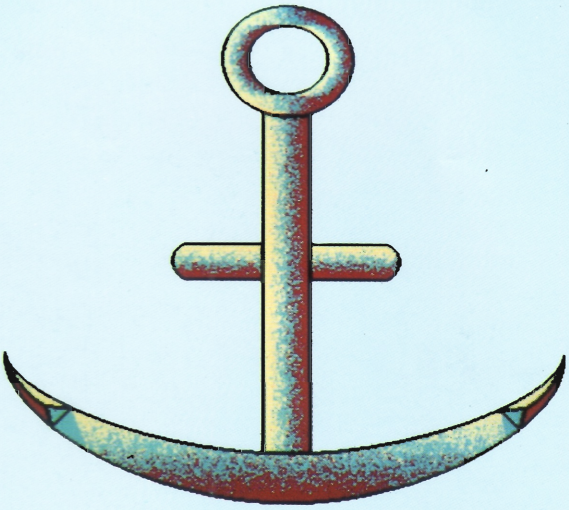
PRODUCT NAME	PRODUCT NUMBER	DISC Please specify 40 or 80 track	QUANTITY	PRICE
			TOTAL	

Name..
....
Address..
....
....
Postcode
Signed...
....
Date...
....

I enclose a cheque/postal order made payable to ASP for
£...............

Please complete the form in block capitals and send it with your remittance to Software Sale, Reader Services, 9 Hall Road, Maylands Wood Estate, Hemel Hempstead, Herts HP2 7BH. Tel: 0442 41221. You can direct technical enquiries about software to Phoenix Software on 0733 53355.

ANCHOR
it NEVER lets go



The complete memory resident programmer's assistant Calculator, Notepad, Monitor, ASCII Table and more Instantly available – just press <CAPS LOCK><SHIFT LOCK> together No more irritating disc swapping. Integrated programming environment. Anchor for BBC Master/B+128, or BBC B/Solidisk SWR. Powerful debugging/memory management facilities. Pop up/Pop down, over your favourite applications. This is it, the program you always promised yourself.

Have you ever wanted a reliable helper, someone to sit by your side night and day. Someone who never complains or goes to the pub for a drink, or asks for money?. Well here is the answer to your dreams. Now you can do all those things that were just too much bother, and still be better organised. Now when the telephone rings in the middle of your latest meisterwerk all it takes is a second to jot a note down, or do a quick calculation, and hey presto you can get back to work straight away.

Fully Documented

Anchor comes complete with a desktop-published, and laser printed manual that will help you get the best out of Anchor's more powerful features. Complete with worked tutorial examples for beginners, the manual is a valuable reference work in itself.

Availability

Anchor is available now in both Master/B+128, and BBC B with Solidisk SWR versions. Please follow instructions at the foot of this page for ordering.

Please send me a copy of Anchor (please tick box of choice).

☐ for the BBC Master 128/B+ 128 (DB92)

☐ for the BBC Model with Solidisk sideways RAM (DB93)

I would like Anchor on:
(please tick box of choce)

☐ 40 track

☐ 80 track

I enclose a cheque/postal order made payable to:
Argus Specialist Publications Ltd for £15.00

Name: ....

Address: ....

....

....

.... Postcode: ....

Total enclosed: £....................................

Send your order to Reader Services, 9 Hall Road, Maylands Wood Estate, Hemel Hempstead, Herts HP2 7BH. Visa and Access accepted. ☎ 0442 41221.

REPTON
THRU TIME
PREHISTORIC REPTON
EGYPTIAN REPTON
VICTORIAN REPTON
PRESENT DAY REPTON
FUTURE REPTON
500000BC
2000B.C
AD1850
AD1988
AD3000

REPTON — THE TIME TRAVELLER

Where did Repton come from? Does he have an Egyptian mummy? Is he the real Jack the Repper? Where is he now? Where is he going to?! We have been inundated with these and many other questions about our lovable hero. Now all is revealed in 40 new screens that vividly reveal Repton's evolution from prehistory to the future.

The PREHISTORIC screens with their caves, mountains and volcanoes, have Repton battling against pterodactyls and dinosaurs to collect edible berries. In EGYPTIAN times, Repton chases around pyramids and sphinxes collecting scrolls and meeting a mummy or two! Amidst the smog and grime of VICTORIAN times, Repton collects gold coins — but can he avoid the police and the gallows? PRESENT DAY Repton has even greater dangers to face as he rushes around the city's jungle of parking meters and skyscrapers, looking for cans of cola. If the gangster's machine gun doesn't get him, the traffic wardens will. Amazingly Repton may make it to the FUTURE. As he zooms through space collecting crystals, the Martians give chase. Will he succeed or disappear forever into the infinite depths of a black hole? Only you can help Repton Thru Time!

Each copy of Repton Thru Time includes:
●the Repton 3 main program ●the Repton 3 editor and ●the 40 new game screens.

BBC Micro Cassette..............£6.95	Acorn Electron Cassette..............£6.95
BBC Micro 5¼" Disc..............£7.95	BBC Master Compact 3½" Disc..£9.95

(Compatible with the BBC B, B+ and Master Series computers).

The screen pictures show the BBC Micro version of the game.

PRIZE COMPETITION

If you complete all 40 screens of Repton Thru Time without using passwords, you can enter our competition. A draw will be made from all the correct entries received to select 25 prize winners. Each will receive an EGYPTIAN REPTON Cuddly Toy and a signed certificate.

Closing date: 30th September, 1988.

REPTON ORIGINAL
Cuddly Toy

SPECIAL OFFER — NEW!

A unique range of Repton Cuddly Toys is now available. Each Cuddly Toy is about 24 inches high and has a realistic face and features corresponding to the particular Repton character. They are only available from Superior Software.

REPTON ORIGINAL	£9.95
ARCTIC REPTON	£9.95
TEENAGE REPTON	£9.95

Postage and packing free.

SUPERIOR SOFTWARE Limited

ACORNSOFT

Please make all cheques payable to "Superior Software Ltd".

Access VISA

24 HOUR TELEPHONE ANSWERING SERVICE FOR ORDERS

OUR GUARANTEE
- All mail orders are despatched within 24 hours by first-class post.
- Postage and packing is free.
- Faulty cassettes and discs will be replaced immediately.
(This does not affect your statutory rights).

Dept. RT3, Regent House, Skinner Lane, Leeds LS7 1AX. Telephone: 0532 459453.