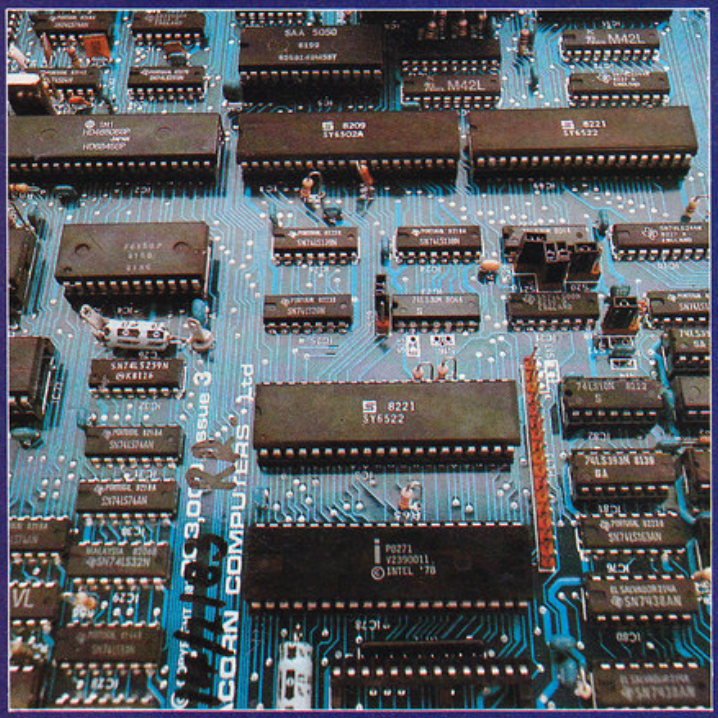
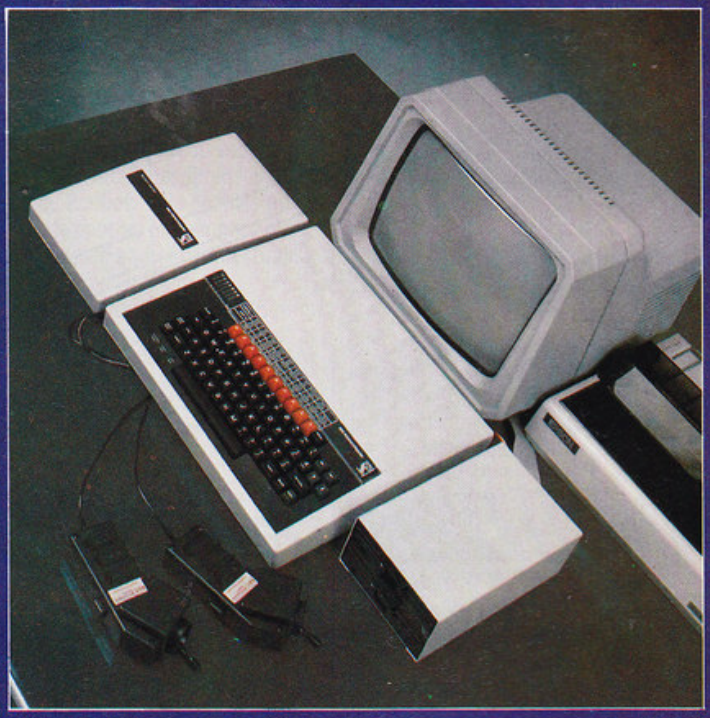
PCN micropaedia

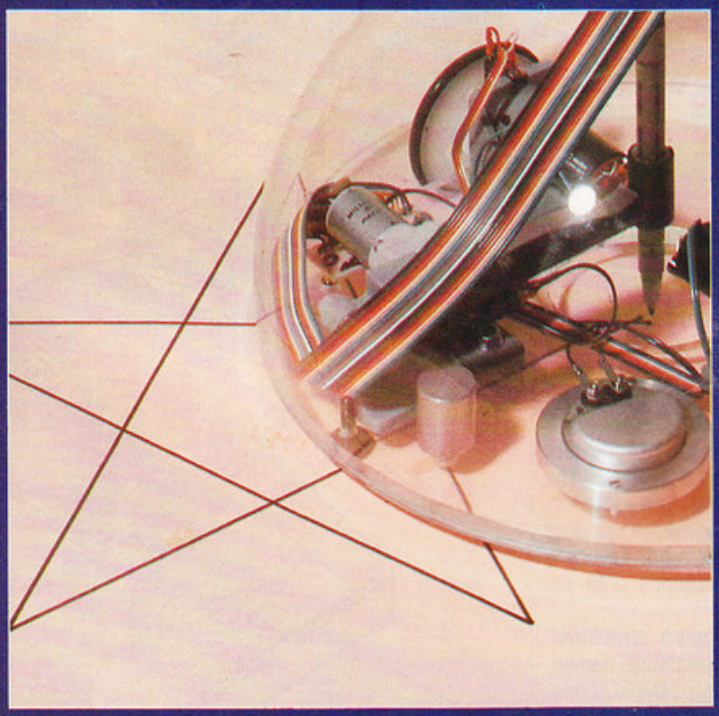
EIGHT-PAGE SUPPLEMENT PULL OUT AND KEEP

ANATOMY OF THE BBC MICRO

Start collecting your complete reference library of microcomputing.

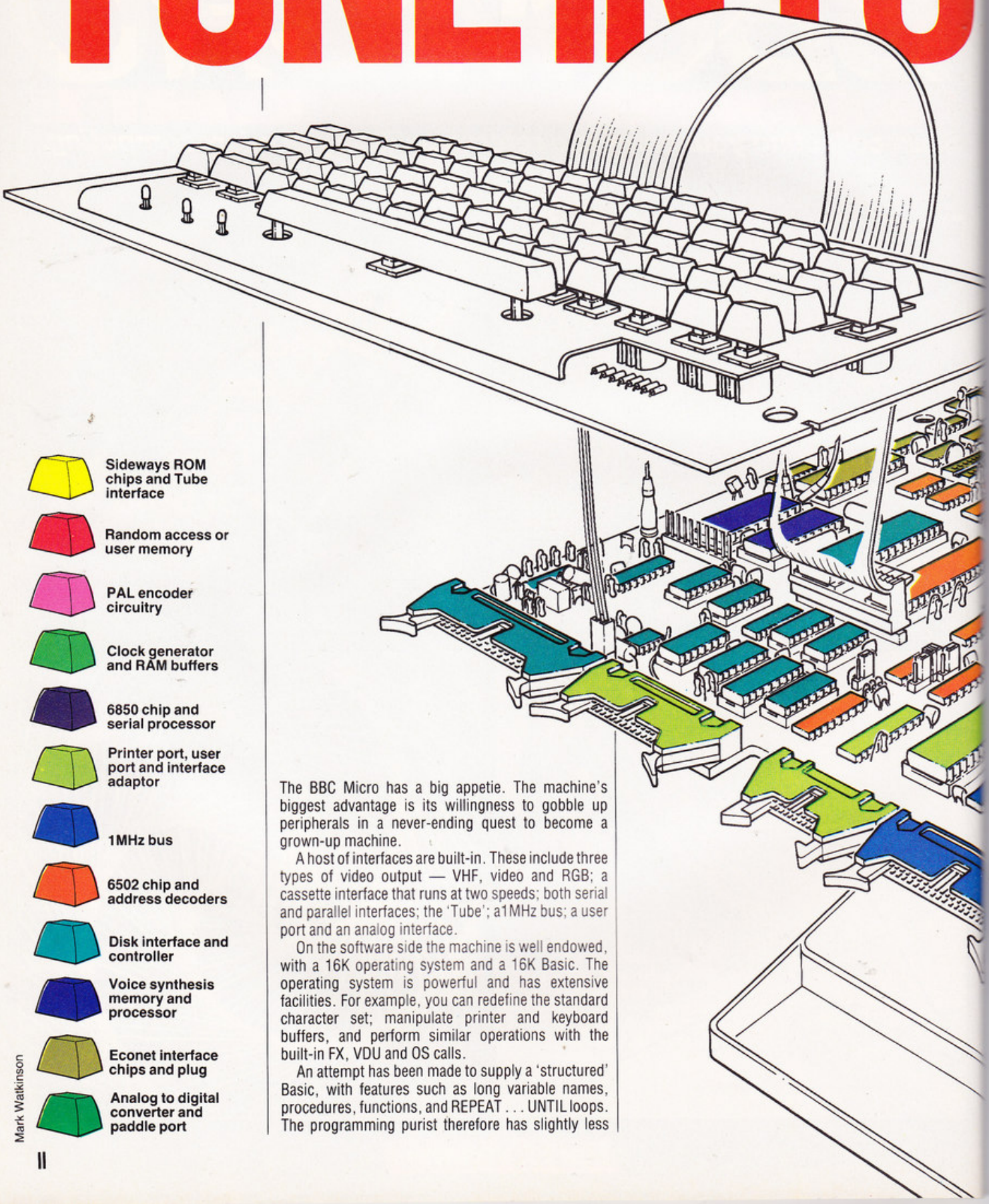






Part 1, Volume 1

TUNE INTO



- Sideways ROM chips and Tube interface
- Random access or user memory
- PAL encoder circuitry
- Clock generator and RAM buffers
- 6850 chip and serial processor
- Printer port, user port and interface adaptor
- 1MHz bus
- 6502 chip and address decoders
- Disk interface and controller
- Voice synthesis memory and processor
- Econet interface chips and plug
- Analog to digital converter and paddle port

Mark Watkinson

The BBC Micro has a big appetie. The machine's biggest advantage is its willingness to gobble up peripherals in a never-ending quest to become a grown-up machine.

A host of interfaces are built-in. These include three types of video output — VHF, video and RGB; a cassette interface that runs at two speeds; both serial and parallel interfaces; the 'Tube'; a1MHz bus; a user port and an analog interface.

On the software side the machine is well endowed, with a 16K operating system and a 16K Basic. The operating system is powerful and has extensive facilities. For example, you can redefine the standard character set; manipulate printer and keyboard buffers, and perform similar operations with the built-in FX, VDU and OS calls.

An attempt has been made to supply a 'structured' Basic, with features such as long variable names, procedures, functions, and REPEAT . . . UNTIL loops. The programming purist therefore has slightly less

THE BBC

justification for condemning Basic as used on the BBC machine. An assembler built into Basic also allows you to perform routines quickly without leaving Basic.

Educational software is also becoming available now that most local authorities are recommending the BBC Micro for schools and colleges.

Word processing is offered in a unique form on the BBC Micro. In addition to the traditional cassette and disk-based word processing, chip-based systems can be operated.

The sound generator can make four sounds at once, and each can have its 'envelope' altered. The sounds are 'interrupt driven', so that complex sequences of sounds can be generated without loss of execution speed.

The 'user keys' also prove their worth once you're familiar with the machine. These ten keys can be pre-programmed with commands, and can also be used to generate 30 extra characters.

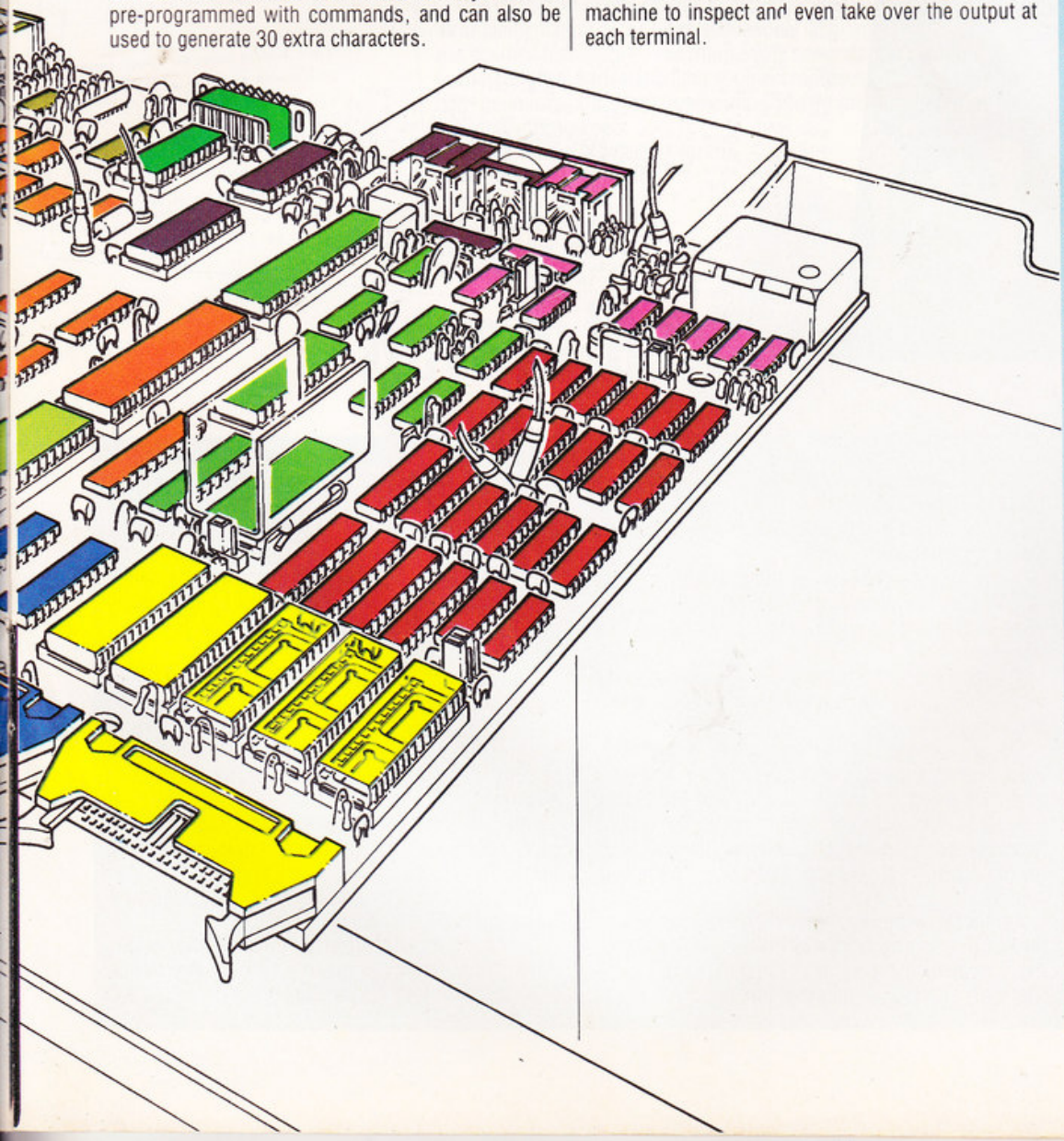
Eight graphic modes are available. The maximum resolution is 640 x 256, and a maximum of eight 'colours' can be used in some modes. Plotting co-ordinates remain the same, whatever the mode in use.

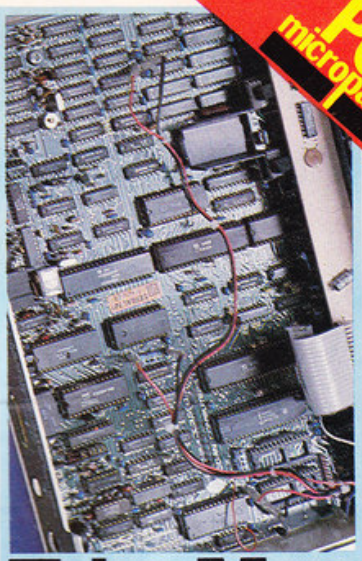
Although the BBC machine has great potential as a cassette machine it can also use disks.

The machine needs a disk interface fitted, which is possible with the 1.2 operating system. Once operating as a disk system the machine takes on a new role, with considerable potential as a business system.

The BBC machine can also be fitted with a networking system called Econet. This enables up to 256 machines to talk to one another on a single circuit, each terminal sharing the use of a single printer or disk system.

Econet is ideal for education as it allows one machine to inspect and even take over the output at each terminal.





Tube Map

The 'Tube' is a fast means of communication through which a Model B is able to be linked with a second microprocessor system.

In such a configuration the Model B acts as an input/output device for the second processor. It handles keyboard, disks, printer, screen, clock and so on, while the second processor does the rest — executing the central processing function of running the program proper, and interpreting the commands issued in whatever high-level language is in use (for example, Basic or Pascal).

The advantages of such a system are that much of the second processor's full addressable memory (64K or more) is available for your programs.

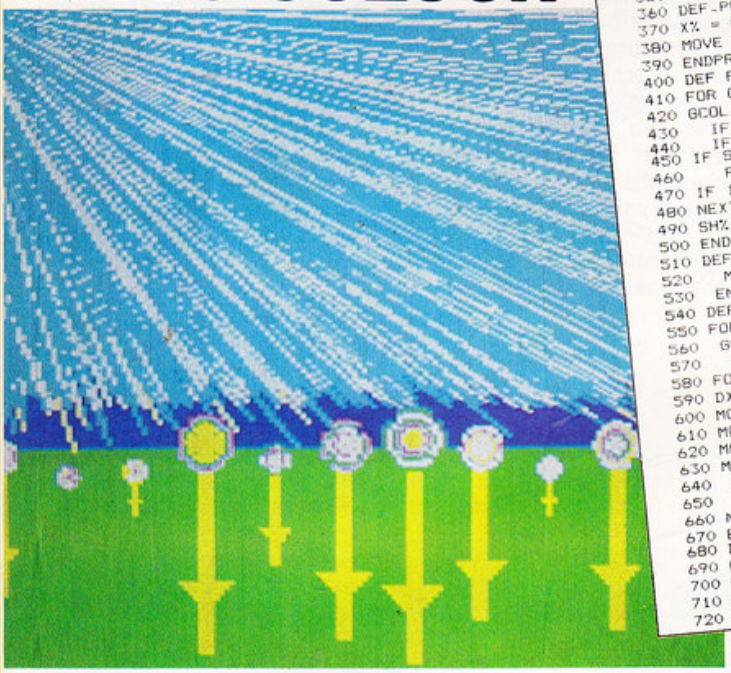
Considerable gains in speed result, partly because you have two processors carrying out different tasks at the same time, and partly because the second processors envisaged run at a higher speed than the Beeb itself.

A further advantage is that there will be a choice of second processor. There is a fast 6502A-based processor board, a Z80 unit which can run CP/M, and a 16-bit National Semiconductor 16032-based machine which can directly address up to 16 megabytes of memory. The 6502A unit will cost around £200.

To run in Basic or any other language, the second processor must have access to an interpreter. Interpreters are currently being written for the Z80 and the 16032. The 6502A second processor unit is able to initiate a copying-over routine when first switched on, which makes a copy in its own RAM of the model B's 16K Basic ROM.

Transfer is fast and takes only about one second, though at first sight it seems wasteful to have to keep two copies of the same 16K of code. But this is a small price to pay for a technique which makes the BBC machine so expandable. Comparable machines allow only add-ons.

BLOSSOMING INTO COLOUR



```
>LIST
  10    MODE 2: VDU 5: VDU 19,2,0;0;19,4,0;0;
  20 GCOL 0,4: MOVE 1279,1023: MOVE 0,1023:PLOT 85,1279,340: PLOT 85,0,340
  30 GCOL 0,2: MOVE 0,0: PLOT 85,1279,340: PLOT 85,1279,0
  40 FC% = 10: DIM PP(FC%,1), SZ(FC%), CO(FC%): SG = FALSE: RC% = 3
  50 PROCPLANT_POTS: RY% = 682: SH% = 30
  60 VDU 19,8,5;0;19,9,4;0;19,10,3;0;19,11,7;0;19,12,1;0;19,4,4;0;19,2,2;0;
  70 REPEAT
  80 PROCSUN_BURST: PROCSTALK
  90 UNTIL SG = TRUE
 100  RC% = 7
 110 FOR CT% = 5 TO 1 STEP -1
 120  L% = 9
 130    FOR W% = 0 TO 9
 140 L% = L% + CT%
 150 PROCSUN_BURST: PROCPETALS
 160 NEXT W%
 170 FOR C% = 0 TO FC%: CO(C%) = 7 + RND(5): NEXT C%
 180 NEXT CT%
 190 REPEAT
 200 ET% = 200
 210 REPEAT: UNTIL TIME >= ET%
 220 PROCSUN_BURST: VDU 19,7 + RND(5),RND(7);0;
 230 UNTIL FALSE
 240 DEF PROCPLANT_POTS
 250 FOR C% = 0 TO FC%
 260 PP(C%,0) = 100 + C% * 109: PP(C%,1) = RND(299)
 270   SZ(C%) = 3 - PP(C%,1)/100: CO(C%) = 7 + RND(5)
 280 NEXT C%
 290 ENDPROC
 300 DEF PROCSUN_BURST
 310  FOR C% = 0 TO 4
 320 GCOL 0,RC%: PROCRAY
 330 GCOL 0,6: PROCRAY: PROCRAY
 340 NEXT C%
 350 ENDPROC
 360 DEF PROCRAY
 370 X% = RND(1279): Y% = RND(RY%)
 380 MOVE 0,1023: DRAW X%,Y% + 1023 - RY%
 390 ENDPROC
 400 DEF PROCSTALK
 410 FOR C% = 0 TO FC%
 420 GCOL 0,3: X% = PP(C%,0): PROCLINE
 430   IF SZ(C%) > 1 THEN X% = PP(C%,0) - 4: PROCLINE: X% = PP(C%,0) + 4: PROCL
 440   IF SZ(C%) > 2 THEN X% = PP(C%,0) - 8: PROCLINE: X% = PP(C%,0) + 8: PROCL
 450 IF SH% = 20 THEN PROCLEAF
 460     PP(C%,1) = PP(C%,1) + 4 * SZ(C%)
 470 IF 1023 - PP(C%,1) < RY% THEN RY% = 1023 - PP(C%,1)
 480 NEXT C%
 490 SH% = SH% - 1: IF SH% = 0 THEN SG = TRUE
 500 ENDPROC
 510 DEF PROCLINE
 520   MOVE X%,PP(C%,1): DRAW X%,PP(C%,1)  + 4 * SZ(C%)
 530  ENDPROC
 540 DEF PROCPETALS
 550 FOR C% = 0 TO FC%
 560  GCOL 0,CO(C%): X% = PP(C%,0): Y% = PP(C%,1)
 570    R = L% / (4 - SZ(C%))
 580 FOR TH = 0 TO PI/5 STEP 4/R
 590 DX% = R * SIN(TH): DY% = R * COS(TH)
 600 MOVE X% + DX%,Y% +DY%: DRAW X% - DX%,Y% - DY%
 610 MOVE X% + DX%,Y% - DY%: DRAW X% - DX%, Y% + DY%
 620 MOVE X% + DY%,Y% + DX%: DRAW X% - DY%, Y% - DX%
 630 MOVE X% + DY%,Y% - DX%: DRAW X% - DY%,Y% + DX%
 640  NEXT TH
 650  IF 1023 - Y% - R < RY% THEN RY% = 1023 - Y% - R
 660 NEXT C%
 670 ENDPROC
 680 DEF PROCLEAF
 690 GCOL 0,3: MOVE PP(C%,0) + 16 * SZ(C%),PP(C%,1)
 700 MOVE PP(C%,0),PP(C%,1): PLOT 85,PP(C%,0),PP(C%,1) - 16 * SZ(C%)
 710 PLOT 85,PP(C%,0) - 16 * SZ(C%),PP(C%,1)
 720 ENDPROC
```

SECOND OPINION

I started with the book.

My first impression of the BBC Model B microcomputer User Guide was that it was well-written and clearly presented. But I quickly discovered it was best suited for reference purposes.

It contains descriptions of the capabilities of the machine, its operating system and language, but has none of the kind of spoon-feeding you expect in a training manual.

The references to sound and graphics suggested wide-ranging capabilities which were borne out when I began programming.

Sound was particularly impressive, with a four-voice synthesiser which allows definition of a plethora of effects using the simple SOUND and ENVELOPE functions. The graphics facility gave me various formats (according to 'mode' selected) depending on my requirements of pixel size, text size and number of colours.

Another attraction for me was that characters within the ASCII set can be defined by the user. In BBC Basic, the implementation of 'PROCEDURE' and 'REPEAT...UNTIL' made programming much easier and allowed better structuring of programs.

When I ran a series of test programs, the speed of operation was impressively fast. Even when I added modules to programs simply by tacking them onto the end, operating speeds did not seem to fall. After I'd written my program and added lines to it, the dynamic RENUMBER function was very rapid in re-aligning the documentation, while the LISTO function facilitated well-laid-out listings.

I spotted a slight drawback to the system in BBC's screen display . The machine used only one screen display area, while the Apple II, for example, can give up to two text screens and two high-resolution graphic screens (admittedly at a cost to available user memory).

However, the BBC compensates for this by combining user-selectable text and graphic windows with mixed-screen use.

With the speed of the machine, animation and sound can easily be added.

Overall, I felt a lot of effort had gone into the design, making even this small micro a very capable device with good facilities for expansion. **NC**

COUNTING ON THE BEEB

Four short programs to demonstrate in a rudimentary way the mathematical capabilities of the BBC Model B. One calculates fuel consumption on any car, in either imperial or metric fuel measurement (gallons or litres) and by the mile or kilometre. Another converts Centigrade temperatures to Fahrenheit, the third will add VAT to any price you are quoted, and the final program will give you percentages. They are short but are easy to expand and develop to handle more complex tasks. They also demonstrate the power of the 'INPUT' request which asks for more information to complete the program.

```
10 PRINT "INPUT THE DISTANCE TRAVELLED"
20 PRINT "FOLLOWED BY A COMMA AND THE PETROL USED"
30 INPUT M,P
35 PRINT "HERE IS YOUR CONSUMPTION"
40 PRINT M/P


 3 REM   A PROGRAM TO CONVERT CENTIGRADE
 5 PRINT "INPUT THE CENTIGRADE VALUE"
10 INPUT C
15 PRINT "HERE IS THE FAHRENHEIT"
20 PRINT 9/5*C + 32


 5 PRINT "INPUT THE PRICE WITHOUT V.A.T."
10 INPUT V
15 PRINT "HERE IS THE PRICE WITH V.A.T."
20 PRINT 1.15*V


 3 REM "PERCENTAGES PROGRAM "
 5 PRINT"INPUT THE PERCENTAGE YOU WANT"
10 PRINT "THEN A COMMA AND THEN THE"
20 PRINT "THE NUMBER YOU WANT THE PERCENTAGE OF"
30 INPUT P,N
40 PRINT "HERE IS THE ANSWER"
50 PRINT N*P/100
```

If the micro is the biggest thing to hit science labs since the test tube, then the BBC machine is the Pyrex of computers.

The BBC micro's 80 per cent penetration rate into schools is probably the best testament to that view.

In the schools, where most scientific and maths software is developed, high resolution graphics, animation and colour have until recently been missing from programs.

The marketplace demands that this BBC 'software gap' be filled — with the emphasis on quantity, not quality. This will happen as existing maths and science programs are translated to run on BBC machines.

Some of the best new software is being produced by the ILEA (Inner London Education Authority) SMILE centre (Secondary Mathematics Individualised Learning Experiment). SMILE programs are not written to prescribed formulas, or to fit existing mathematical structures. They are based on the ideas of teachers who have worked with children on micros.

A program called Snooker helps students judge angles and understand the geometric principles behind them, while in the Lines program talk between the two players is encouraged as they work together to defeat the computer.

One important maths capability of the BBC micro is its implementation of the Logo educational language. It is claimed that children can more easily understand spatial concepts and relationships, object representation, geometry, arithmetic, computing and problem-solving by using Logo.

Less software is available for the scientific uses of the BBC than for the mathematical, and much of it is of highly specific use in the classroom.

Dave Holland of the University of London Institute of Education offers a BBC-based Mass of Magnesium program that allows real-time interaction.

Another ideal scientific use for the BBC machine centres on the user port (a device made possible by the micro's 6502 base and 6522 VIA chip). This port makes it possible, for example, for a BBC Model A to use a joystick.

The user port can also be used for inter-machine communications and control of machinery — which would allow two BBCs to play against one another.

In addition to the user port, the BBC Model B has a digital to analog converter which makes short work of data collection in school science experiments. Unlike single-channel digital to analog converters available for other computers, the BBC multiplexes four inputs into one converter.

The conversion speed makes it too slow for speech conversion and recognition, or for recording a sine wave more than 1Hz, but the sampling speed matches the resolution of the screen. This useful compromise allows the machine to become a digital oscilloscope and record any experiment that inputs analog information at a slow enough speed.

The BBC's EVAL command is also a valuable feature in science work. You can type an equation into the computer while a program is running and allow the computer to evaluate the equation. This makes it comparatively easy to write programs that allow you to input an equation and have the computer plot a graph.

Graphics or text can be mixed on the BBC, and this makes the machine useful in labelling graphs. Text can be moved around easily, so Mg 0 need not appear as Mg20. Finally, the dual processor capability of the BBC makes a parallel computation possible.

Keys to the Store

The BBC Micro's 10 red function keys may look bland, but once you get used to them they are among the machine's most versatile features.

One marathon session of loading cassette programs by endlessly typing in CHAIN"" and/or LIST, LOAD and RUN should convince you.

Take the CHAIN"" function. By simply typing ⋆KEY 1 CHAIN"" and hitting Return, function key 1 will automatically respond with CHAIN"" when pressed. Number that instruction 10 ⋆KEY 1 CHAIN"" and you have the beginnings of a program to load all your most common instructions onto the function keys.

If you then go on to load all the function keys with useful commands (don't forget function key 0) and save the program, you will have an easily accessed group of key utilities that can be downloaded every time you turn on your machine.

You can then go on to make up a chart that will tell you what you have programmed each key to do. This can be slipped under the clear plastic ruler above the function keys. If you have more than one program using the function keys, you can devise interchangeable charts — although if you use the VIEW word-processing system, your chart space will be occupied by the VIEW command ruler.

Once loaded, the functions will stay on the specified keys until you do a 'hard' reset (hit CONTROL and BREAK keys together), turn the machine off or reprogram the keys.
This means that you can load and exit as many programs as you wish without losing the special functions.

Learning by Logo

The BBC is multi-lingual and will converse in several computer languages. Among the most interesting is Logo, a language developed to help children learn to talk to computers.

The Turtle is to Logo what the hand is to drawing. It can be either a cybernetic robot such as the Edinburgh Floor Turtle, consisting of a clear plastic dome on wheels with a retractable pen at the centre of its base, or it can be a triangular image on a VDU screen that creates a pattern by tracing a line as it moves.

There are many BBC Logo dialects, but even the most primitive can accommodate the same simple commands as the most powerful.

For instance, an equilateral triangle can be drawn in any Logo in this way, programming in 'direct mode':

```
FORWARD 100
RIGHT 120
FORWARD 100
RIGHT 120
FORWARD 100
```

Units for forward and backward movement are arbitrary, but angle movement left or right is in degrees. Usually, Logos allow single or two-letter entry for commands — FO or FD, LE, RI, etc.

Most Logos also incorporate a REPEAT command so the triangle can be drawn more easily, typing a single line command.

This loop procedure can then be defined as TRIANGLE and saved for later use as a command.

Computer Concept's Logo II is the most powerful Logo available for the BBC Micro. To make a series of ever-growing triangles involves a recursive use of variables:-

```
TO SERIES
  MAKE SIZE = 50
  REPEAT 10, TRIANGLE: MAKE
    SIZE = SIZE + 10
  END
```

All Logos have a PENUP and PENDOWN command so that lines need not appear connected, but even BBC Logo II lacks an ERASE command, which could allow animation and the removal of accidental lines.

BASIC FACTS

Basic is often condemned as harmful to the development and understanding of programming principles. But one of the most vociferous of its critics, Roy Atherton, has said that BBC Basic 'represents a major advance in educational and practical terms'.

There are four areas where BBC Basic really scores over other popular versions.

1 Long variable names

Compare:

```
FOR I=0 TO N STEP S
```

with:—

```
FOR total=0 TO GRANDTOTAL STEP GROSS SUBTOTAL
```

The second statement is clearly much more comprehensible than the first. Its great merit is that it is effectively self-documenting, and this makes subsequent debugging and alteration much easier.

Many Basics seem to allow the use of long variable names but actually recognise only the first two letters: this means that they will not differentiate between GRANDTOTAL and GROSS__SUBTOTAL.

Moreover, embedded keywords, such as the TO in GRANDTOTAL or SUBTOTAL, can cause problems in many versions of Basic.

But in BBC Basic all letters of a variable name are recognised, and embedded keywords are allowed.

A variable cannot begin with a keyword, which is why we write 'total' and not 'TOTAL'. In BBC Basic, all keywords use upper case letters only, which gives you complete freedom to use lower case characters when naming variables.

2 Procedures, multi-line functions and local variables

Compare:—

```
X=T1:Y=T2:R=T3: GOSUB 2000
```

with:—

```
PROCCIRCLE (XCENTRE,YCENTRE,DIAMETER/2)
```

or:—

```
N=N1:S=S1:U=U1: GOSUB 1000:A=A1
```

with:—

```
ANSWER=FNNET__PROFIT (NUMBER, SIZE, UNIT__COST)
```

In both cases, the self-documenting benefit is obvious. The parameters in both the procedure and the function are passed to local variables in the procedure definition.

So if the definition for PROCCIRCLE begins DEF PROCCIRCLE (XCOORD, YCOORD, RADIUS) then any value taken by XCOORD, YCOORD and RADIUS within the procedure are purely local to it and will not affect their values if they are also used in the rest of the program.

Moreover, if the procedure also uses the variables STORAGE and PLACE it can ensure that these are local too by writing LOCAL STORAGE, PLACE before the first use of STORAGE and PLACE within the procedure.

In this way, one can safely build up and use a library of procedures (and functions) without fear of contamination from variables used in the library routine. And one can safely utilise recursive definitions of procedures (and functions) while retaining initial condition values.

In BBC Basic, functions can be exceptionally complex constructions of many lines. The function is terminated by the equals sign, followed by the required return value of the function.

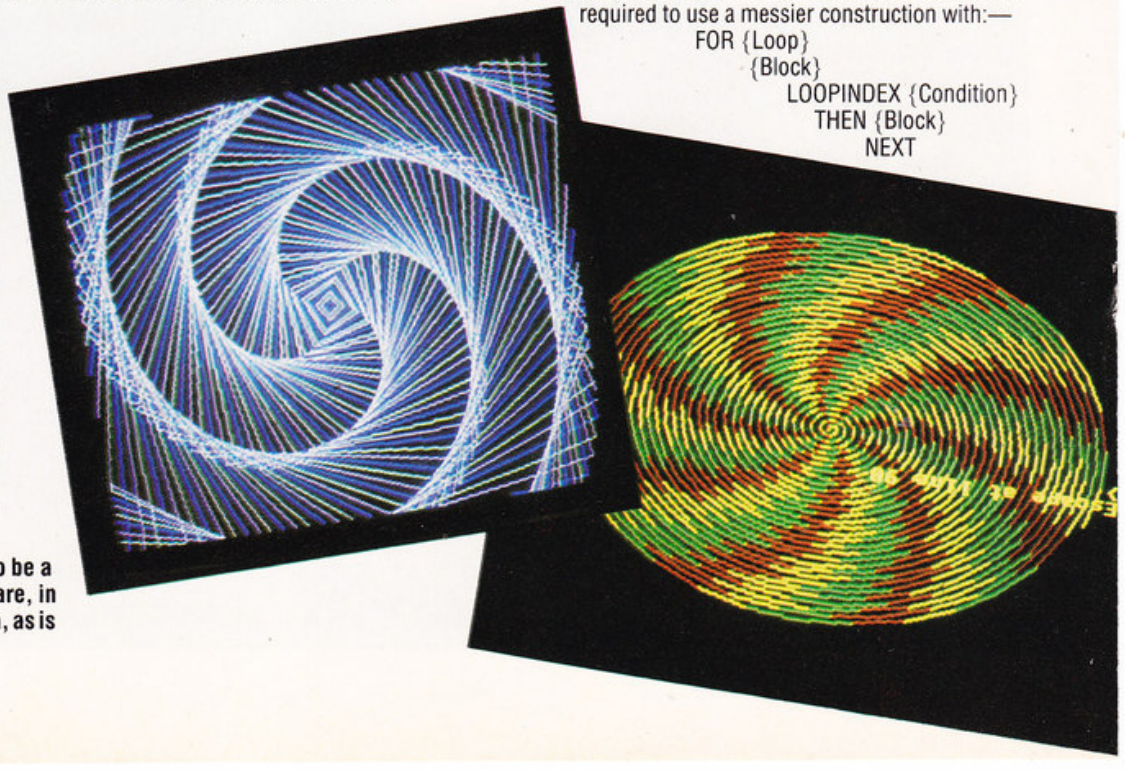
Many versions of Basic allow only one-line functions. And few utilise local variables, despite all the benefits they confer. Still fewer of them implement procedures at all.

3 Some extended structure

All Basic dialects allow the IF . . . THEN construction. Some extend this to IF . . . THEN . . . ELSE. BBC Basic is one of these, but only on a single-line basis. However, this is not as great a disadvantage as it might seem since a line in BBC Basic can be up to 238 characters long.

Few Basics implement the REPEAT . . . UNTIL structure. Without this structure, one would be required to use a messier construction with:—

```
FOR {Loop}
  {Block}
    LOOPINDEX {Condition}
      THEN {Block}
        NEXT
```



Although these graphics may seem to be a product of BBC Basic language, they are, in fact, a product of the operating system, as is the machine's sound capabilities.

at the test condition point.

Unfortunately, the other vital structure REPEAT-WHILE . . . ENDWHILE is missing from BBC Basic (as it is from virtually all Basics), and this means that if you want to implement a loop structure which allows exit before the loop is performed, you must fabricate one using:—

IF {Condition}
THEN {Block}
ELSE GOTO . . .

It is important that, when listing, the structure inherent in a well-written program can be displayed by use of indentation at the entry and exit points of loops. BBC Basic goes some way to providing this with its LISTO command.

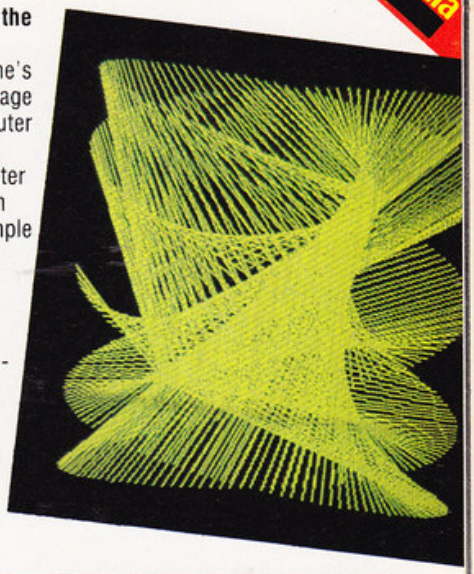
Almost all Basics implement an ON . . . GOTO/GOSUB structure, and it would have been helpful if BBC Basic had extended this to ON . . . PROC; but unfortunately it does not.

4 A built-in assembler with easy access to the resident operating system.

The BBC's built-in assembler is one of the machine's most valuable assets. Working in assembly language allows a valuable insight into the way the computer functions, as well as providing increased speed.

Also unusual in the BBC dialect is an interpreter that contains an assembler entered directly from Basic. This allows convenient editing and the simple mixing of Basic and assembler, which in turn allows other facilities.

All the main operating system functions are available to the assembler (and some are available in BBC Basic also) and these are well documented in the user guide. Acorn has done this because of the importance of writing software in a way which is compatible with the Tube, a high-speed interface which allows the connection of other processors.



SPECIAL WORDS

ADVAL (OS) Reads the appropriate analog port.
AUTO Generates Basic line numbers.
CALL Allows Basic variables to be passed to a machine-code program.
CLG (OS) Clears graphics area (leaves text windows alone).
CLS (OS) Clears text area (leaves graphics windows alone).
COLOUR (OS) Changes text colours.
COUNT Counts characters output to any output channel for current line.
DELETE Deletes a contiguous group of Basic lines.
DIV Integer division (*ie* result with no remainder).
DRAW (OS) Draws a continuous straight line.
ELSE Used in IF . . . THEN . . . ELSE etc.
ENDPROC Terminates a procedure definition (which begins DEF PROC).
ENVELOPE (OS) Defines a frequency and amplitude envelope.
ERL The line number at which the last 'error' (including ESCAPE) occurred.
ERR The error number of the last 'error'.
ERROR Used in ON ERROR to provide error trapping.
EVAL Applies Basic's evaluation routines to user input. Very powerful.
GCOL (OS) Changes graphics colours.
HIMEM Sets the top of usable memory below screen.
INKEY (OS) Reads input with time limit. Can also be used to sense if one or more keys pressed.
INSTR Returns position of substring in a string.
LINE Used with INPUT to allow unrestricted input.
LISTO Produces structured output with LIST.
LOCAL Sets local variables within procedures and functions.
LOMEM Sets bottom of usable memory above Basic program.
MOD Remainder on integer division.
MODE (OS) Selects screen mode. There are eight possibilities ranging from 640 × 256 pixel two-colour display, through 160 × 256 pixel 16-colour display to standard teletext.
MOVE (OS) Move to specified point on graphics screen.
OLD Allows recovery from NEW.
OPT Specifies assembler output (assembler is entered with [and exited with]).
PAGE Allows multiple programs in memory.
PLOT (OS) The main graphics command — allows plotting, drawing and filling.
POINT (OS) Allows pixels to be read.
POS (OS) Horizontal position of text cursor.
PROC Accesses a specified procedure.
RENUMBER Very fast renumbering of a Basic program.
REPEAT Part of REPEAT . . . UNTIL structure.
REPORT Gives text of last 'error'.
SOUND (OS) Sound generation command — there are three channels plus a white noise channel, any of which may be combined.
STRING$ Outputs copies of a specified string.
TAB (OS) Two-dimensional tabulation, text and graphics.
TRACE Fairly simple program trace.
UNTIL See REPEAT.
USR Allows access to the 6502's registers on return from a machine-code program.
VDU (OS) General-purpose entry to OS screen-handling functions like windows, user-defined characters and colour palette changes.
VPOS (OS) Vertical text cursor position.
WIDTH Sets a limit to computer 'page' width.
★FX (OS) Allows access to many general-purpose OS functions such as disabling cursor keys, disabling ESCAPE, turning on the printer and acting on various input/output buffers.
★OPT (OS) Allows filing error messages and actions to be altered.
★KEY (OS) Allows programming of the user-defined keys.
★LOAD/ ★SAVE (OS) Allows filing control over any section of memory.
★TAPE/ ★DISC/ ★ROM/ ★NET Easy interchange between filing systems.
★EXEC/ ★SPOOL Allows loading & saving of ASCII files.

USER GUIDE EXTRA

You may have been told the User Guide is the bible of the BBC, but you'll soon find all you need isn't there. Even guide author John Coll believes that it probably needs a supplement.

Here are some of the things that are missing from the user guide or are very difficult to find.

One of the first problems you're likely to have is screen flutter caused by the BBC's interlaced PAL video signal. To fix this jitter, use the operating system command *TVO, 1 and then hit Return.

The change may not be apparent immediately, as it doesn't take effect until you next change modes. Now try Mode 3 and Return — you should achieve a stable picture.

The *HELP and the *FXO commands are actually missing from the user guide. Either will give details of the operating system, and tell you what extra ROMs are installed in the machine and what programs are on those ROMs.

The *FXO operating system command is one of a number of *FX commands not mentioned in the guide. The *FX 16,0 turns off the analog-to-digital converter on the Model B (theoretically allowing programs to run faster).

The *FX 4,2 command (on the 0.1 operating system) enables cursor keys to be read with INKEY or GET. But on the 1.0 operating system those commands would be *FX4,1 to enable the cursor keys to read INKEY or GET and *FX4,0 to return them to normal.

Some cassette-loading fixes on the 0.1 operating system are also omitted (see part 2 of this Micropaedia), as well as detailed commands for use with the Disk Operating System, Econet and the Teletext adapter.

Instructions and operating details for each of these add-ons will, however, be included when you buy them.

BBC boards being assembled and tested by Acorn at Race Electronics in Mid-Glamorgan, South Wales. Here a chip is being checked.

Likely Languages

Three popular languages besides Logo are advertised for the BBC micro.

All are available from Acornsoft, the software wing of Acorn Computers. Forth and Lisp are put on to the BBC using cassette tapes and books, and the BCPL heavy systems language will be available on a 16K ROM chip from Acorn.

Acornsoft's Forth is claimed to be a complete implementation to the 1979 standard specification, and will run on the Model B significantly increasing program execution speed.

Acorn claims up to five times the speed of Basic, although you will have to get used to the idiosyncratic Reverse Polish Notation employed by the language.

The Forth package costs £7.50 and includes a Forth dictionary, compiler, tape interface or screen editor, macro-assembler and high-resolution graphics demonstration.

The company also offers a version of Lisp, the language used in artificial intelligence research, which includes a 5.5K machine code interpreter, and 3K of initialised Lisp workspace with constants and utilities. This package also costs £7.50.

Contributors: Ian Birnbaum, Nigel Cross, David Graham, Jeff Taylor, Geof Wheelwright and Sheridan Williams.

Next Week

Part 2 of Anatomy of the BBC Micro looks at peripherals, word-processing on a chip and games for the Beeb.

And in two weeks' time, the third and final part of our first Micropaedia series highlights the Acorn Econet system, the mysterious plastic turtle and a guide to servicing your BBC.