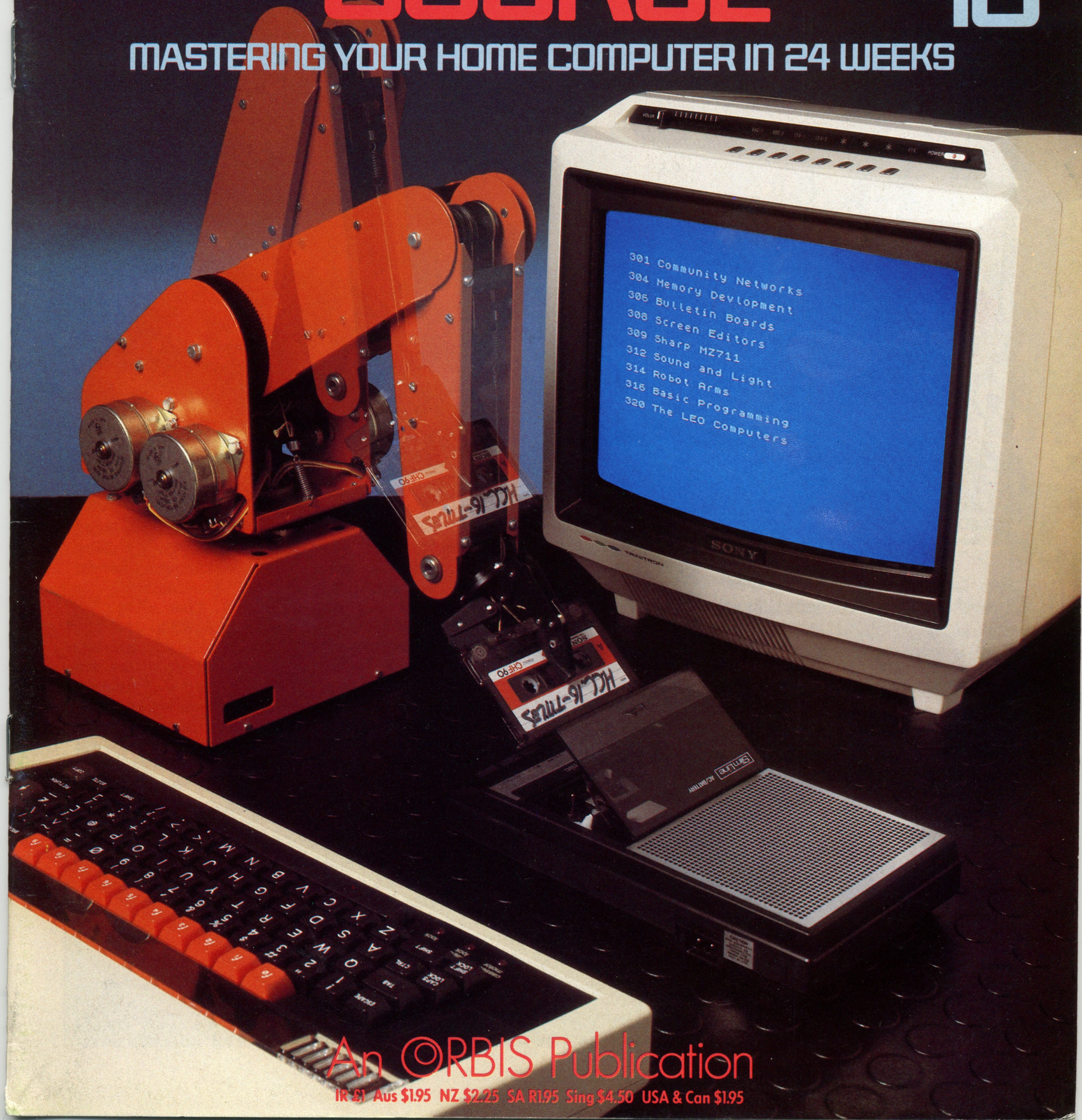
ISSN 0265-2919

R 80p

THE HOME COMPUTER COURSE 16

MASTERING YOUR HOME COMPUTER IN 24 WEEKS



An ORBIS Publication

IR £1 Aus $1.95 NZ $2.25 SA R1.95 Sing $4.50 USA & Can $1.95

CONTENTS

Hardware

Past Memories We trace the development of computer memory and look at possibilities for the future — 304

Sharp MZ-711 We review this rather under-rated Japanese home computer — 309

Software

Board Meeting Bulletin Boards are an easy and inexpensive way to get in touch with other computer users — 306

Basic Programming

Expanding Files We examine file structures in our programming project — 316

Insights

Wired Society As well as numerous television channels, cable distribution could bring a range of other services into the home — 301

One-Armed Bandits Robot arms are now available for even the cheapest home computers. We see how they operate — 314

Passwords To Computing

Editorial Control We study the various methods of program editing directly on the screen — 308

Pioneers In Computing

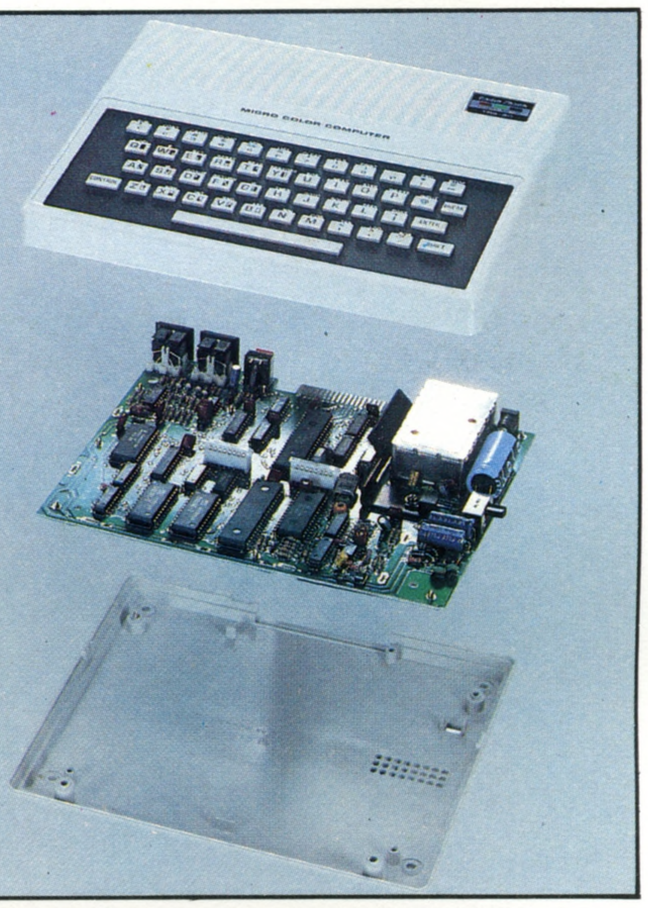
Lyons' Share The corner teashop was the driving force behind the development of the first British commercial computer — 320

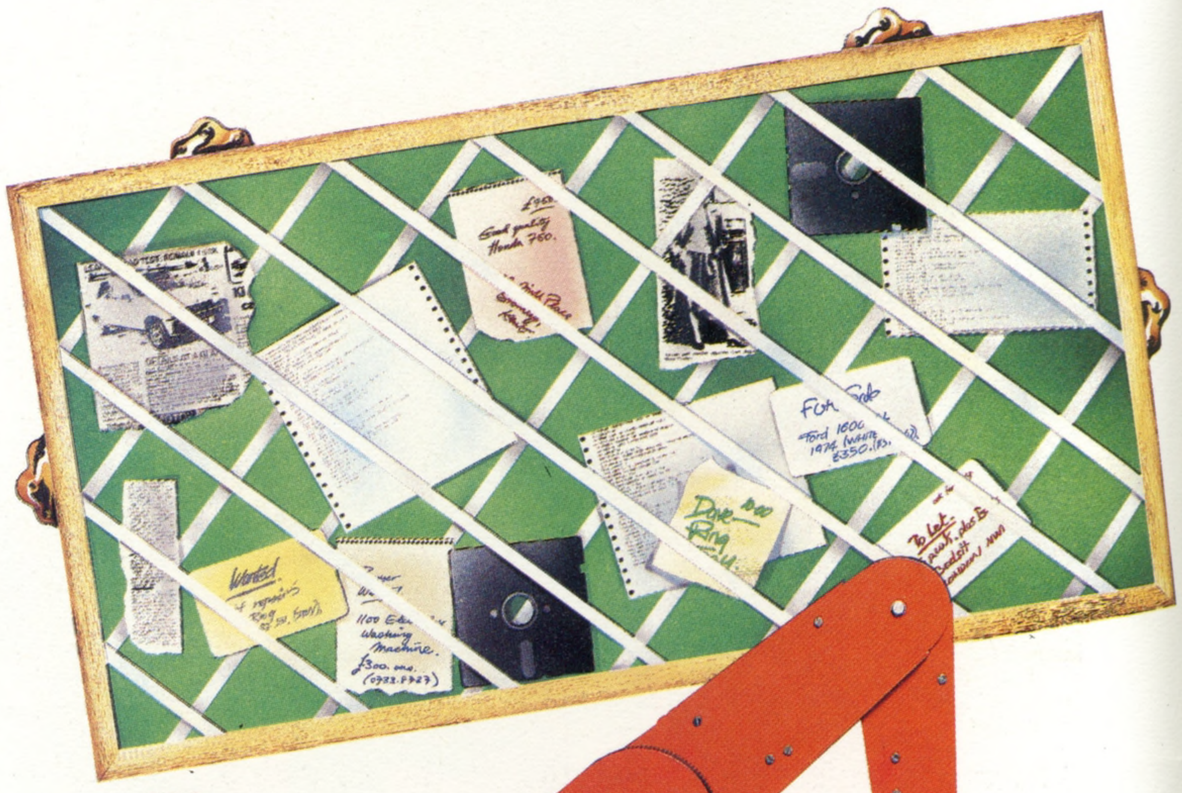
Sound And Light

Simple Sounds...Primary Pictures We look at graphics on the Commodore Vic-20 and sound on the Sinclair Spectrum — 312

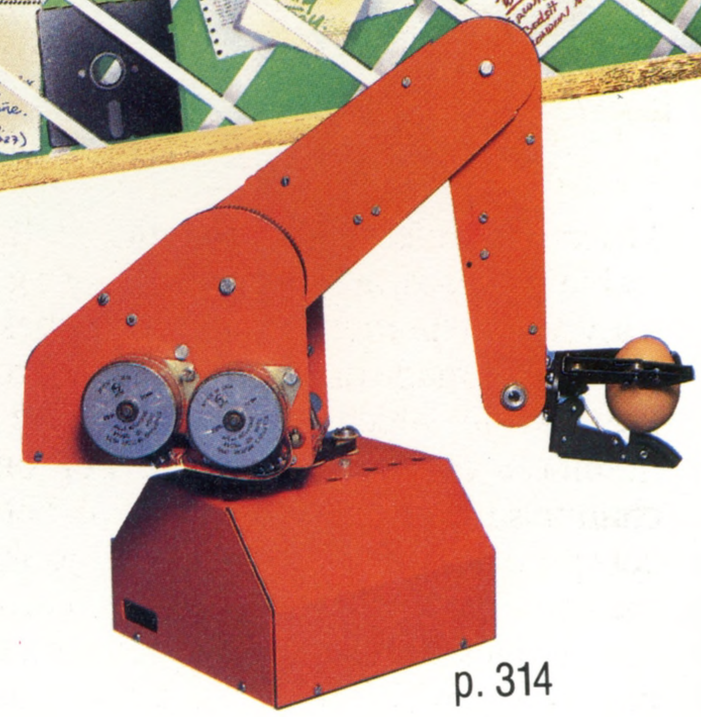
Next Week

- We look at the Tandy Model 10, a home computer designed for the beginner, but with many features found on more sophisticated machines
- 'Ergonomics' is a computer term that means designing machines that are easy and pleasant to use. We examine how this applies to home computers
- The Disk Operating System keeps track of where everything is stored on a disk. We explain how this complex piece of software works





p. 306



p. 314

Editor Richard Pawson; **Consultant Editor** Gareth Jefferson; **Art Director** David Whelan; **Production Editor** Catherine Cardwell; **Staff Writer** Roger Ford; **Picture Editor** Claudia Zeff; **Designer** Hazel Bennington; **Art Assistants** Liz Dixon, Safu Maria Gilbert; **Sub Editors** Robert Pickering, Keith Parish; **Researcher** Melanie Davis; **Contributors** Tim Heath, Henry Budgett, Brian Morris, Lisa Kelly, Steven Colwill, Richard King, Geoff Nairns; **Group Art Director** Perry Neville; **Managing Director** Stephen England; **Consultant** David Tebbutt; **Published by Orbis Publishing Ltd: Editorial Director** Brian Innes; **Project Development** Peter Brookesmith; **Executive Editor** Chris Cooper; **Production Co-ordinator** Ian Paton; **Circulation Director** David Breed; **Marketing Director** Michael Joyce; **Designed and produced by Bunch Partworks Ltd; Editorial Office** 85 Charlotte Street, London W1;

HOME COMPUTER COURSE – Price UK 80p IR £1.00 AUS $1.95 NZ $2.25 SA R1.95 SINGAPORE $4.50 USA and CANADA $1.95
How to obtain your copies of HOME COMPUTER COURSE – Copies are obtainable by placing a regular order at your newsagent.
Back Numbers UK and Eire – Back numbers are obtainable from your newsagent or from HOME COMPUTER COURSE. Back numbers, Orbis Publishing Limited, 20/22 Bedfordbury, LONDON WC2N 4BT at cover price. AUSTRALIA: Back numbers are obtainable from HOME COMPUTER COURSE. Back numbers, Gordon & Gotch (Aus) Ltd, 114 William Street, PO Box 767G Melbourne, Vic 3001. SOUTH AFRICA, NEW ZEALAND, EUROPE & MALTA: Back numbers are available at cover price from your newsagent. In case of difficulty write to the address in your country given for binders. South African readers should add sales tax.
How to obtain binders for HOME COMPUTER COURSE – UK and Eire: Please send £3.95 per binder if you do not wish to take advantage of our special offer detailed in Issues 4, 5 and 6. EUROPE: Write with remittance of £5.00 per binder (incl. p&p) payable to Orbis Publishing Limited, 20/22 Bedfordbury, LONDON WC2N 4BT. MALTA: Binders are obtainable through your local newsagent price £3.95. In case of difficulty write to HOME COMPUTER COURSE BINDERS, Miller (Malta) Ltd, M.A. Vassalli Street, Valletta, Malta. AUSTRALIA: For details of how to obtain your binders see inserts in early issues or write to HOME COMPUTER COURSE BINDERS, First Post Pty Ltd, 23 Chandos Street, St. Leonards, NSW 2065. The binders supplied are those illustrated in the magazine. NEW ZEALAND: Binders are available through your local newsagent or from HOME COMPUTER COURSE BINDERS, Gordon & Gotch (NZ) Ltd, PO Box 1595. Wellington. SOUTH AFRICA: Binders are available through any branch of Central Newsagency. In case of difficulty write to HOME COMPUTER COURSE BINDERS, Intermag, PO Box 57394, Springfield 2137.
Note – Binders and back numbers are obtainable subject to availability of stocks. Whilst every attempt is made to keep the price of the issues and binders constant, the publishers reserve the right to increase the stated prices at any time when circumstances dictate. Binders depicted in this publication are those produced for the UK market only and may not necessarily be identical to binders produced for sale outside the UK. Binders and issues may be subject to import duty and/or local taxes, which are not included in the above prices unless stated.

Wired Society

The growth of cable television provides a further avenue for communication between home computer users

IAN McKINNELL

Circuit City
Milton Keynes, a town built from new beginnings, was supplied with cable television right from the start. 22,000 homes in the town are supplied with seven television channels (six live and one that shows films), and six VHF radio channels. The next step towards an integrated information network allows the supplies of electricity and gas to be metered centrally, rather than in every building. Local Area Networks, to be installed in public buildings, will soon be added, as will a public viewdata system

Much is made of the benefits of the so-called 'cable revolution', which will bring 20 or 30 television channels into our homes, but little mention is made of a side-effect that could change the way in which we communicate with other members of the community. For one. of those channels may be set aside as a link between home computers. We have already looked at two methods of using home computers as communications terminals: viewdata systems (see page 268), which allow access to a large general database, and Local Area Networks (page 218).

Using a viewdata system like Prestel's Mailbox, it is possible to send messages between subscribers, and any subscriber to Prestel can buy goods and services (holidays, for example) directly from a central Information Provider. However, these two services are limited in their application. Similarly, if your computer is one of the workstations on a Local Area Network, you can communicate with any other station — but the largest and most powerful microcomputer based network will extend over only two or three kilometres, and your bank, butcher and insurance broker are hardly likely to be part of it. You could, of course, be both a Prestel subscriber and a network member (in fact, that would be a very sensible way of spreading the cost of Prestel). But that is still a far cry from being a member of a community whose every part is linked electronically to every other.

The most important physical factor preventing the creation of city-wide networks of this type is the signal loss in the transmission medium, whether it be twisted pairs of cables, co-axial cables (like the aerial lead on your television set) or optical fibres. It is this phenomenon that limits the size of Local Area Networks at present, and the only way to overcome the problem is by inserting 'booster' or re-amplifier stations at frequent intervals in the network.

A second consideration is that of 'traffic density', or the volume of information to be communicated. This influences the bandwidth, or range of transmission frequencies, required. As a general rule, we need two hertz of bandwidth for each bit per second that we wish to transmit. A 300 baud (300 bits per second) transmission requires 600Hz; a 1,200 baud transmission needs 2.4kHz. Normal speech uses a minimum of 3kHz across a telephone line, so this is the nominal bandwidth of a telephone line. To transmit a colour television picture, however, requires 8MHz — three thousand times as much as speech. In other words,

the bandwidth required to carry a television picture could carry 3,000 separate telephone conversations.

The capacity of a transmission medium is perhaps best expressed in terms of the number of telephone conversations it can hold. The largest capacity cable used by British Telecom at the present time is capable of handling 20,000 conversations at once, but in experimental tests BT has successfully evaluated co-axial cable as high as 500MHz — making it potentially capable of carrying 167,000 conversations at once. The cable itself is one cm (3/8th inch) thick. Contrast that with fibre optic technology, in which a single strand of glass thinner than a hair offers the ability to carry up to 10,000 telephone conversations.

The communications hardware is already on the market and it would be a simple matter to network any town or city. Let us consider how we could set up a community computing network based on the BBC Model B Micro, using Acorn's Econet. When we looked at Local Area Networks, we noted that the furthest any workstation could be from the network controller was 500 metres (1,650 feet), which means that our network can cover a circle one kilometre (1,100 yards) in diameter. We saw also that it can accommodate up to 254 members, leaving one station dedicated to file serving (handling the communal disk) and another to controlling the printer. But if we were to reserve one further station as the channel of communication to a second network, then a fairly simple piece of programming would allow us to link two networks together. The link would require two machines dedicated to passing messages from one network to the other, communicating via their respective parallel ports. The more machines we were prepared to reserve for this purpose, the more networks could be interconnected.

Of course, this is very much a makeshift solution to the problem of wiring a community together, but if it proved popular it could be superseded by a purpose-built network link. However, it is unlikely to be implemented since it requires all its members to use the same type of microcomputer. To be really effective, a network of this type must be completely 'transparent', which means it must allow a Spectrum, for example, to communicate with a Dragon 32. This requires a central system controller to handle the conversion between the various protocols used by different makes of computer. The controller could also be the link point into Prestel and other viewdata services, the banking system and building societies, the Health Service and other public utilities, and all the other areas of society that have been computerised without any consideration for machine compatibility.

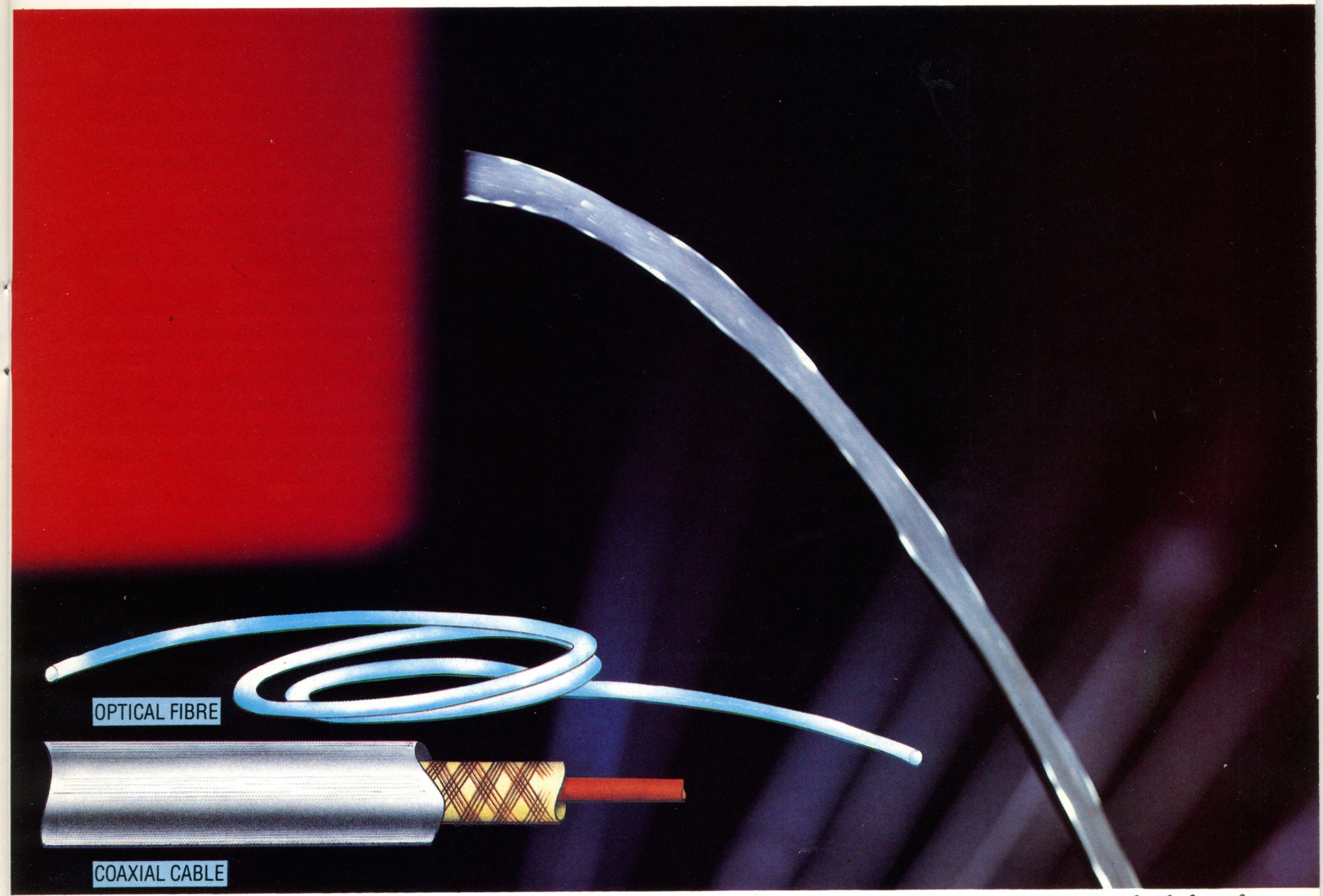
Who would provide the funds to install and run such a service? The British experience suggests that this is the stumbling block. The British government, when it first made proposals for a cable television system to be implemented, emphasised the benefits that such a system would bring to the information technology industry, and consequently a number of the potential users of such a system set up research projects. A national supermarket chain even went so far as to set up a pilot tele-shopping scheme of its own to test its feasibility and the public reaction to it. The scheme did not survive even its short trial period, and the same conclusion — that such services were rather unpopular — was reached elsewhere.

When the draft bill proposing cable television was presented to Parliament, it still contained references to computer services such as tele-shopping and home banking, but said more frankly that the cable authority would be only 'expected to encourage the provision of two-way computer services'. The emphasis had thus changed from information technology to entertainment technology. In fact, cable operators will be specifically forbidden to offer any form of telephone service, even video conference facilities, though a draft strategy paper envisages the cable network eventually taking over all telecommunications services.

COURTESY OF MTV

Satellite Station
MTV uses a satellite distribution system to beam programs to its 1,650 affiliates, who then despatch it via cable to the homes of the 13.5 million individual subscribers across the continental United States. MTV provides 168 hours of programs every week, all with stereo sound. Each hour contains eight minutes of advertising. Though the service is strictly one-way, the company does encourage a degree of audience interaction via charge-free telephone numbers. With the introduction of two-way cable traffic, this kind of instant audience reaction will become far more commonplace

NICK DALY



GARY MARSH

Rapid Pulse
Optical fibres rely on the phenomenon of total internal reflection. Light introduced at one end of a column of transparent material will travel along it, being reflected repeatedly from the inside walls. Losses through the sides of the column will be extremely small. Thus red light from a source in the tank above could be led away efficiently by the stream of water. Similarly, signals in the form of a rapidly fluctuating light beam can be transmitted over great distances along an optical fibre. To achieve this result the purity level of the glass used must approach 100 per cent.

Coaxial cables, familiar from their use as connections to TV antennae, are composed of a single strand of heavy duty copper wire, surrounded by a screen of finer wire woven into a tube. These two conductors are insulated from each other

The experience of other countries is similar. In the United States, where cable television has long been established, there is still little use of the cable system to provide a two-way digital communications medium. Even where it is used, it seems to be for rather trivial purposes. In one well-publicised incident, audiences of Shakespeare's *Hamlet* were asked to vote on how the characters should be developed. Most of those who responded voted to stop the play!

In Scandinavia and Holland there has been a movement towards community-based interconnecting home computers. Local electronic mailing and computer-based baby sitting networks are now being operated, and some communities even hold informal referenda on subjects of local interest. These social benefits have, unfortunately, been largely overlooked by proponents of the cable network systems in Britain. The benefits of communications networks will perhaps be felt most keenly by those members of the community who have little interest in buying even a very simple microcomputer for themselves. Once interactive cables are installed, there will be incentives for service industries such as the banks or the telecommunications service to distribute terminals at little cost — or even completely free of charge, as in some areas of France, where the national telephone network has experimented with replacing telephone directories and the directory enquiry service with viewdata terminals.

©BBC STILLS

Is A Picture Worth a Thousand Words?
It is difficult to determine the number of digital bits required to make up a colour television picture, but considering that the 9MHz bandwidth will allow transmission at 4.5 million baud, and that each whole picture is transmitted 25 times per second, simple arithmetic gives us a figure of 180,000 bits per frame. A thousand words, on the other hand, requires around 60,000 bits

Past Memories

The development of more compact means of storing information has been the aim of designers since the computer was invented

Every computer needs some capacity to store information because even in the simplest processes, such as addition, a digit may have to be stored or 'carried' until it is needed in the next step of the computation. Since the birth of the computer, memory sizes have steadily increased, and this has allowed not only larger problems to be attempted but also types of problems that were insoluble before.

Ever since electricity was first investigated scientists have been looking for ways of using it for storing information. Electricity can be thought of either as a flow of electrons or as a moving wave, but in both descriptions the implicit characteristic is movement. The impossibility of retaining something that by its very nature must always be moving has led to the adoption of indirect storage methods. A battery, for example, stores energy in a chemical form and a hydroelectric power station releases the potential energy in water when it is driven through the turbines. We will see that many solutions have been found to the problem of handling data in the form of electrical signals.

In the Second World War a great deal of work was done with radar to separate the signal reflected from a moving aeroplane from pulses echoing off fixed objects such as trees. A device called a 'mercury delay line' was invented that could temporarily store the pulses of radio waves and thus compare reflected waves at successive 'sweeps' of the radar so that permanent patterns could be eliminated.

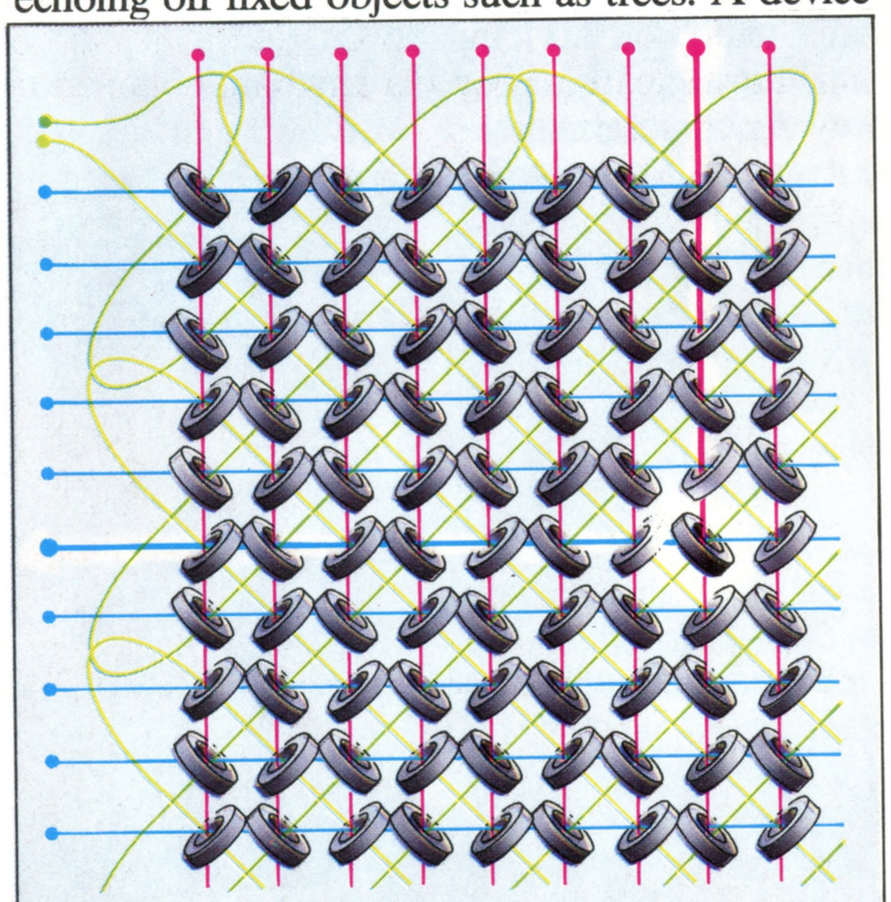
A mercury delay line consists of a glass tube, one metre (39 inches) long, filled with mercury and with a quartz crystal at each end. When an electrical signal is applied at one end, the quartz crystal vibrates and creates a physical wave that travels to the other end of the tube, where it is detected by the other crystal and converted back to an electrical signal. It takes the wave about 660 microseconds to travel down the tube, but by feeding the signal in a loop back through the mercury, pulses can be stored for several minutes before the signal becomes too distorted.

With a clock regulating the train of pulses at 500,000 cycles a second, 330 bits could be stored in a metre tube. The memory size could be increased by lengthening the tube, but only at the expense of increasing the access time, since a pulse could be read only when it was in the electrical circuit outside the tube. These systems were bulky and expensive, requiring skilled engineering to construct sets of tubes to an accuracy of 100th of a millimetre (1/2500th inch).

In Britain another method of storing data was invented by F C Williams at Manchester University, who investigated the use of static electricity generated on the inside surface of a cathode ray tube (a television screen) as the storage medium. An electron gun set up a pattern of static charges on the screen, and this pattern could be detected by a wire grid close to the outer surface of the glass screen. By 1947, it was possible to store 2,048 bits of information on a single screen for a number of hours. Although the speed of access was fast, the static charges had to be refreshed every 30 microseconds, otherwise they would fade and be lost.

The use of magnetic tape was first tried out with LEO in Britain (see page 320) and in the United States on UNIVAC (Universal Automatic Computer) in the late 1940's. This was the first technique by which large amounts of information could be stored cheaply and reliably. Eight bits were stored in a 'frame' across the tape at a density of 6,000 frames per inch (2,360 frames per cm). With a total length of tape of 200 feet (61 metres) or more, permanent memories of at least a megabyte were possible. However, even with the fastest of drive motors, the access time for data in the middle of the tape was some seconds. Consequently, this sort of storage found its natural use in files where the data is required sequentially, as in the calculation of a company's payroll. The tape could be speeded up by floating the 'head' that detects the magnetic signals on a cushion of air, which also relieves frictional wear and tear.

A memory with a large capacity, high access



KEVIN JONES

Core Memory

Iron rings (tori) are threaded onto a lattice of intersecting wires. Any particular ring can be addressed by sending a current through the appropriate horizontal and vertical wires. The reading wire, which threads its way through all the rings, picks up changes in the magnetic flux in the ring, indicating whether a 1 or 0 is stored

speed and low cost per bit was invented at the Massachusetts Institute of Technology in 1950. The 'core' memory used 'tori' (rings) of iron threaded onto a grid of intersecting wires. By passing a current through a wire threading through a ring of iron, the metal became magnetised. There are two possible directions of the magnetic field in the ring and these are used to represent the two binary states, 0 and 1. The direction of the ring's magnetism can be 'flipped' by an appropriate current. And just as a current induces magnetism, the reverse also occurs: when the direction of magnetism 'flips', a small but detectable reverse current is induced. The magnetic field will flip only when a current above a certain threshold is used. By supplying just over half the threshold current through a vertical wire and the other half through a horizontal wire, only one torus (ring) in the lattice will receive sufficient current to flip.

The diameter of the individual rings was eventually reduced from about four millimetres (1/6th inch) to several hundreths of a millimetre, thus reducing the power needed to write data into a cell. Information was read out of a particular memory cell by attempting to flip it. By watching for an induced current it could be discovered whether the cell had flipped or not and hence its original state could be deduced. However, this method of reading set all the cells to one magnetic state, so after each reading the data had to be rewritten.

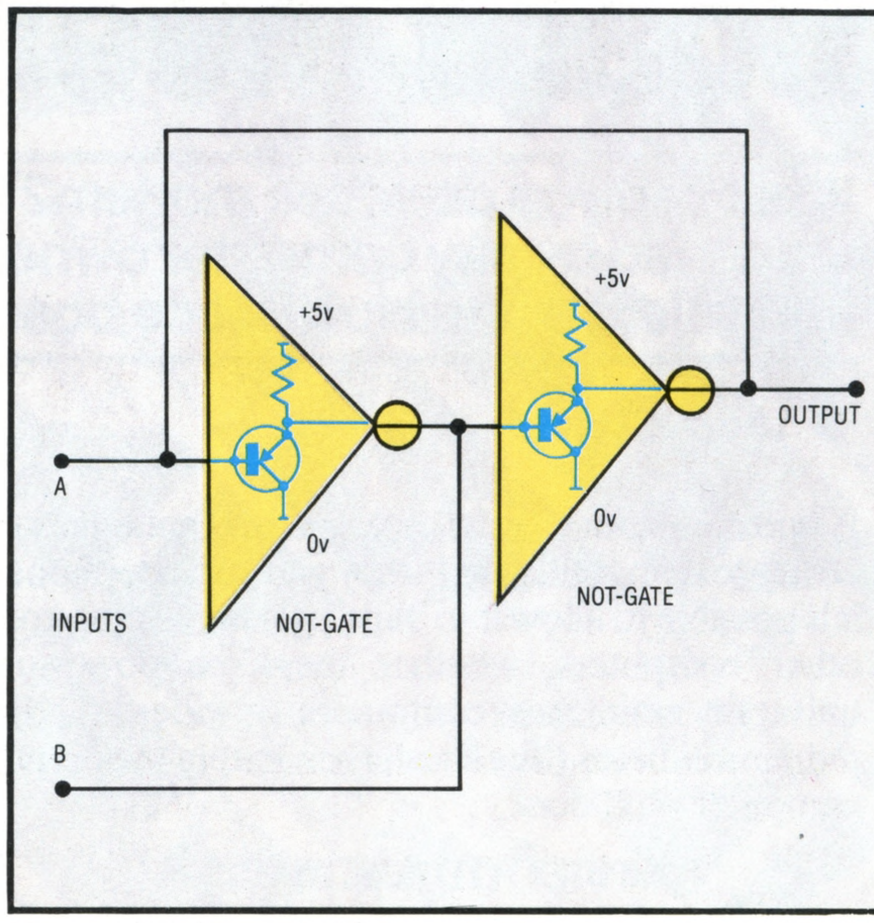
With large-scale integration of transistors onto a single chip, one of the earliest means of holding data — in use since ENIAC — has been revived: the 'flip-flop'. A flip-flop is a device that is stable in one of two states — and is hence referred to as being 'bistable'. It is constructed out of two electronic switches joined back to back, with the output of each switch fed into the input of the other. In this way a pulse can be 'trapped' in the flip-flop until it is needed.

Each bit requires two switching devices, and the early valve flip-flops were expensive, unreliable and frequently burnt out. However, with integrated circuits, in which hundreds of thousands of transistor switches are built into a single chip, the flip-flop is once again important.

'Static' RAM chips, found in all early home computers, contain thousands of bistable circuits — one for each bit. Most of the newer machines, however, feature 'dynamic' RAM. The user won't notice any difference, but dynamic RAM is faster and cheaper to build. It is really like an array of capacitors, each of which can store an electrical charge. The problem is that a stored charge tends to leak away, so the contents need to be 'refreshed' thousands of times every second by special circuitry built into the chip.

The most recent kinds of memory to become available are 'bubble' and optical laser memories. The latter are similar to the discs that are used for storing music or video films and are read by a laser beam. In bubble memory, tiny magnetic domains or bubbles are created within a chip of magnetic material. These bubbles can be moved around in loops and the presence or absence of a bubble represents a binary 1 or 0. The loops are kept small to give rapid access time. Their density is potentially so high that this kind of memory may come to replace the present moving-surface cassette and disk systems.

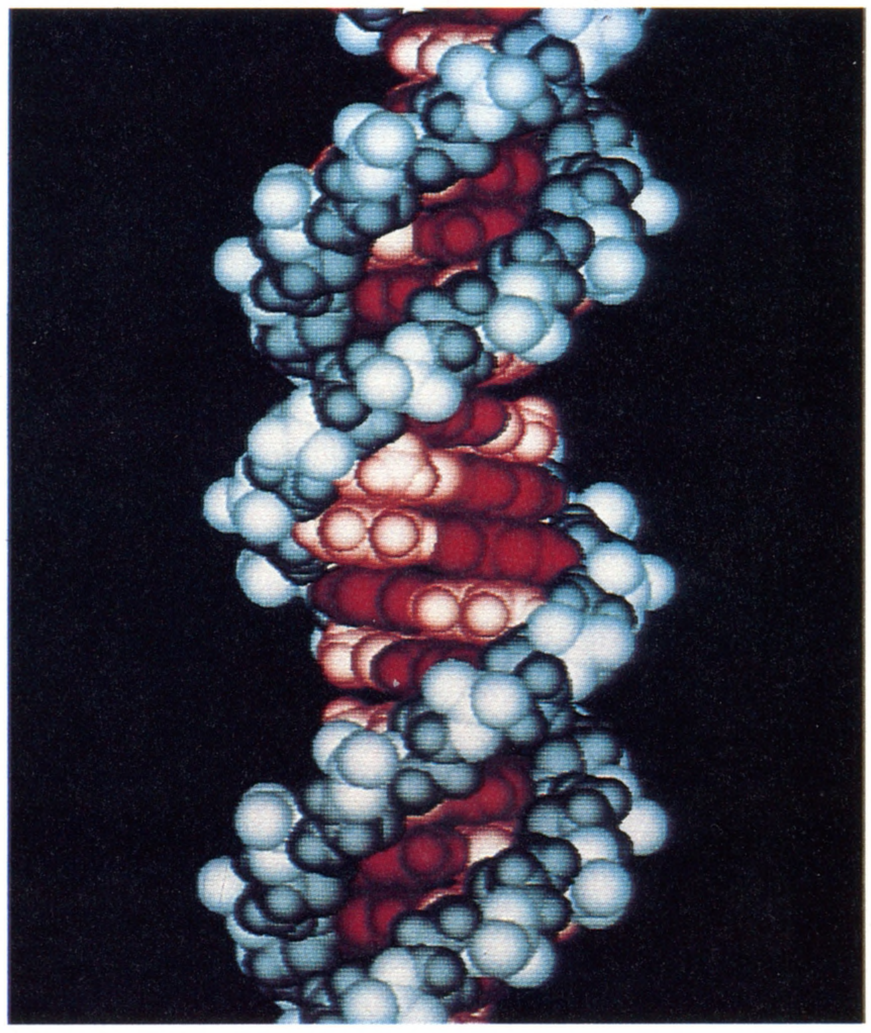
Cryogenic memories may yet become available. At temperatures near to absolute zero, electrical resistance in a medium all but disappears to give what is called superconductivity, and it then becomes theoretically possible to store a charge equivalent to that of a single electron. The charges on which the present type of chip memories operate are very small but each is still equivalent to a few million electrons. With the superconductivity obtained by cooling these memory media to cryogenic temperatures in baths of liquid helium, the speed and capacity of memories leap far beyond anything obtained by present systems.



Flip-Flop

The flip-flop is the basis of most forms of semiconductor memories, and is constructed from two invertors or NOT-gates (the actual circuit of transistors and resistors is shown in blue). The output of each invertor is connected to the input of the other. If the A is set to logic 1, then the output will be a 1. If input B is set to 1 then the output will be 0. Because this circuit has two stable states, it is sometimes called a 'bistable'

KEVIN JONES



Biotechnology
Even though microelectronics has reached an incredible level of miniaturisation, every operation in a semiconductor memory will involve the flow of many thousands of electrons. Scientists are already trying to devise memories that work at molecular level i.e. with individual electrons, and this comes under the heading of biotechnology rather than microelectronics. It is known, for example, that the structure of a DNA molecule stores the genetic code that controls the way in which our bodies grow. It will be many years, however, before we can reproduce this effect in usable form

Board Meeting

What can you do with a modem? Electronic Bulletin Boards can be accessed by any computer owner, and provide the computerised equivalent of the club notice board

A home computer on its own is rather limited in its ability to communicate, but by way of a telephone it is possible to allow it to 'talk' to a whole range of other computers, whether these be powerful university mainframe computers, or wide-ranging public databases (like Prestel) or simply the home computer next door.

COMPUTER TELEPHONE DIRECTORY

(all 300 baud unless otherwise stated)

Bulletin Board Services		
Forum 80 Hull	1900–2300 wkdays	0482 859169
CBBS North East	1430–0100	0207 543555
Liverpool Mailbox	24 hours daily	051 4288924
TBBS Blandford	24 hours daily	02584 54494
TBBS London	0900–0100	01 348 9400
Mailbox80 Stourport	1800–0800 daily	0384 635336

Commercial Services		
Rewtel	24 hours	0227 232628
Maptel	24 hours	0702 552941
Distel	24 hours	01-679 1888
Estelle	0700–1800	0279 443511
Estelle (1200/75)	0700–1800	0279 441188
Prestel (1200/75)	24 hours	depends on area
Micronet 800	24 hours	Prestel page *800

Using the 'Bulletin Board Services' (BBS) you can leave messages for other home computer users, advertise goods for sale and even swap programs. A BBS is really the electronic equivalent of the cork bulletin boards found in most club buildings. They are usually set up and run by volunteers using an ordinary home computer with a large storage device such as a disk, and can be accessed by anyone. Distance to the other computer is no object: you can access machines in your own town, at the other end of the country, or even overseas. The only restriction, of course, will be your telephone bill.

Before your home computer can start 'talking' on the phone, however, you need a piece of hardware to connect the two together. Such a device is called a modem (see page 108), and this plugs into the serial (RS232) socket at the back of your computer. Some home computers, such as the Vic-20 and the Spectrum, do not have such a socket, so you might also need to buy a serial interface card to plug in the back. The modem is connected to your telephone line either directly via an extra phone socket, or acoustically using an acoustic coupler. There are two modem 'speeds' (baud rates) in general use: 300 baud and 1200/75 baud. The commercial operators, such as Prestel and British Telecom, use the higher speed of 1200/75 whereas the BBS (which is mostly for home computers) uses 300 baud. 1200/75 means that information is sent to the user at 1200 baud, and from the user to the central computer at 75 baud. Of course, it is preferable to buy a modem with switchable baud rates. Recently, the price of modems has fallen considerably and you can now buy a 300 baud battery-powered modem for about £80. If your modem is a 'direct-connect' type, then you will also need to rent an extra socket for it from British Telecom.

To operate the modem it is also necessary to obtain software that allows the machine to function like a computer terminal. Two types of terminal software are available – 'dumb' or 'smart'. A smart terminal can download programs and save messages from other computers. A dumb terminal is unable to do these things, but is easier to set up and so is more suitable for your first attempts at communicating. Such software is available for the 'older' home computers, such as the Apple II and the Tandy TRS80, as well as for some of the newer models – BTERM is such a program for the BBC Model B. For other computers, this software will often be supplied free of charge by their user groups. If this is not available, it isn't too difficult to write your own 'terminal emulator' software. What you need is a program that sends any characters typed on the keyboard to the RS232 socket, and displays data received from the socket on the screen.

To illustrate how to use a modem, let's go through the steps involved in communicating with a BBS. First, you need to find out the phone number and technical details (such as the baud rate) of the computer you want to access. After running the communications software on your computer you dial the phone number. If the remote computer is 'listening' you will hear a high-pitched tone in the earpiece. This is the 'carrier tone'. If you are using an acoustic coupler, you just insert the receiver into the rubber cups. With a direct-connect type of modem you flick the switch marked 'LINE' or 'DATA' and replace the receiver. Either way your computer should now be connected.

The first thing you will probably see is some kind of 'greeting' on the screen, generated by the computer on the other end of the telephone line. You will then be asked for your identification and/

TELECOM GOLD

Electronic Mail

Telecom Gold is British Telecom's system of electronic mail, a more sophisticated version of the Bulletin Board Services aimed primarily at the commercial market. Registered users can rent a 'mailbox' on a British Telecom minicomputer, to which other users can 'post' messages. These messages can then be read or copied onto the user's computer. Using Telecom Gold, it is possible to access other electronic networks in other parts of the world, with the advantage that all the technical differences between the various users are handled automatically by the network computers

COURTESY OF BRITISH TELECOM

TONY LODGE

Computerised Notice Board
One of the most rewarding applications of a modem is accessing an electronic bulletin board. These are computerised versions of the traditional club notice boards. Messages can be 'pinned on' to be read by anyone, or specific individuals who have the correct passwords. Meetings and secondhand computer equipment for sale are advertised. It is even possible to download games or other program listings onto your disk or cassette

or password. If the service you have dialled is private and you are not a registered user, then you probably will not get much further – unless you are either good at guessing or very patient! A lot of computers will, however, allow restricted access to 'guests' who are not registered users, so it is worth trying a few words like NEWUSER, GUEST or HELP.

The BBS, on the other hand, is open to anyone and is free of charge (other than the cost of the phone call). You just give your name and town when asked, and once 'signed on' you will be asked for details of your computer, such as the screen width or perhaps the specific make. This is to enable the 'host' computer to identify you in future calls and set up the system to function properly with your computer.

Once this has been done you will be given some system information, such as operational times and technical details, and a time limit for your call – perhaps 30 minutes. You now come to the main menu, which consists of a list of half a dozen commands, and you choose the required one simply by pressing the corresponding initial letter. For example, to register as a 'New User' you would type N; to finish your call type G for 'Goodbye'. Further menus will appear with most of the options, and you carry on selecting the relevant option until you reach what you require. The BBS is rather like a tree – you start at the trunk, the main menu, and choose different 'branches' with each sub-menu. The same idea is used with Prestel and most other databases.

The 'New User' section is for those who wish to register with the BBS. Your name and address are requested, and you can choose your own password to use in future calls. The 'Information' command gives detailed technical information about the system. The 'Utilities' section informs you how long your call has lasted, and also gives details on all the other Bulletin Board Services.

The 'Bulletins' section contains public notices that other users have put on the system for anyone to read. You can also add your own notices. The 'Messaging' section is for you to read private messages that people have left for you. Similarly, you can send private messages to other users, or to a group of users who have something in common (for example all BBC Model B owners). These features are perhaps the most exciting aspect of the services, and are probably responsible for their growing popularity.

A slightly different service, called Rewtel, offers a specialised database for electronic components and related information. It also offers direct ordering from the keyboard of stock items, but only for subscribers. Unlike the BBS, you enter 'keywords' to specify your area of interest. For example, if you wanted some help in using the shopping facility you would enter HELP REWSHOP. It also has a BBS facility: you enter the keyword CHALK and you can then leave a message. Non-subscribers can use the service and will be allowed eight minutes on the system. Distel and Maptel are similar database services primarily intended for ordering electronic and computing equipment. The advantage of these commercial systems is that they operate 24 hours a day, and are not as frequently engaged as the Bulletin Board Services.

While mainframe computers have been 'on the phone' for some time, it has been only recently, with the fall in modem prices and the increased sophistication of microcomputers, that such communication has become feasible for home users. Over the next few years the modem is likely to become as common an accessory for the home computer as the printer or cassette recorder.

Editorial Control

Most microcomputers allow programs to be edited on the screen, saving a considerable amount of time and effort

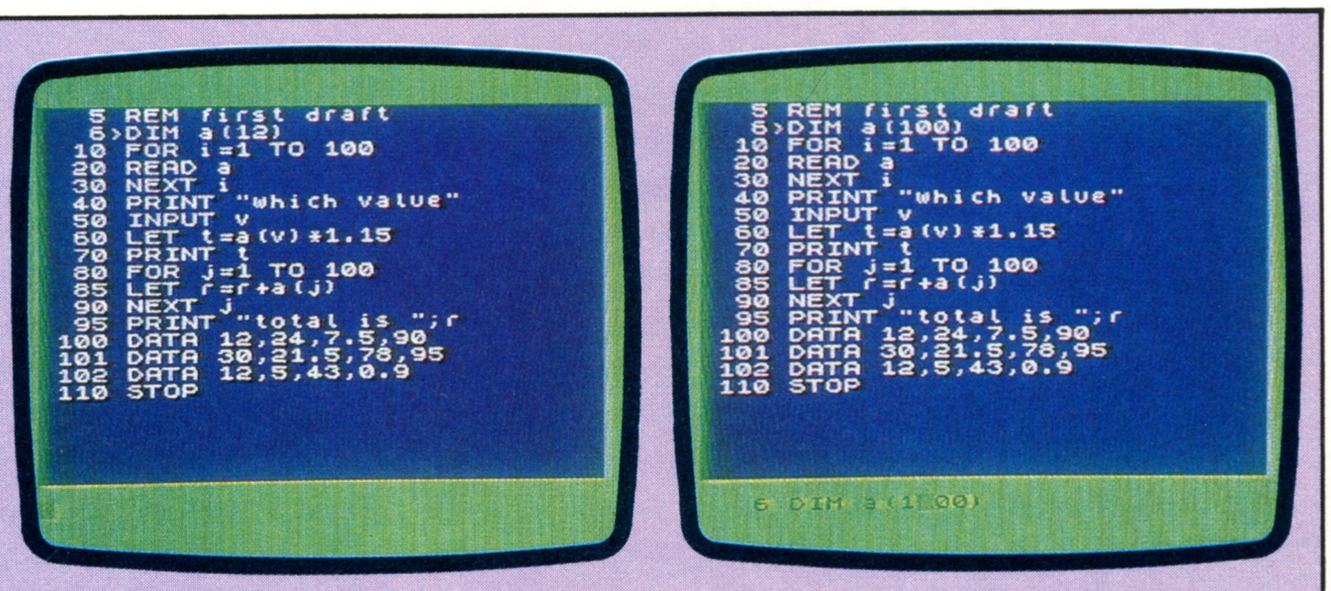
Everybody makes mistakes when they use a computer keyboard, and for that simple reason it is necessary to have editing facilities. There are many situations where we may want to change the data displayed, from simply correcting a typing mistake or altering some erroneous statement, to updating information that has changed since it was first entered.

Many applications programs include some form of specialised 'editor'. The editing functions of a word processor, for example, are designed to cope with the kind of alterations that are made to the draft stages of reports and letters. These include the ability to delete sentences, move whole paragraphs to other places in the text, and change all occurrences of a name or phrase to a new one.

However, almost all home computers have some sort of editor — built into their operating system in ROM — geared to editing program listings. Programs are extremely vulnerable to mistakes. In any program, syntax errors will occur with great regularity, and there will almost certainly be bugs in its operation that will need to be eliminated, not to mention the enhancements that may be required later. The facilities offered by your computer's editor can make a great deal of difference to the development time of a long program. We must stress, however, that a good programmer spends a considerable amount of time testing out the operation of his program on paper before typing the listing into a computer. It is very bad programming procedure to type in the first solution that comes into your head and spend 90 per cent of the program development time de-bugging it.

Line By Line

The editing facilities on a Sinclair Spectrum are considerably better than most types of line editor, though by no means as easy to use as a full screen editor. To change a particular line in a program, the current position marker (>), which appears between the line number and the line itself, must be moved to the correct line using the cursor up and down keys



IAN McKINNELL

The editors on many home computers still behave as though they have only a one-line buffer, when in fact the whole screen is memory-mapped — each character location corresponds to a byte of memory.

A screen editor is far more efficient. It allows you to move text or graphics around the screen with ease. Whenever you press RETURN, the editor reads the whole line on which the cursor is lying into the interpreter, where it is executed (if the line consists of a command) or entered into the program (if it begins with a line number). By using the four arrow keys, the user can move the cursor to any point in the program as displayed, and then insert, delete or overwrite characters as desired.

A screen editor has to be written in machine code to achieve the necessary speed, and it can feature some extremely useful facilities. The best screen editors will allow a listing to be scrolled up as well as down, and allow whole lines as well as individual characters to be inserted or deleted. Some even feature commands similar to those of word processors that find and alter all occurrences of a particular character string.

Editors are becoming more sophisticated and easy to use with each new generation of computers. With the introduction of mice (see page 296) and software that mimics the manual processes of cutting and pasting pieces of text, the time taken to edit a document or listing to its final form is being gradually reduced.

There are two kinds of editors: 'screen editors' and 'line editors'. The former are considerably more flexible and easier to use, but the latter are far more common in home computers. Line editors derive from the days when all computing was done through teletypes or terminals to a remote computer. Teletypes had a buffer memory of just one line of 80 characters (80 bytes). The programmer could obtain a printed list of the whole program by typing LIST, but if, for example, a correction to line 120 was required, it was necessary to type in that whole line again. On some systems typing EDIT 120 would result in a printout of that particular line, and alterations or deletions could then be made with the 'backspace' and 'rubout' keys (insertions were still impossible). Other commands such as DELETE (a specified range of line numbers) were added, but the restriction of having to call up and modify a whole line was still there.

Sharp MZ-711

A low-cost home computer that allows its optional peripherals to fit inside its case

The Sharp MZ-711 is an interesting machine with some quite unusual features. In construction it's up to the usual high standard of Japanese design, and it performs well. The colour is good — very steady and bright — and apart from a slight fuzziness when certain colour combinations are used, it produces extremely readable text.

However, there are aspects of its design that are quite extraordinary. The keyboard, for example, would be highly agreeable if it were not the exact opposite to what you would normally expect: upper case is the default mode, while lower case is obtained by pressing the SHIFT keys. Although this is unquestionably handy for BASIC programming, it makes any other operation extremely difficult. However, it is a simple matter to convert this to the usual arrangement. Pressing CTRL and E simultaneously changes the keyboard from upper case to lower case, while CTRL with F converts it from lower case to upper.

The cassette unit for the Sharp MZ-711 is intended to be an optional extra, as the unit slots into the main casing, but most users will require it. This system is slow but reliable, and every tape seems to load first time. Nevertheless, the cassette operation system does have several weaknesses. There is provision for only the most rudimentary of filenames, and nothing by way of motor control — which could easily have been included. This restricts the usefulness of the deck to virtually one-file-per-tape storage. The limitations of this system are surprising, considering Sharp's expertise in other fields, particularly the hi-fi industry in which their solenoid-controlled tape decks are commonplace.

The built-in printer/plotter (again, an optional extra that fits into the case) has an extremely delicate and vulnerable print head mechanism and pen units. These must be taken out of the machine and capped if they are not going to be used for any extended period. Unfortunately, they are very small units and could get lost when

CHRIS STEVENS

Keyboard
The Sharp MZ-711 keyboard gives a professional feel to the machine. The keys are correctly spaced, full-travel, full-sized, and suitable for word processing. Five programmable function keys, responding to the SHIFT keys, offer ten functions and this makes operation easy. The graphic symbols are arranged in logical sequence while the other keys — being in the QWERTY format — are not. This means that it can be difficult to find which key combinations produce the required symbol

removed from the printer. They also have a tendency to dry up easily and the quality of the print is sometimes difficult to read.

The BASIC interpreter is loaded from tape instead of being built into ROM. This frees the machine for use with alternative languages.

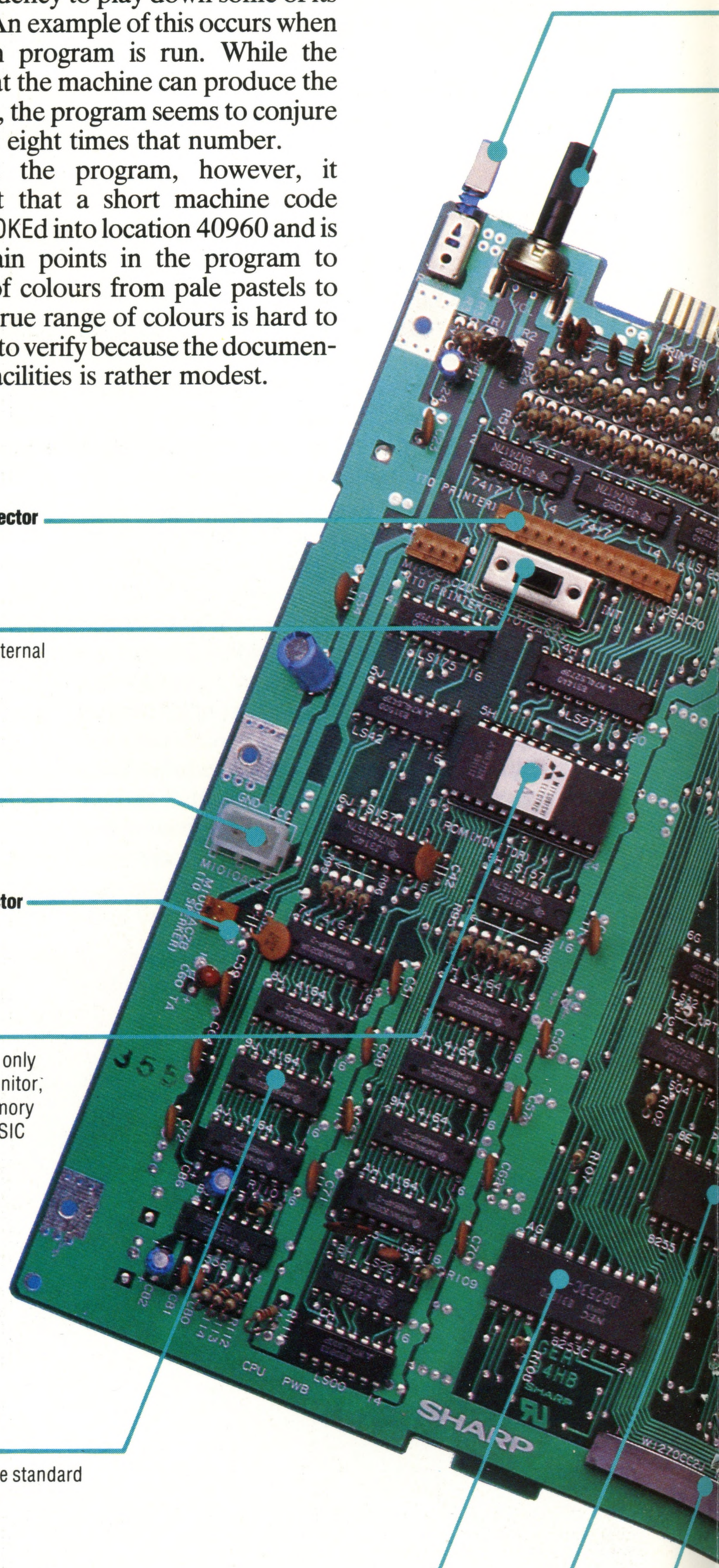
The technical documentation is of a high standard. It features memory maps and circuit diagrams, an Assembly language listing of the system software that is built-in, and other technical details. It isn't comprehensive, however, since it lacks timing diagrams, but it is far better than most manuals.

The Sharp MZ-711's manufacturers have an unaccountable tendency to play down some of its unusual features. An example of this occurs when the demonstration program is run. While the manual implies that the machine can produce the usual eight colours, the program seems to conjure up a range at least eight times that number.

If you inspect the program, however, it becomes apparent that a short machine code routine has been POKEd into location 40960 and is called up at certain points in the program to produce a range of colours from pale pastels to dark shades. The true range of colours is hard to judge. It is difficult to verify because the documentation on colour facilities is rather modest.

Cassette Deck
The cassette deck on the MX-711 is an optional extra, though it can be built into the main casing. The operating system, however, has no control over the cassette motor, which limits its applications

Printer/Plotter
Like the cassette, the printer/plotter is an optional but built-in extra. It uses four miniature ballpoint pens to produce text and graphics on 11.5cm (four inch) paper. Three operating modes permit three text sizes, and hence 26, 40, or 80 columns across the paper



Reset Button

Volume Control

External Printer Port
This is a parallel interface, but not Centronics standard, so it works only with Sharp printers

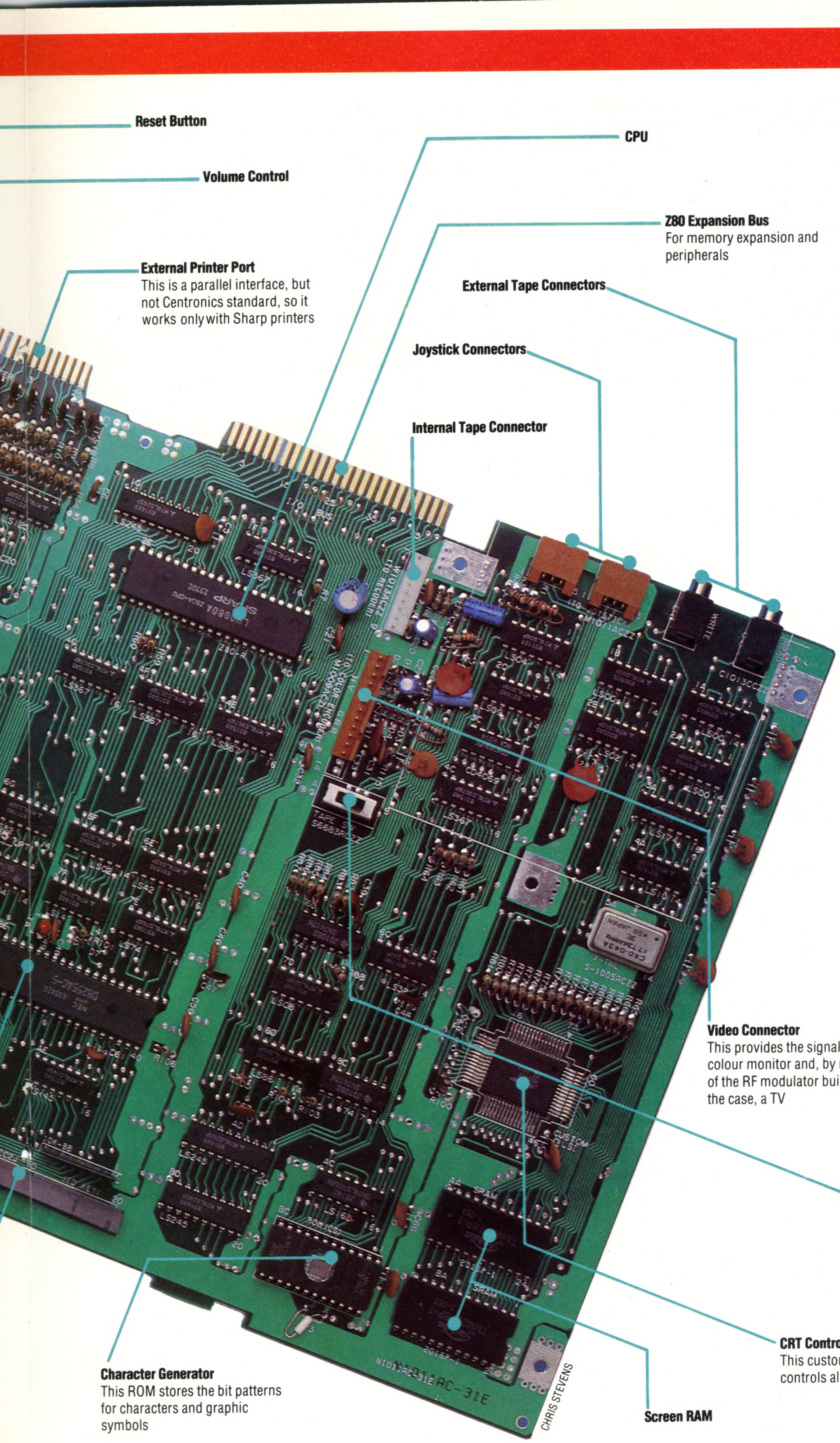
CPU

Z80 Expansion Bus
For memory expansion and peripherals

External Tape Connectors

Joystick Connectors

Internal Tape Connector



Video Connector
This provides the signals for a colour monitor and, by means of the RF modulator built into the case, a TV

Cassette Switch
If your cassette deck consistently won't load programs, this switch will alter the phasing of the signal, and should eliminate the problem

CRT Controller
This custom-designed chip controls all aspects of the video

Screen RAM

Character Generator
This ROM stores the bit patterns for characters and graphic symbols

CHRIS STEVENS

SHARP MZ-711

PRICE
Computer and cassette deck £249.95, printer/plotter £129.95

SIZE
440 x 305 x 86mm

CPU
Z80A

CLOCK SPEED
4 MHz

MEMORY
ROM 12K, RAM 64K

VIDEO DISPLAY
24 lines of 40 characters, or low-resolution (50 x 80) pixel graphics, 8 named colours and an unspecified number of shades, with background and foreground independently settable; 127 predefined characters and 127 user-definable characters

INTERFACES
RS232 serial, cassette, system bus, parallel printer

LANGUAGES SUPPLIED
BASIC on cassette tape

OTHER LANGUAGES AVAILABLE
PASCAL, ASSEMBLER, FORTRAN, FORTH, DAFMOD TOOLKIT

COMES WITH
Installation and BASIC manuals, TV lead, cassette lead, BASIC and demonstration cassettes

KEYBOARD
Full-travel keys and five programmable function keys with shifted and unshifted values

DOCUMENTATION
Very comprehensive and covering important technical details. It is rather dull but serves as a good introduction to the machine

Simple Sounds

Sound generation on the Sinclair Spectrum

The Spectrum is rapidly becoming the most popular home computer. It features excellent colour graphics facilities and useful memory options at a low price. Unfortunately, some facilities have been sacrificed in order to keep the price down. Most complaints concern the keyboard and the use of a non-standard BASIC, but it can be argued that its minimal sound capabilities are its weakest feature. The Spectrum provides the barest essentials for sound generation and produces unrewarding 'music' from a single 'pulse'-type oscillator. It is possible to control the duration of a note and its pitch, but there is no way to alter the tone of a note or change its envelope (see page 276). Another handicap is the standard output, which is through a very small internal piezo-electric speaker that makes the sound of a harsh 'beep'. However, Sinclair have provided alternative signal outputs from the 'mic' or 'ear' cassette ports, suitable for external amplification via a hi-fi system, but because of the lack of sound quality this is a dubious benefit. The Spectrum's sound capabilities do have the advantage of the simplicity of the associated BASIC commands of BEEP and PAUSE, which enable the user to understand the principles of computer-generated sound more easily. And the machine does have a very impressive frequency range of 10 octaves.

Before any sound can be produced on the Spectrum, it is first necessary to either connect it to a hi-fi or make the internal 'beeper' audible by entering the following direct command before RUNning a sound program:

```
POKE 23609,100
```

This also increases the volume of the 'click' feedback that occurs as a key is pressed on the keyboard.

Sound Control

To instruct the computer to output a particular note the BEEP command is used like this:

```
BEEP d,n
```

where 'd' represents the duration of a note, and 'n' represents the pitch of the note. The duration value can be set between 0.00125 and 10 seconds; and the pitch is expressed as the number of semitones from middle C (given a value of 0) in the range from −60 to 69. For example, the following statement plays the note 'A' at 440Hz, which is nine semitones from middle C, for a duration of half a second:

```
BEEP .5,9
```

To play a whole string of notes with an accurately timed space between each one, we can use the PAUSE command:

```
PAUSE ms
```

Primary Pictures

The graphics capabilities of the Commodore Vic-20

Like Commodore's other home computers, the Commodore 64 and the PET, the Vic-20 is a well-constructed machine, but its makers do not give much away in the machine's BASIC instruction set. No special graphics commands are available to the Vic-20 user, who has to have either a very good knowledge of the machine's internal workings or buy one of the accessories designed to make graphics programming easier. However, Commodore provide an extensive set of special characters that can, with a little ingenuity, be put together to produce interesting results.

Sixteen colours are available on the Vic-20 and each character square can contain four colours. The screen display is made up of 23 rows and 22 columns of eight by eight pixel character cells, but characters can also be displayed in a 16 by 8 rectangular format. High resolution graphics are possible on the standard Vic-20 but this is rather difficult for the average user to program.

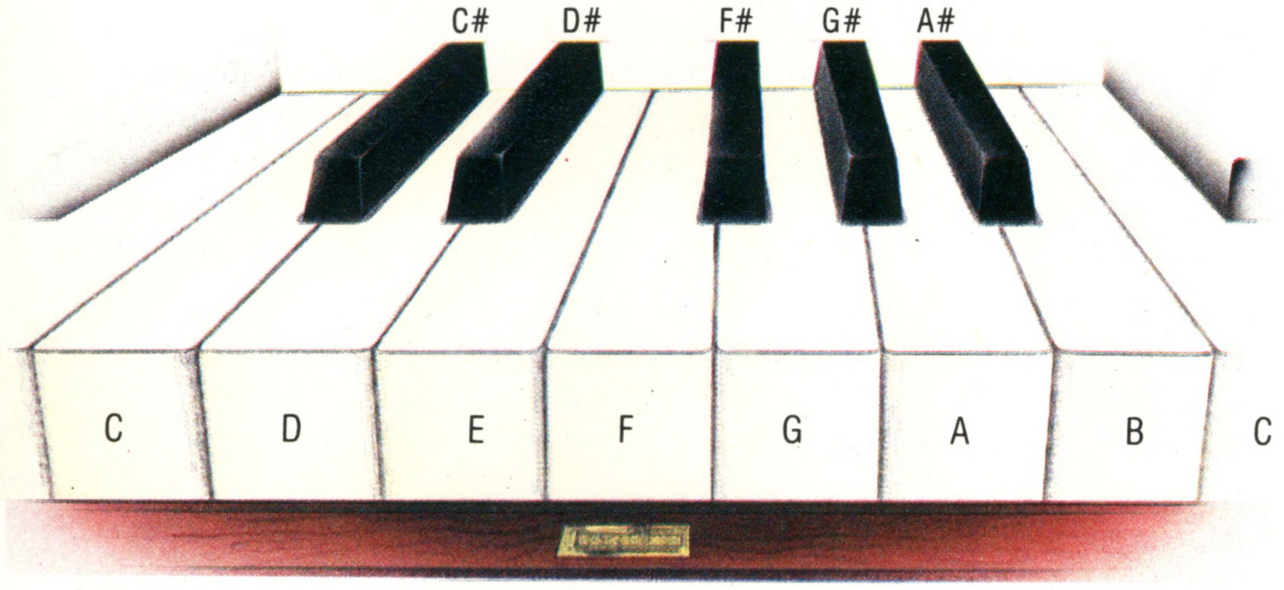
Low Resolution Capabilities

Upper and lower case alphabetic characters are available, as well as more than 60 special PET graphics characters. Two small squares are marked on the front of many of the Vic-20's keys each displaying a particular pattern. In addition to half- and quarter-character squares there are playing card symbols, chequerboard designs,

where 'ms' represents the time in units of 0.001 second (milliseconds). To play an octave in a key of C major (C, D, E, F, G, A, B, C) from middle C, with each note lasting half a second and a PAUSE of a quarter second between notes, the following format can be used:

```
10 FOR I = 1 TO 8
20 READ N
30 BEEP .5,N
40 PAUSE 250
50 NEXT I
60 DATA 0,2,4,5
70 DATA 7,9,11,12
```

This program is a good illustration of the format of a scale. An octave extends from the root note (in this case middle C) to the next note with the same name (the next C above) and spans 12 semitones. It is called an octave because it consists of eight notes in a major scale.



GARY MARSH

C major

Scales And Pianos

By using the INKEY$ command, the Spectrum keyboard can be used to approximate the keyboard of a piano:

```
 10 REM **************
 20 REM *OCTAVE PIANO*
 30 REM **************
 40 IF INKEY$="Q" THEN BEEP1,0
 50 IF INKEY$="2" THEN BEEP1,1
 60 IF INKEY$="W" THEN BEEP1,2
 70 IF INKEY$="3" THEN BEEP1,3
 80 IF INKEY$="E" THEN BEEP1,4
 90 IF INKEY$="R" THEN BEEP1,5
100 IF INKEY$="5" THEN BEEP1,6
110 IF INKEY$="T" THEN BEEP1,7
120 IF INKEY$="6" THEN BEEP1,8
130 IF INKEY$="Y" THEN BEEP1,9
140 IF INKEY$="7" THEN BEEP1,10
150 IF INKEY$="U" THEN BEEP1,11
160 IF INKEY$="I" THEN BEEP1,12
170 GOTO 40
```

Many improvements can be made to a program like this in order to create a more efficient keyboard 'instrument'. Generally, the Spectrum is of little use as a sound source because of its lack of facilities for reasonable sound generation. It does, however, make a very useful aid to teaching music. The only way that acceptable sound can be obtained is by purchasing additional sound generating hardware that is compatible with the machine.

circles, and numerous other symbols that can be put together on the screen to draw graphs, make up tables, produce large lettering and create other effects. Each character can be displayed inverted (black on white instead of white on black), further increasing the possibilities. With patience and imagination, high-quality displays can be produced.

Characters can be made to appear on the screen either by use of the PRINT statement or by POKEing the required codes into the Vic-20's screen and colour memories. Commodore's version of the PRINT statement is extremely powerful, allowing the user to define the colour of each character individually. Cursor movement can also be controlled from within the PRINT statement to produce moving displays with ease. POKEing characters to the screen is not as fast as PRINTing them but this method can be useful in certain circumstances.

High Resolution Capabilities

High resolution graphics are possible on an unexpanded Vic-20, but there is enough memory available to utilise only about half the screen. The process by which high resolution can be achieved is known as 'bit-mapping', a technique that enables the programmer to control each individual pixel within a given area of the screen. Each character cell on the grid of 23 rows by 22 columns consists of 64 pixels arranged as eight rows of eight. There is a mathematical relationship between the given co-ordinates of a pixel dot (x,y) and the corresponding bit in the character matrix. This may be used, together with a combination of POKE commands, to produce displays made up of individual pixels.

Super Expander Cartridge

Producing high resolution graphics by bit-mapping can be a long and difficult process. An alternative approach is to buy Commodore's Super Expander cartridge, which gives the user high resolution, colour and music commands in BASIC. High resolution commands include GRAPHIC to set the display mode, POINT to plot a pixel dot on the screen, and PAINT.

There are two main drawbacks to using the cartridge: there is no UNPOINT command to rub out the pixel dot, and it is not possible to PAINT both sides of a diagonal line without disturbing the line itself.

The Vic-20's low resolution graphics are well designed and very flexible, but high resolution graphics are difficult to use without incurring the expense of a plug-in cartridge.

```
100 REM ** VIC 20 RABBIT **
110 POKE36879,110
120 PRINT"
130 PRINT"
140 PRINT"
150 PRINT"
160 PRINT"
170 PRINT"
180 PRINT"
190 PRINT"
200 PRINT"
210 PRINT"
220 PRINT"
230 PRINT"
240 PRINT"
250 PRINT"
260 PRINT"
270 PRINT"
280 PRINT"
290 PRINT"
300 PRINT"
310 PRINT"
320 GOTO320
```

Run Rabbit

This program listing for the Vic-20 demonstrates the versatility of the special set of Commodore characters. The rabbit figure created is constructed entirely from this predefined character set. It is easier to see exactly how it is made up by moving the cursor over the screen display when the program has been typed in and run

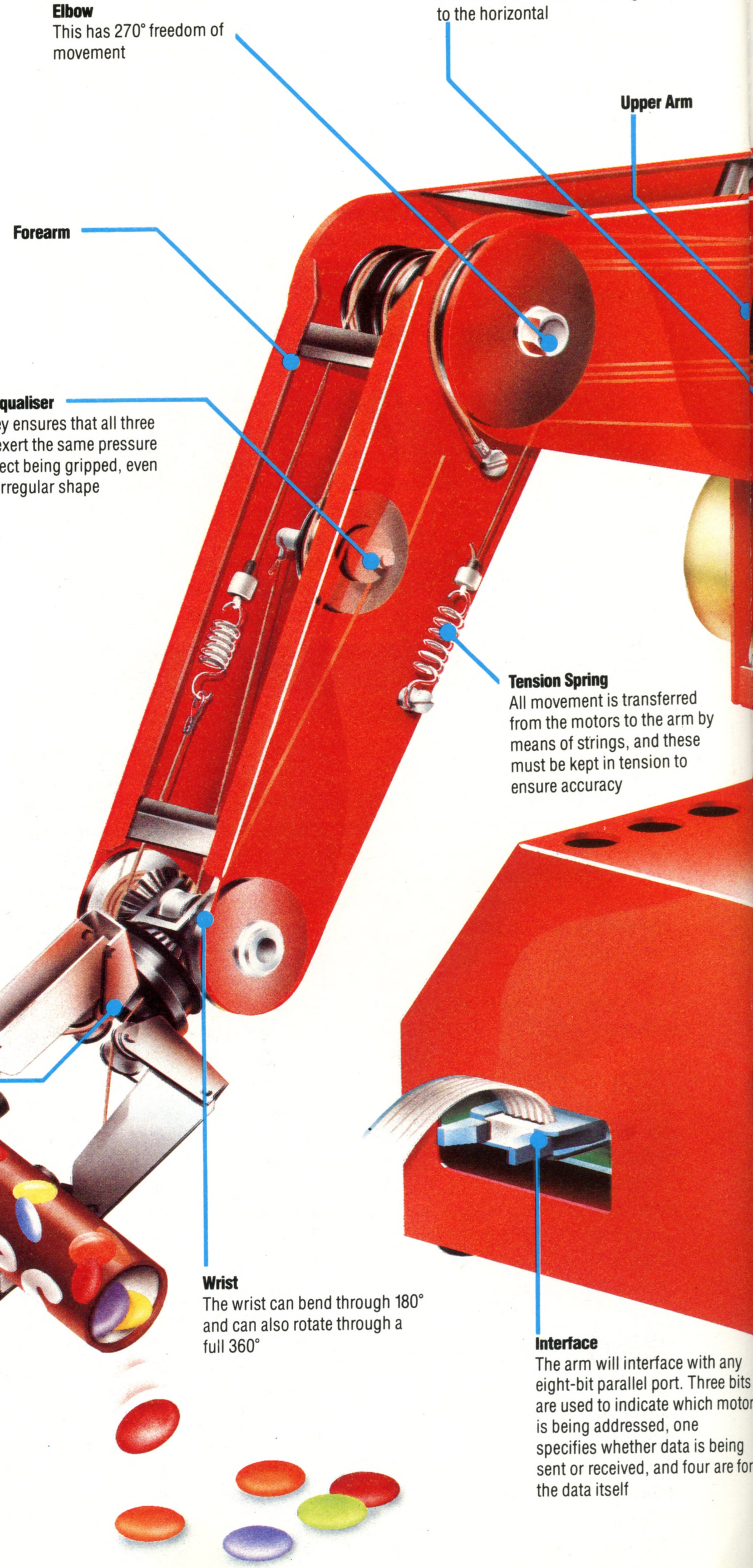
One-Armed Bandits

Small robot arms can provide an insight into control programming, and they can be interfaced to any home computer with a parallel port

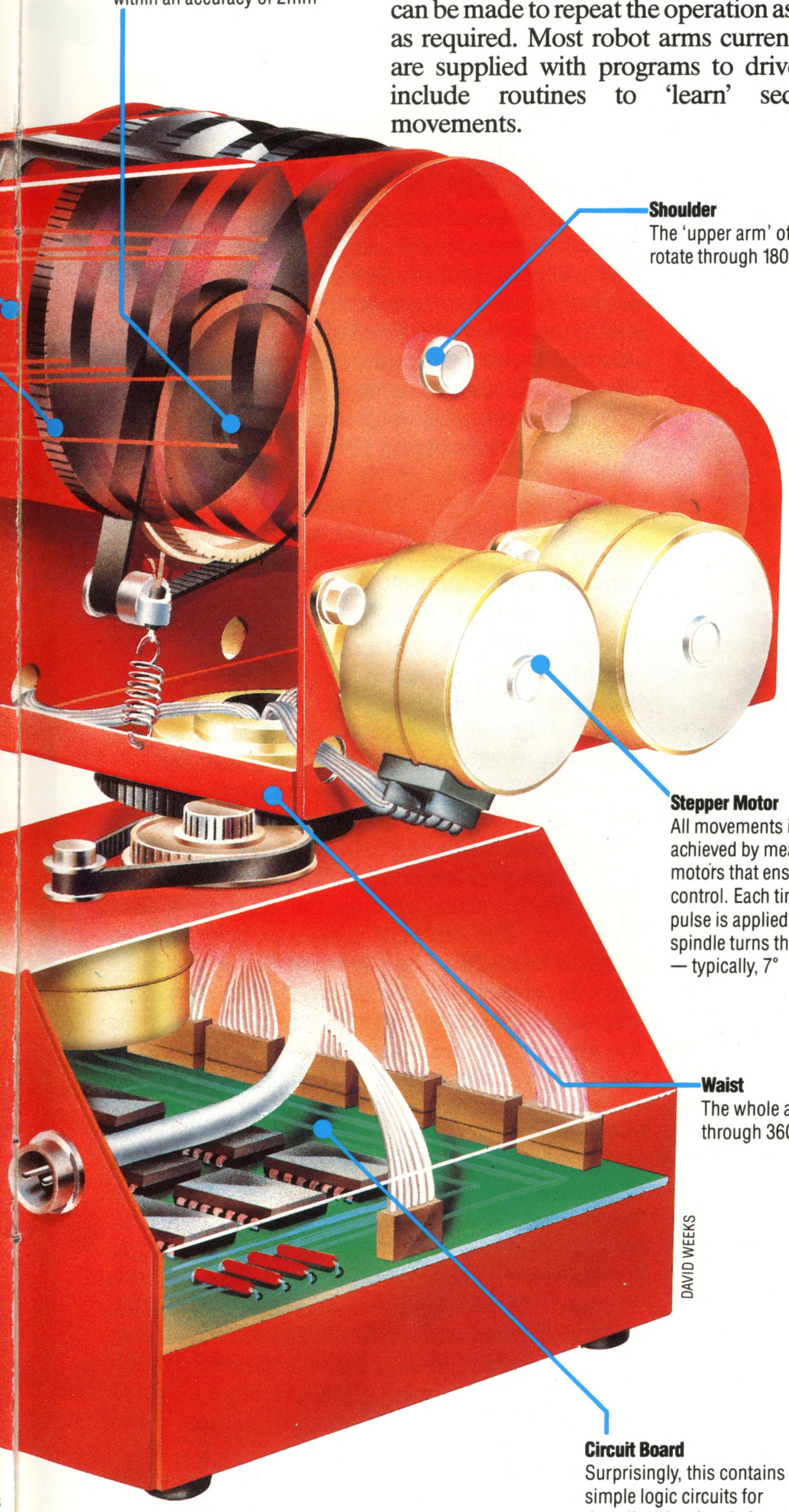
Have you ever wished there was some way that your computer could perform a simple task like make a cup of tea? There is no problem, given the correct interface, in programming a computer to switch the kettle on and off. But when it comes to physically manipulating objects, like tipping a kettle to pour hot water into a teapot, then a mechanical arm is needed. Recently, such devices — called robot arms — have become available for home computer users. These are smaller versions of the industrial arms used by companies like British Leyland and Fiat for welding and painting work on their car assembly lines. The Colne Robotics 'Armdroid', which was probably the first robot arm suitable for use with a home computer, first appeared in 1981. Although the arm is not mobile (unless you were to mount it on a floor robot), it does allow objects to be manipulated with a remarkable degree of precision.

The main components of the robot arm, apart from the metal sections themselves, are the stepper motors that facilitate movement of the sections by precise amounts. There are six motors: one to rotate the arm at the 'waist', one each to control the 'shoulder' and 'elbow' joints, and three to control movement in the 'hand'. All these motors can be controlled very simply by a computer.

All that is needed to interface the arm to a computer is a single eight-bit parallel port. One bit determines whether information is passed to or from the robot. Three address bits are used to select the desired motor, and the other four bits control the direction and speed of movement. Clock signals are also sent to synchronise the movements of the robot arm with the computer's instructions. To speed things up and enable the arm to perform more complex manoeuvres, electronic latches are built into the circuitry that allow any combination of motors to operate simultaneously by 'holding' the instruction to one motor while the other motors are being instructed.



In order to make the arm move into position and grip an object, it is first necessary to divide the overall movement into a set of simple steps. Each motor will need to be instructed separately in a precise movement that will together compose the total motion of the robot arm. This information is then stored in the computer's memory and the arm can be made to repeat the operation as many times as required. Most robot arms currently available are supplied with programs to drive them that include routines to 'learn' sequences of movements.



Gearing Mechanism
Toothed rubber drive-belts and large cog wheels provide a geared reduction so that the arm can be positioned repeatedly to within an accuracy of 2mm

Shoulder
The 'upper arm' of the robot can rotate through 180°

Stepper Motor
All movements in the arm are achieved by means of stepper motors that ensure precise control. Each time an electrical pulse is applied, the motor's spindle turns through one step — typically, 7°

Waist
The whole arm can rotate through 360°

Circuit Board
Surprisingly, this contains only simple logic circuits for decoding the signals from the computer. There is no microprocessor, ROM or RAM

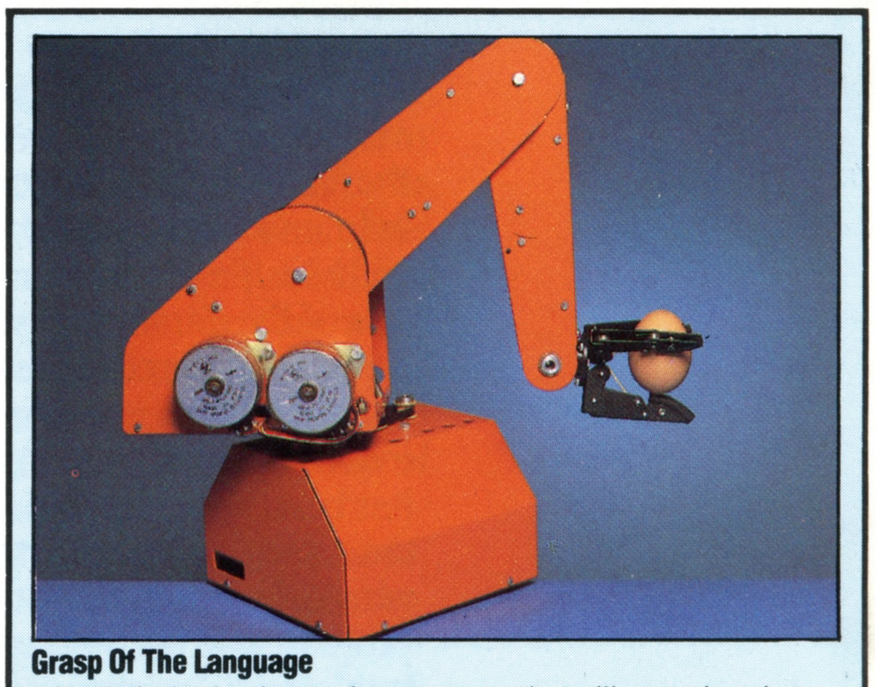
DAVID WEEKS

If the arm is handling delicate objects — the normal testpiece is an egg — the computer must be made to monitor the pressure of the grip. If it is too light the egg will fall; if it is too tight the shell will be broken. Various methods are used to convey information from the arm to the computer, but the most common involve simple microswitches. These can be fitted to set the limits of travel of the arm (most low-cost arms don't include sensors), or they can be built into the grip to detect a pre-set pressure limit.

The main alternative system to microswitches, used on most of the bigger arms, is based on pressure sensing. Certain materials alter their electrical resistance when subjected to change of pressure and these fluctuations can be measured. Although this method is more expensive it does provide very accurate results.

If the program allows no feedback of information from the arm to the computer, it is known as 'open loop', or deterministic. In our example above, such a program would undoubtedly result in a broken egg. If there is, however, some form of feedback that adjusts the actions carried out under the program, then the system becomes 'closed loop', or stochastic. Here the microswitches or pressure sensors are used to limit the closing of the grip at a point where the egg is firmly gripped but not crushed.

Many of the more sophisticated robot systems include multiple sensors to measure light, heat and other variables. These sensors can be used to keep track of what is happening while the arm performs its task and report back if something is going wrong: a robot welder happily burning holes in itself, for example!



IAN McKINNELL

Grasp Of The Language
It is relatively simple to write a program that will control a robot arm. In BASIC the main task would be to enable your computer to accept control commands from the keyboard and pass these to the arm through the port using POKE. Similarly, input from the arm could be read from the associated port by the PEEK function. If speed is required above all else then machine code programming is essential. FORTH is a language that offers the programming ease of BASIC and most of the speed associated with machine code. This language is becoming available for an increasing number of home computers. Sections of a program, rather like subroutines or procedures, are given names that can be incorporated into the command set of the language. This makes it highly efficient for specialised applications such as robot arm control programs. A complete gripping operation, for example, could be controlled by the single command GRIP

Expanding Files

Having established an overall structure, we now continue our programming project with a look at file handling

The address book program that we have been developing in recent instalments of the Basic Programming course is actually a type of simple database, and as such involves the concept of 'files'. The word is used in a number of related, but slightly different, ways. We shall start by discussing these in a little more detail so that we can subsequently use the word with more precision.

In computer programming, a 'file' can be thought of in much the same way as a file in a filing cabinet. It is a collection of related pieces of information stored together. Computers store files on magnetic tapes or disks. Each 'file' of information is given a unique name so that the computer can gain access to it whenever necessary. The information on a cassette tape or a floppy disk may be a program, or it may be 'data' used by a program. Taking the computerised address book as an example, the information needed consists of two separate parts: the program itself, and the data that the program works on. The program is the set of instructions that allows the computer (and the user) to manipulate and work on the data.

The data used by the program is the set of records containing the information you would expect to find in an address book — names, addresses and so on. It also includes certain types of data not normally available to the user. This is the 'housekeeping' data used by the program to help it work. Examples of this type of data might include: 'flags'; information relating to the current size of the database (i.e. the number of records in it); whether or not a sort has been conducted since a new record was last inserted; or possibly an indication of how many times a particular record has been accessed or printed out. The reason why data such as this — and the data comprising the records — needs to be treated separately from the program will become apparent as soon as we try to implement the program.

In earlier parts of the Basic Programming course we have used the READ and DATA statements as ways of putting data to work within a program. This is suitable only if the data is not subject to change, such as the number of days in a month. If the data is liable to change, the program can prompt for it on the screen, and INPUT, INKEY$ or other methods can be used to convey the data to the program. An example of the appropriate use of this form of data input might be a numbers guessing game, in which part of the program might take the form of:

```
PRINT "GUESS THE NUMBER"
INPUT N
IF N < > COMPNUM THEN ...
```

The data in the address book program, however, is subject to considerable modification. In theory, all the records could be stored within the program and read into appropriate arrays using READ and DATA statements. But then all the data comprising the records would have to be entered as part of the program. Whenever changes were made — names and addresses added or removed, for example — considerable alterations would need to be made to the program itself. At the very least, this would involve printing out the program, checking to see where the changes were needed, writing new segments of the program and then typing them in. The biggest problem, however, would be that the new program segments would not be complete program modules that could be independently tested — the changes would be scattered haphazardly throughout the program. The only way of knowing that the modified program worked properly would be to run it and see.

Fortunately, none of this is necessary because data can be stored independently of the program. This is done by creating data files on the cassette or disk. These files are collections of records treated in much the same way as the data in a DATA statement. The program is able to 'open' one or more of these files, read the data from it (usually into an array) and then 'close' the file. If an alteration to the data is needed, the program opens the appropriate file, reads in the data, modifies it, and then writes the modified data back to the file.

With disk-based computer systems, locating a particular file and reading from it or writing to it is quite fast — the location of the file takes only a fraction of a second and the read or write operations usually take a few seconds at most. A cassette-based computer system, on the other hand, may be quite a lot slower and may involve the user in rewinding the tape and waiting for the tape to play through until the right file has been found. Another advantage of using disks is that it is possible to have more than one file 'open' at a time, whereas this is not practical with cassette-based systems.

Files, then, are collections of data stored on a bulk storage medium that are available to be used by one or more programs. A word processing program, for example, might want access to the same set of names and addresses for 'personalised' automatic letter writing.

Files are handled in different ways according to the version of BASIC used. The best way to find out how your computer deals with them is to see what the manual has to say about the OPEN and CLOSE statements and try out a few examples. The description we are presenting here is very generalised and is designed to give an overall impression of using files.

Files may be either sequential or random. In a serial file, the information is stored with the first piece of information first, followed by the next piece, followed by the third and so on. A random file is organised so that the computer can go directly to the piece of data required, without having to start at the beginning and go through the data until the required piece has been located. Watching a film is more like a serial file; you start at the beginning and watch it all the way through to the end. Watching a film on a video recorder at home is a little bit like a random file; you can wind the tape back and forth and watch any part you choose. We shall consider only sequential files, because they're more appropriate to cassette systems.

Suppose you want to keep a record of average daily temperatures for a week. These might be:

MONDAY	13.6
TUESDAY	9.6
WEDNESDAY	11.4
THURSDAY	10.6
FRIDAY	11.5
SATURDAY	11.1
SUNDAY	10.9

To keep things simple, all this data will be treated as numeric data, with Monday being Day 1 and Sunday being Day 7. The data can then be represented like this:

```
1,13.6,2,9.6,3,11.4,4,10.6,5,11.5,6,11.1,7,10.9
```

To store this data in a sequential file, the following steps will be needed in the program:

```
OPEN the file
Write the data to the file
CLOSE the file
```

Whenever the OPEN statement is used it is necessary to state whether we are writing data from the computer to the file (an output) or reading data from the file into the computer (an input). In the BBC Micro, this is done using the OPENOUT and OPENIN statements. The equivalent in Microsoft BASIC is OPEN "O" and OPEN "I". A short program fragment to write the data above into a file (in Microsoft BASIC) would be:

```
100 OPEN "O", #1, "TEMP.DAT"
110 PRINT #1,1,13.6,2,9.6,3,11.4,4,10.6,5,
    11.5,6,11.1,7,10.9
120 CLOSE #1
```

The word OPEN in line 100 makes the file available to the program. OPEN is followed by "O" to indicate that data will go out from the program to be stored in the file. This is followed by #1, which tells the computer that we'll be referring to this as file number 1 in our program. Each file is given an arbitrary number that will subsequently be used with the INPUT# or PRINT# statements when we want to read or write data to that file. Finally, we have the filename in double quotation marks. We've called our file TEMP.DAT to indicate that it contains temperature readings, and is a data file rather than a program.

A complete Microsoft BASIC program to enter the data into a file and subsequently read it out and print it is given below:

```
100 OPEN "O", #1, "TEMP.DAT"
110 PRINT# 1,1,13.6,2,9.6,3,11.4,4,10.6,5,11.5,6,
    11.1,7,10.9
120 CLOSE #1
130 REM LINES 130 & 140 ARE 'DUMMY' LINES TO
140 REM REPRESENT INTERVENING PROGRAM
150 OPEN "I", #1, "TEMP.DAT"
160 FOR X = 1 TO 7
170 INPUT #1, DAY, TEMP
180 PRINT "DAY ";DAY,TEMP
190 NEXT X
200 CLOSE #1
210 END
```

This opens a file, numbered #1 and named TEMP.DAT, writes data into it using the PRINT# statement and then CLOSEs the file. Later in the program the same file is opened using both the number and the filename (the number does not need to be the same as when the file was created, but the number used in the PRINT# or INPUT# statements must be the same as the one assigned to the filename when the file was opened). INPUT #1 in line 170 indicates that the input will come from a file numbered #1 (that is, the file TEMP. DAT) and not from the keyboard.

We shall leave this look at file handling for the moment and return to the address book program and some of the components involved in the INITIALISE subsection of the program. First, let's look at the amount of memory space required for a single record in the address book file (the word 'file' here is being used in the database sense of being the set of all related records, not in the operating system sense of being a named group of data stored on tape or disk).

The use of fixed-length fields is somewhat wasteful of memory space, but makes the programming a lot simpler. If we allow one whole line for each field, with 40 characters to a line, all of this will be saved in an array even if most of the line consists of blank spaces. In some versions of BASIC, however, when string arrays are DIMensioned, each element can be up to 256 characters long. The dimensioning merely sets the number of elements in the array, not the size of each element.

If you have a BASIC that can handle multi-dimensional arrays it would be possible to use a separate dimension for each of the fields, but many versions of BASIC cannot do this so we shall explore alternative approaches. The simplest method is to use a separate string array for each of

the fields. But here's a way to 'cheat' if you want to use multi-dimensional arrays and your BASIC cannot handle them.

The trick is to treat all the elements that would be in the multi-dimensional array as though they were elements in a one-dimensional array. For example, a two-dimensional array with three rows and five columns would be dimensioned like this: DIM A(3,5) and would contain a total of 15 elements: A(1,1) to A(3,5). The same information could be held in an ordinary subscripted array like this: DIM A(15). Everywhere that a two-dimensional array referred to A(R,C), we would substitute A((R–1)*5+C).

If we use a separate string array for each field, we have to decide how to DIMension the arrays. The simplest way is to use a fixed array size, but this limits the total number of records we can store in the database. A better approach would be to set the array size according to how many records there are in use. Not all dialects of BASIC, however, allow string arrays to be as big as you would like. Even if they did, a large number of records in the database could soon use up all the available memory in the computer. Here is a program that will enable you to find out the maximum number of elements your computer will allow. Many versions of BASIC, however, will allow as many elements in an array as you want, right up to the point where all available memory is used up. Each time the program asks you 'WHAT ARRAY SIZE?' enter a larger value until you eventually get an error message. The CLEAR in line 100 has the effect of eliminating the array at the end of each pass. Without this statement, you would get an error message in line 30 through attempting to re-dimension an array.

```
10 READ D$
20 INPUT "WHAT ARRAY SIZE";A
30 DIM N$(A)
40 FOR L = 1 TO A
50 LET N$(L) = D$
60 NEXT L
70 FOR L = 1 TO A
80 PRINT L, N$(L)
90 NEXT L
100 CLEAR
110 GOTO 10
120 DATA "HOME COMPUTER COURSE"
130 END
```

Even if only 40 characters were allowed in each element, with five fields per record, and if there were 256 elements set aside for each array, the amount of memory required to hold all the data within the main memory becomes prodigious. If one byte is required for each character to be stored, we would need 51,200 bytes (5×40×256 bytes) for the data alone. It is obviously not practical to use up so much main memory on the data, and that's why separate data files are used.

Unfortunately, as we have already suggested, file handling routines can be a little difficult to use. If we wish to avoid using external files, the only alternative is to put the data in a DATA statement so that it is always present in the program. Apart from the strain this imposes on the computer's memory capacity, this technique makes modification to the data extremely tedious and prone to error. Therefore it is preferable to use external data files. Once you have tried a few short programs to write data to and from external files, however, the whole process will become much clearer and easier to understand. By way of illustration, we have chosen two very different machines and versions of BASIC to supplement our short daily temperature program given in Microsoft BASIC. These are for the Sinclair Spectrum and the BBC Micro. Both these versions of BASIC differ considerably from our usual Microsoft BASIC, and readers are referred back to the 'Basic Flavours' boxes in earlier parts of the Basic Programming course for details on some of these differences.

In Spectrum BASIC, OPEN# and CLOSE# are reserved for use with the Microdrive. When cassette storage is used, special versions of the SAVE and LOAD commands are needed. The ordinary SAVE command is used for storing programs and program variables on tape (and, of course, ordinary data in DATA statements). Arrays can be saved on tape using the SAVE-DATA statement. This takes the form:

```
SAVE filename DATA array name()
```

The filename is the name given to the file (TEMP.DAT in the Microsoft program). The array name is simply the name of the string array followed by a pair of closed and empty brackets. To SAVE the daily temperature results, we would first have to create a DIMensioned string array and write the data into it, perhaps using READ-DATA statements. To make the difference between the filename and array name more apparent, we will call the array c$ and the filename will be "TEMPDAT".

```
10 DIM c$(14,4)
20 FOR x=1 TO 14
30 READ c$(x)
40 NEXT x
50 DATA 1,13.6,2,9.6,3,11.4,4,10.6,5,11.5,6,11.1,7,10.9
60 SAVE "TEMPDAT" DATA c$()
70 STOP
80 LOAD "TEMPDAT" DATA c$()
90 FOR L=1 to 14 STEP 2
100 PRINT "DAY";c$(L),c$(L+1)
110 NEXT L
120 STOP
```

Line 60 saves all the data in the string array c$ in a data file with the filename TEMPDAT. The program will then stop at line 70, and you should rewind the tape. The keyword CONT will restart the program. Line 80 reverses the process and stores the data in TEMPDAT in the c$ array.

The BBC Micro has one of the most sophisticated dialects of BASIC available on home computers. It allows structured programming with such advanced features as a REPEAT-UNTIL

construct and 'procedures'. Before a procedure can be used it must first be defined, and we will see how this is done in the version of the program for the BBC below. Notice that BBC BASIC defines the direction of data flow in the 'open' statement, using either OPENOUT or OPENIN:

```
10 DIM C$(2,7)
20 FOR R = 1 TO 7
30    FOR C = 1 TO 2
40       READ C$(C,R)
50    NEXT C
60 NEXT R
70 DATA 1,13.6,2,9.6,3,11.4,4,10.6,5,11.5,6,11.1,7,10.9
80 INPUT "TYPE S TO SAVE DATA",K$
90 IF K$ <> "S" THEN GOTO 80
100 PROCSAVE: CLEAR: DIM C$ (2,7)
110 INPUT "REWIND DATA TAPE THEN TYPE L", K$
120 IF K$ <> "L" THEN GOTO 110
130 PROCLOAD
140 PRINT "DAY    TEMP"
150 FOR R = 1 TO 7
160    FOR C = 1 TO 2
170       PRINT C$(C,R);"          "
180    NEXT C: PRINT
190 NEXT R
200 END
300 DEF PROCSAVE
310 X = OPENOUT ("TEMPDAT")
320 FOR R = 1 TO 7
330    FOR C = 1 TO 2
340       PRINT # X, C$(C,R)
350    NEXT C
360 NEXT R
370 CLOSE #X
380 ENDPROC
400 DEF PROCLOAD
410 X = OPENIN ("TEMPDAT")
420 FOR R = 1 TO 7
430    FOR C = 1 TO 2
440       INPUT # X, C$ (C,R)
450    NEXT C
460 NEXT R
470 CLOSE # X
480 ENDPROC
```

One of the advantages of BBC BASIC is that the file handling statements work equally well for cassette files or disk files, and programs written to run on a cassette can be used with no modification if a disk drive is added later.

So far we have seen how data can be transferred from DATA statements via arrays to data files on tape (or disk) and vice versa. The next step will be to look further at the INITIALISATION process to see exactly how many arrays will be needed, how many elements each will need and at what points in the program data will need to be transferred into and out of them.

Exercise

Write a program with the following components:

```
BEGIN
   INITIALISE
   ENTER DATA
   CHOOSE
      Save data
      Load data
      Exit program
END
```

INITIALISE will initialise any variables and arrays needed by the program. The data will comprise, say, 15 names, entered from the keyboard with prompts on the screen. CHOOSE will provide the user with a menu asking if you want to SAVE DATA?, LOAD DATA? or EXIT PROGRAM? See if you can create a flag in the EXIT PROGRAM? part that will automatically save the data if, and only if, a SAVE has not already been carried out.

Basic Flavours

VARIABLES

The BBC Micro supports long variable names such as FULLNAME$. The Spectrum allows long numeric variable names, but only single letter string variable names. The Dragon 32, Vic-20 and Commodore 64 support long variable names, but only the first two characters are significant. On the Oric-1 variable names must consist of two characters (a letter followed by a letter or numeral), while the Lynx requires single letter names.

OPEN

On the Dragon 32 you must use this format:

```
OPEN"O",#—1, "TEMPDAT"
PRINT#—1,1,13.6,2,9.6,3,11.4, etc.
CLOSE#—1
```

and

```
OPEN"I"#—1,"TEMPDAT"
INPUT#—1,D,T
CLOSE#—1
```

CLOSE

On the Commodore 64 and the Vic-20 use this format:

```
OPEN 1,1,2,"TEMPDAT"
PRINT 1,1:PRINT 1,13.6:PRINT 1,2: etc.
CLOSE 1
```

and

```
OPEN 1,1,0,"TEMPDAT"
INPUT #1,D,T
CLOSE 1
```

PRINT#

LYNX ORIC-1

The Lynx and the Oric-1, in their standard form, do not support cassette files. For the 96K Lynx, see Issue 24.

ZX 81

The ZX81 does not support READ and DATA structures, so use this program:

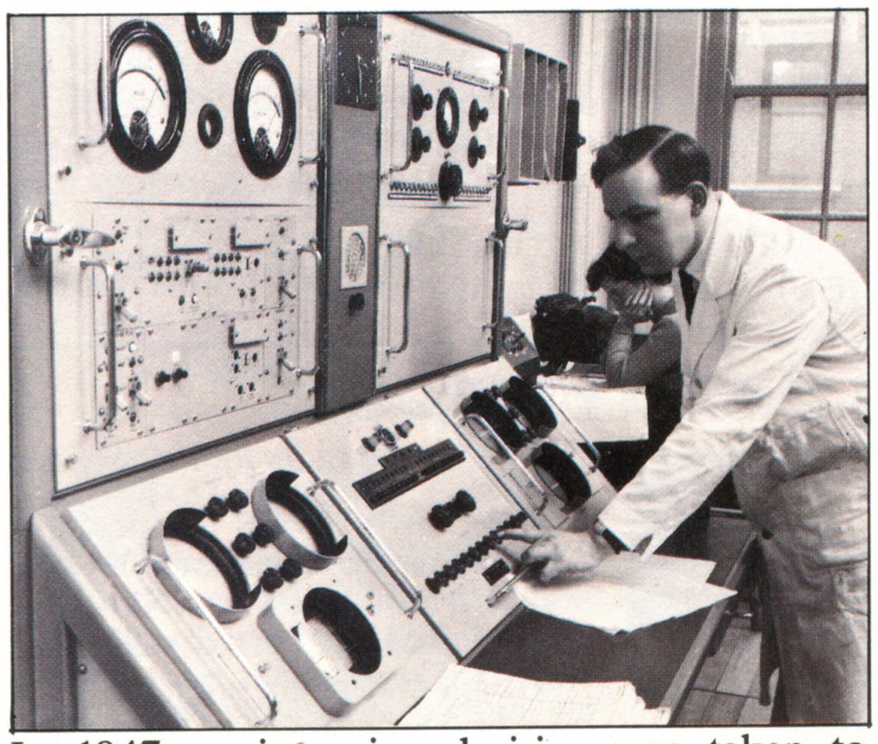
```
10 DIM C$(14,4)
20 FOR X = 1 TO 14
30 PRINT "INPUT ENTRY NO.";X
40 INPUT C$(X)
50 NEXT X
60 STOP
```

Add lines 90-120 from the Spectrum program. When execution stops at line 60, SAVE the program. When you LOAD it again, array C$ will contain the data. Do not type RUN, which would reset the variables to 0; instead,use GOTO 100

TONY LODGE

Lyons' Share

Commercial computing in Britain has its beginnings in a rather unexpected place



Electronic Office
Unlike all previous computers, which were designed for scientific or military applications, the LEO 1 was designed to perform only simple arithmetic operations, but on thousands of items or transactions per day

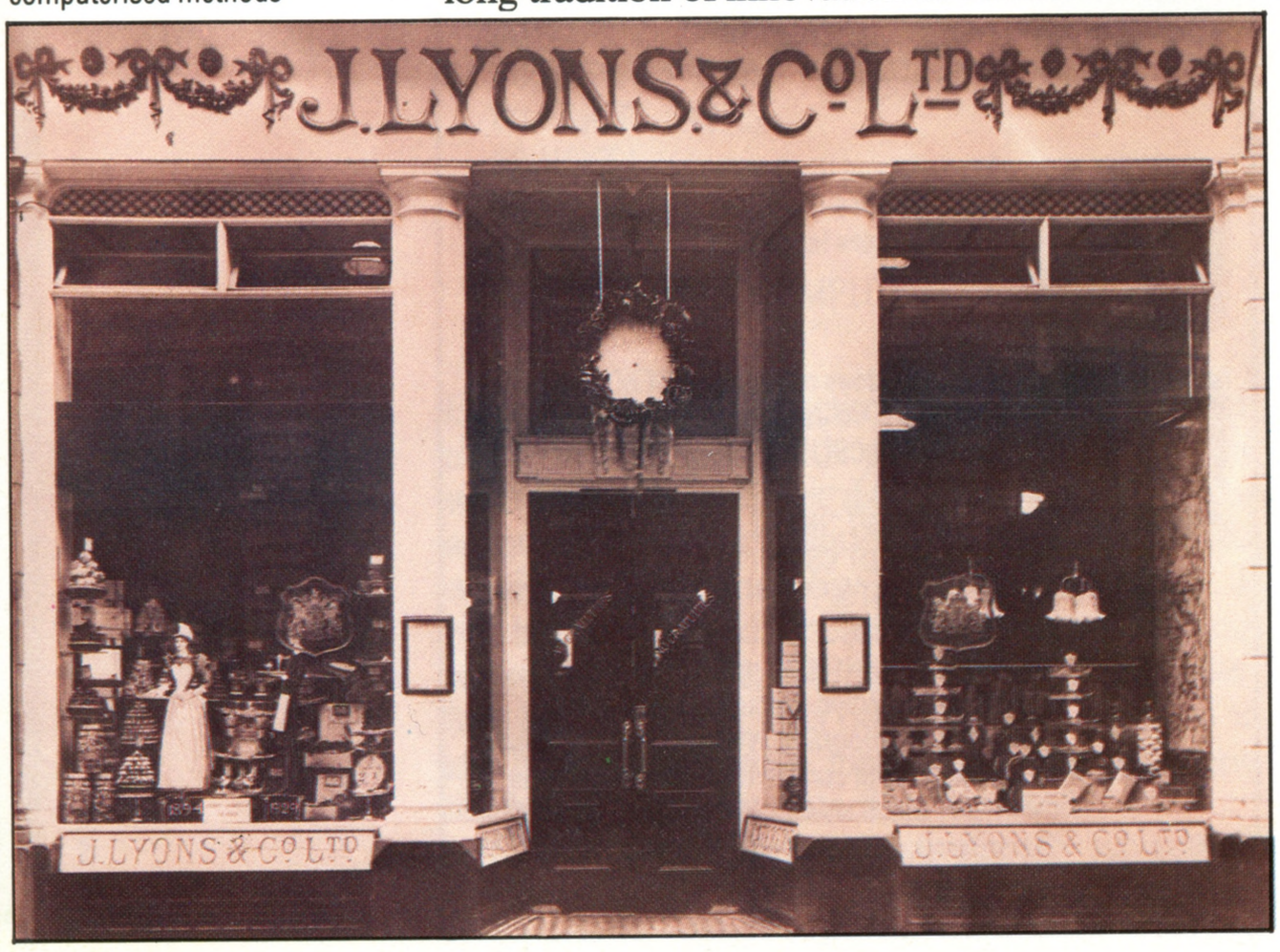
In 1947 a pioneering decision was taken to attempt to build a computer that could be used to automate clerical office work. This was to be the world's first commercial business computer. The imaginative decision came from a rather surprising source: J Lyons, the corner house teashop company. Lyons' business involved a large number of small transactions, and in order to make such a business operation profitable it was necessary to keep the paperwork under strict control. Even after the devastation of the Second World War the company employed more than 1,000 clerks for sorting out the receipts from the teashops.

Because of these considerations Lyons had a long tradition of innovation in business methods: they introduced calculating machines into their teashops as early as 1896 and by the 1930's they were experimenting with microfilm records of transactions. At the same time they set up the first business research centre to study operating methods.

Every few years Lyons would send representatives abroad to investigate new developments that might be useful to them, and in 1947 two employees were sent to the United States to look into the new 'electronic brains'. The most useful discovery they made was that a computer was being built much nearer home, in Cambridge.

Lyons' board of directors commissioned a feasibility study to consider the possible development of the company's own computer. The estimate suggested that a computer could be constructed for £100,000 and that it would save £50,000 a year. Consequently, in October 1947 Lyons began work on the project. The enterprise was all the more daring because at that time the Cambridge computer was only at the design stage. Lyons gave a grant of £3,000 to Cambridge University to help build what became known as EDSAC (Electronic Delay Storage Automatic Computer). The grant was used to buy up government surplus valves. In 1949, EDSAC successfully completed its first job — to calculate a table of prime numbers.

Electronic Teashop
The traditional Lyons teashop seems an unlikely venue for the first major commercial application of computers, but it was precisely this kind of business, with its considerable number of small transactions, which lent itself to computerised methods

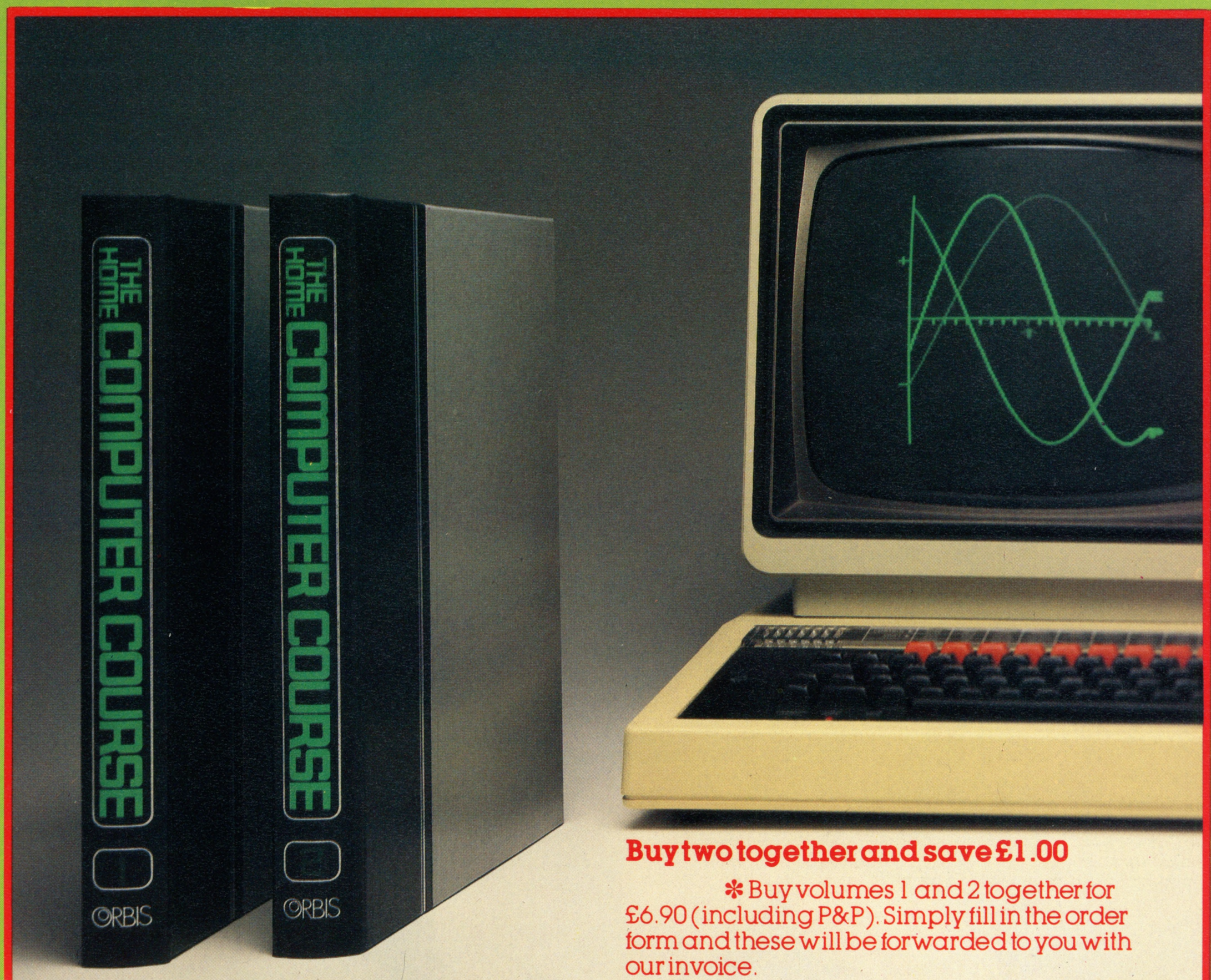


Lyons analysed the problems their computer would have to solve, sketching out the routines that it would need. These studies became the blueprints for the first programs and helped determine the design of the hardware. But it soon became apparent that a business computer was fundamentally different from a university research machine. Whereas EDSAC was designed to execute long and complicated mathematical operations on an input consisting of just a few numbers, the business computer had to solve the opposite kind of problem. Mathematical operations were minimal — just adding and multiplying — but the amount of data processed was enormous.

LEO (the Lyons Electronic Office) became operational on 9 February 1954, and was used to calculate the weekly payroll for 1,700 members of staff. It performed in one and a half seconds an operation that had previously taken a clerk eight minutes.

LEO was a great success for Lyons, and they soon realised they needed more than one machine. Other companies expressed an interest and Lyons set up a company to use the skills they had learnt in manufacturing and marketing computers. Leo Computers was highly successful and went on to produce a series of improved versions of LEO that were used by industry, government and business. The company was bought up in 1963 by the English Electric Company.

THE HOME COMPUTER COURSE BINDER



Now that your collection of Home Computer Course is growing, it makes sound sense to take advantage of this opportunity to order the two specially designed Home Computer Course binders.

The binders have been commissioned to store all the issues in this 24 part series.

At the end of the course the two volume binder set will prove invaluable in converting your copies of this unique series into a permanent work of reference.

Buy two together and save £1.00

✻ Buy volumes 1 and 2 together for £6.90 (including P&P). Simply fill in the order form and these will be forwarded to you with our invoice.

✻ If you prefer to buy the binders separately please send us your cheque/postal order for £3.95 (including P&P). We will send you volume 1 only. Then you may order volume 2 in the same way – when it suits you!

Overseas readers: This binder offer applies to readers in the UK, Eire and Australia only. Readers in Australia should complete the special loose insert in Issue 1 and see additional binder information on the inside front cover. Readers in New Zealand and South Africa and some other countries can obtain their binders **now**. For details please see inside the front cover.

Binders may be subject to import duty and/or local tax.

THE LAST WORD IN LOGIC

Mentathlete

Home computers. Do they send your brain to sleep – or keep your mind on its toes?

At Sinclair, we're in no doubt. To us, a home computer is a mental gym, as important an aid to mental fitness as a set of weights to a body-builder.

Provided, of course, it offers a whole battery of genuine mental challenges.

The Spectrum does just that.

Its education programs turn boring chores into absorbing contests – not learning to spell 'acquiescent', but rescuing a princess from a sorcerer in colour, sound, and movement!

The arcade games would test an all-night arcade freak – they're very fast, very complex, very stimulating.

And the mind-stretchers are truly fiendish. Adventure games that very few people in the world have cracked. Chess to grand master standards. Flight simulation with a cockpit full of instruments operating independently. Genuine 3D computer design.

No other home computer in the world can match the Spectrum challenge – because no other computer has so much software of such outstanding quality to run.

For the Mentathletes of today and tomorrow, the Sinclair Spectrum is gym, apparatus and training schedule, in one neat package. And you can buy one for under £100.

sinclair