ISSN 0265-2919

R 80p

THE HOME COMPUTER COURSE 12

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

Software

Insights

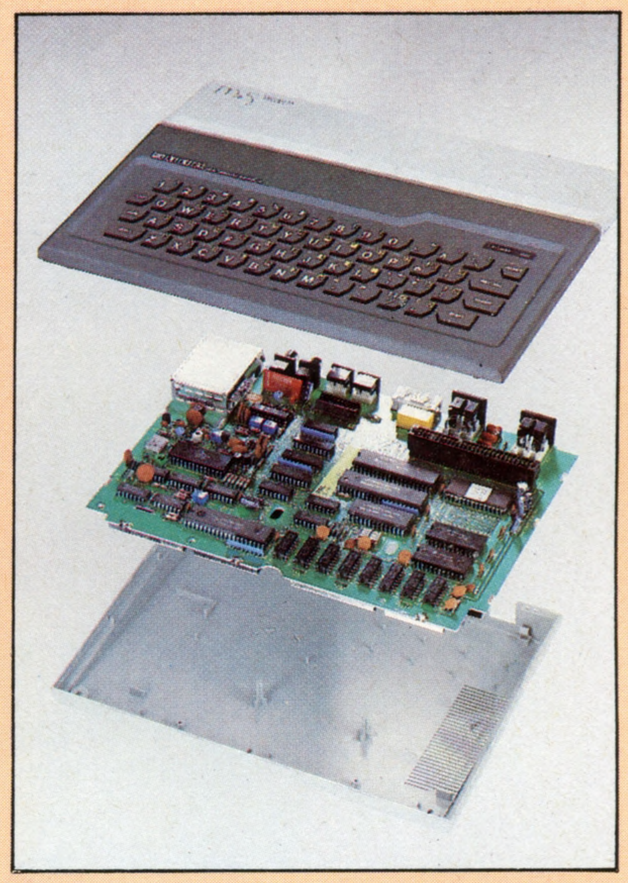
Basic Programming

Passwords To Computing

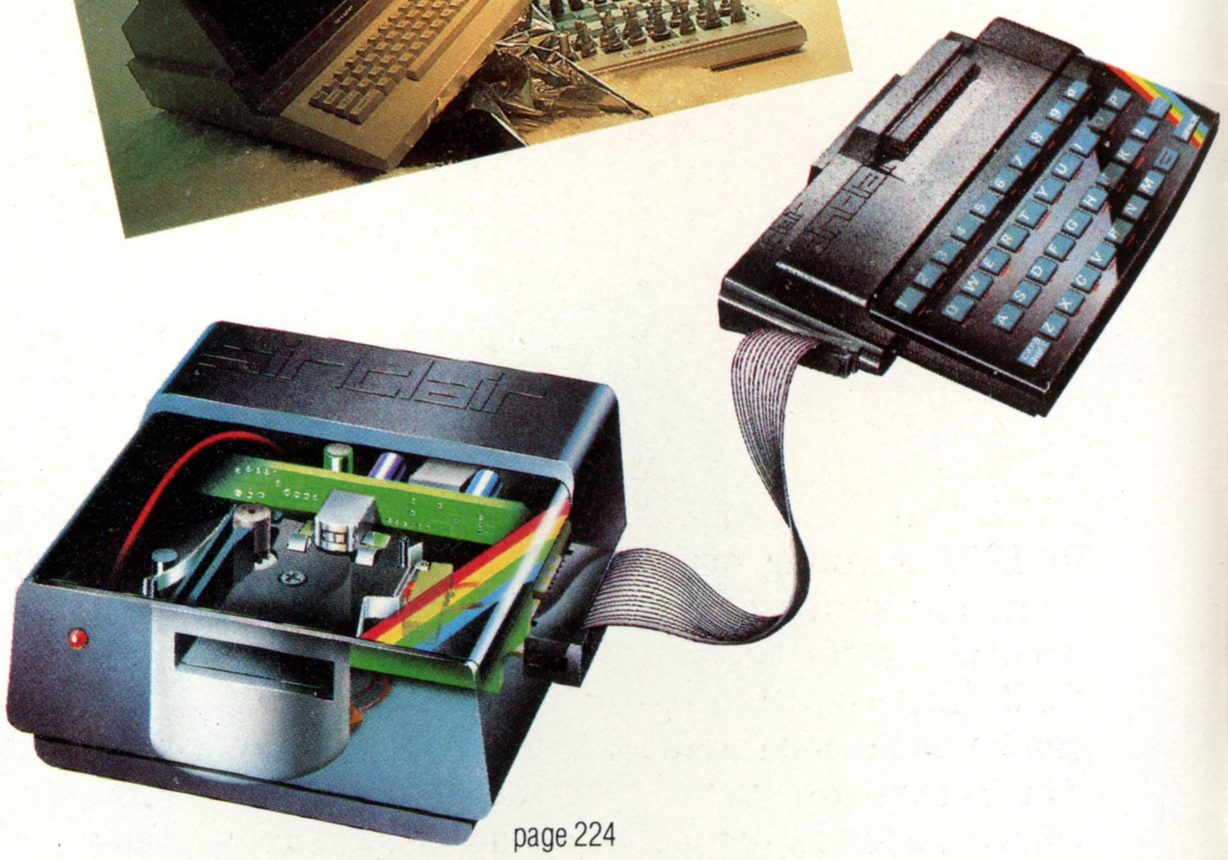
Pioneers In Computing

Next Week

- We look at the Sord M5, the first serious contender for a place in the home computer market to come from Japan
- Cruise missiles are the first generation of weapons that 'think' for themselves. We look at the hardware and software techniques that drive them
- We start a new series that will compare the graphic and sound generating abilities of a wide range of home computers



page 226



page 224

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 Editor** Tracy Ebbetts; **Researcher** Melanie Davis; **Contributors** Tim Heath, Henry Budgett, Brian Morris; **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** 39 Goodge Street, London W1; **© 1983 by Orbis Publishing Ltd: Typeset by Universe; Reproduction by Mullis Morgan Ltd; Printed in Great Britain by Artisan Press Ltd, Leicester**

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.

Man Against Machine

Some of the most powerful microcomputers in use today are to be found not in offices or high-tech factories, but in amusement arcades, cafés and bars

ARCADE COURTESY OF RUFFLER & DEITH LTD
MARCUS WILSON-SMITH

Alien Invaders
'You've got to zap him before he gets to ground zero, or he's going to clone . . . Watch it! The Mothership's popping her Pod . . . you'd better use the Smart Bomb . . .'
No, not dialogue from a sci-fi movie, just conversation overheard in an amuse nent arcade

In 1971 a young man named Nolan Bushnell spent a great deal of time and energy in trying to persuade cafés and bars around Sunnyvale, California, where he lived, to try out a new type of game that he had invented. It was coin-operated, which meant instant money for the subscriber, but involved a totally new concept that we know now as interactive television.

Eventually a bar owner agreed to try it out. Two days later he was on the telephone to Bushnell, complaining that the game had broken down. When Bushnell arrived at the bar, he discovered that the fault was a very simple one — the coin box was full, and the slot jammed. He solved the problem by installing a much larger coin box.

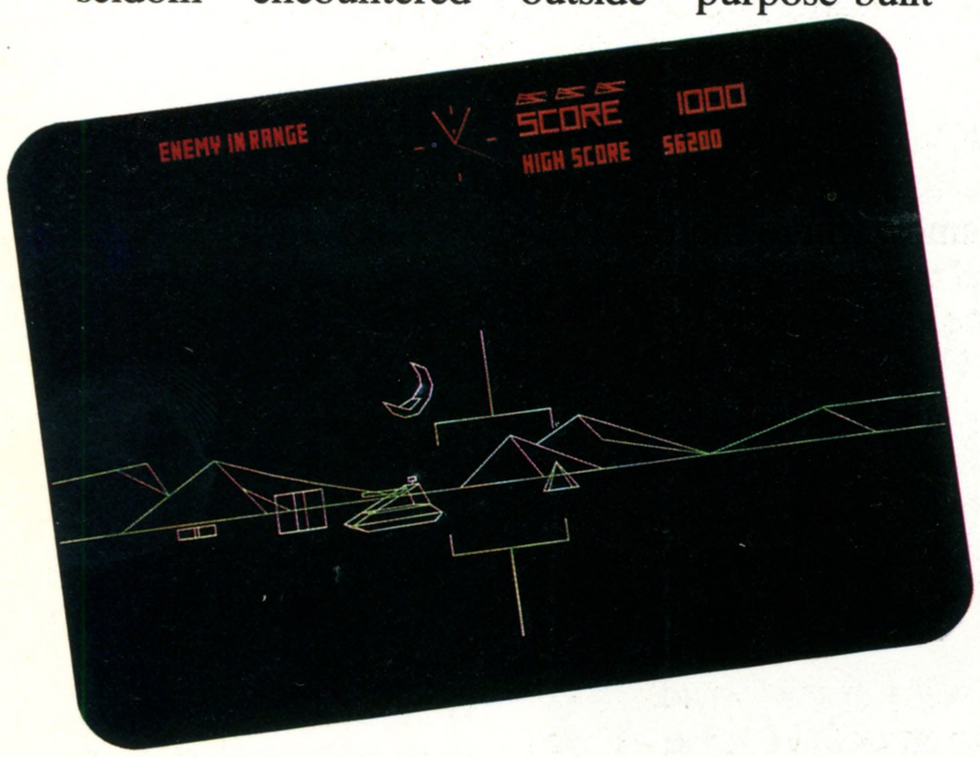
Pong — a derivative of table tennis, or ping-pong — was the precursor of all the exciting and innovative microcomputer-based arcade games that are now to be found all over the world. It is perhaps interesting to note that, while Bushnell was the first to suggest that games like table tennis could be simulated by a computer, he had still not made the essential conceptual breakthrough — his game required two human players to compete against each other, rather than allowing one player on his own to pit his wits and skills against the machine. It took a surprisingly long time for the next generation of arcade games to come along. It wasn't until 1977 that a Japanese company called Taito arrived on the scene with the hugely successful Space Invaders.

In modern terminology, Space Invaders is known as an 'alien zapping with shields' game. The player moves his firing station along the bottom of the screen, dodging behind shields whenever he feels threatened, and firing at an ever-advancing line of stylised aliens, who fire back at him at random intervals. The rhythm of the aliens' advance is totally predictable, and is accompanied by an appropriate two-tone electronic noise that keeps pace with its slow acceleration. When described thus, it sounds really quite simple, but the overall effect is mesmeric and compelling.

The original Space Invaders used a monochrome television display and raster scan

graphics (see page 132). It was hardly innovatory in terms of either hardware or software, but when it first appeared it was the cause of a social revolution the like of which had not been seen since the birth of the cinema. The hardware that supported Space Invaders and all its near relatives differed very little from the home micros of the day. The manufacturers of those home micros were quick to realise that there existed a huge untapped market in the affluent homes of the West. The marketing focus changed rapidly away from learning to write computer programs, towards the use of the computer as an entertainment medium, and machine design followed suit.

Six years later, machines are still being sold on their games-playing power, though by this time the latest generation of arcade games has gone far beyond the abilities of all but the most advanced home computers. It is not uncommon to find up to one million bytes of memory in an arcade game, together with the sort of graphics capabilities seldom encountered outside purpose-built



graphics terminals used with mainframe computers by designers, architects and the like.

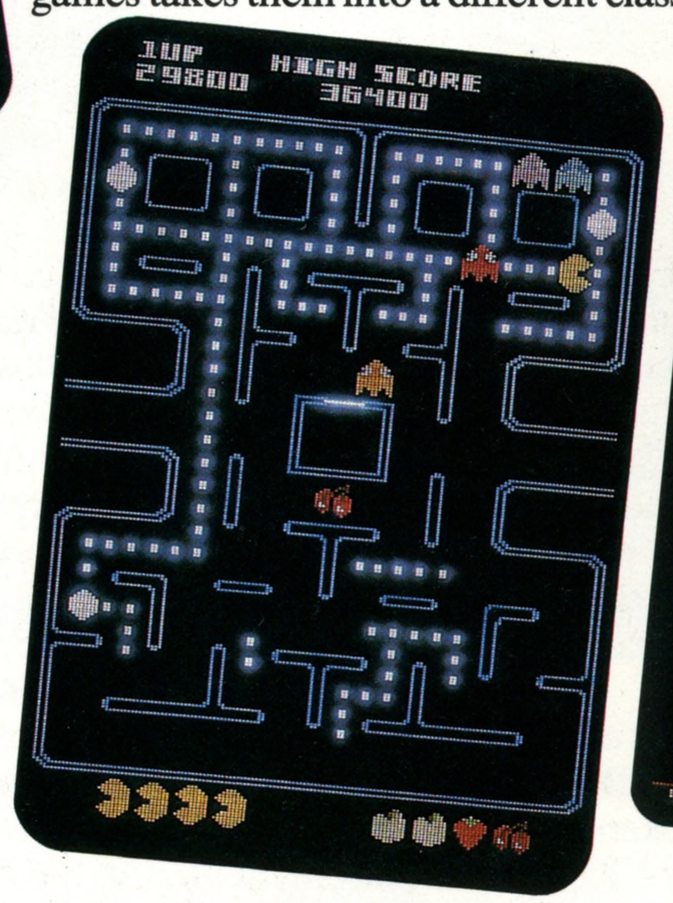
Many of the advances made in the field of home computing can be traced back to arcade games. The move towards 16- and even 32-bit processors, for example, occurred as a result of users' requirements to address more than 64 Kbytes, and their need for faster processing speeds. The manufacturers of arcade games were in the forefront of this movement, and were amongst the first customers for 16-bit microprocessors like Motorola's 68000 and Intel's 8086. Their requirement for faster processing and larger memories predated those of even the users of office microcomputers.

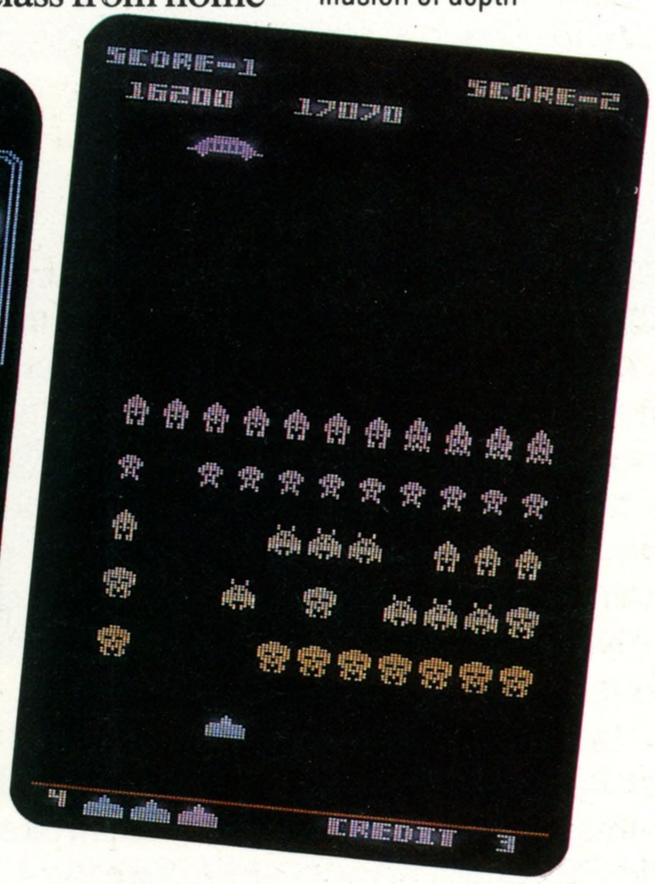
There have been derived benefits for the home user even at the lower end of the scale. Sprite graphics, for example, were developed for arcade games, and later became available for home computers. Dedicated chips for graphics and sound generation, like Commodore's Video Interface Chip and Sound Interface Device, all sprang from that same source, as did the three chips used by Atari for the same purpose in its 400 and 800 series home computers.

Atari, whose fortunes are founded on Nolan Bushnell's original game, is a particularly appropriate example of the trade-off between arcade games and home computers. Atari has long seen the home computer as an entertainment medium before all else (a reflection, perhaps, of its parent's chief interest as Atari is owned by Warner Bros), and in addition to its range of micros, also offers a dedicated games computer known as VCS (Video Cartridge System), which brings many of the games available on arcade machines into the home virtually unchanged. Atari has a great advantage here, of course, being a prime producer of arcade machines itself. However, other leisure- and entertainment-based companies, notably the Columbia Broadcasting System and toy makers Mattel, are also very heavily involved in the same business. They are buying up the rights to produce many arcade games for the home market, under licence, for use on their Colecovision and Intellivision games consoles, respectively.

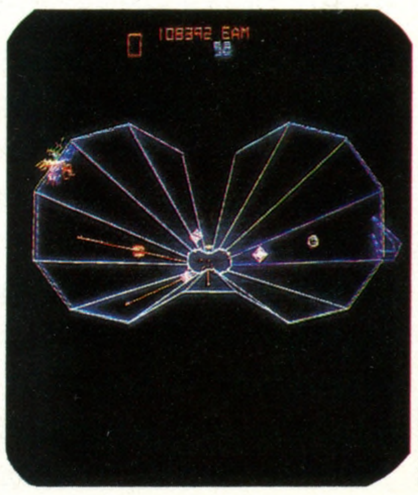
The entire games market, whether for stand-alone arcade games, or for games software for home micros, is fast becoming an industry all on its own. In terms of marketing strategy it closely resembles the record industry, with Top 20 best-selling games listings being produced regularly. The two phenomena are similar in other ways, too. Games software seems to be the thing that schoolchildren swop these days, and there is a thriving software piracy business (see page 192).

We mentioned earlier that the degree of hardware development to be found in arcade games takes them into a different class from home





micros. The most recent of these developments involve the use of video discs to provide a backdrop on the television monitor, against which the games are played. Whether refinements such as this will ever reach the domestic market is a matter for speculation. The technology certainly exists within reasonably priced consumer electronics, though the games software produced for the home market, whether for dedicated games consoles or for home micros, has never really come up to the standard of arcade games in



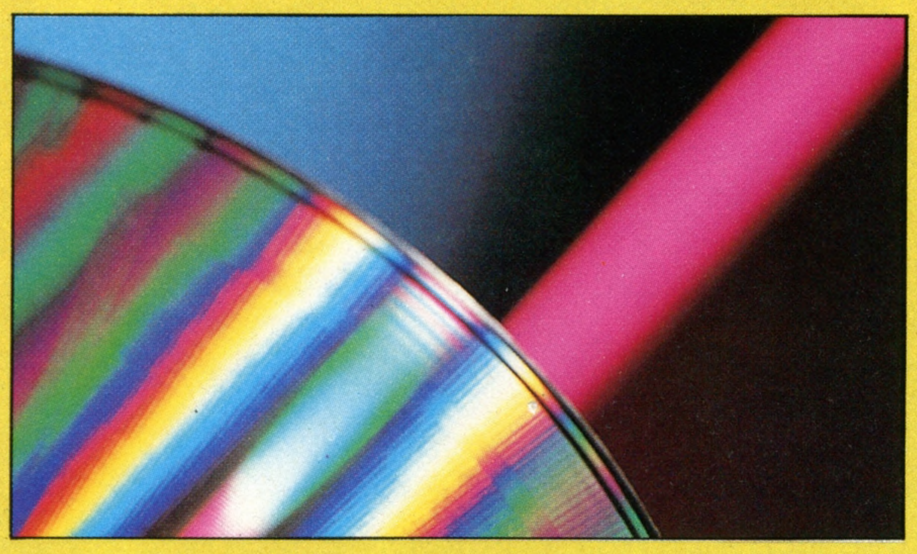
COURTESY OF HUTCHINSON & CO LTD

Playing Fields

Arcade games like Space Invaders, where the player can move his 'token' only along a fixed line, have been overtaken in popularity by 'maze chase' games like PacMan. They both use sprite graphics, and as a result were of poor visual quality. Recently, games designers have been producing much more abstract representations, such as Battlezone, a futuristic player versus tank and missile game, or Tempest, which uses stunning graphic design to produce an outstanding illusion of depth

Space Capsule

Astron Belt is typical of the new generation of arcade games, which use laser discs (see below) to provide a moving background to the game being played. Because discs are random access devices, it is possible to move directly from one scene to another — a confrontation with an alien ship that results in an explosion, for example. To the player, this represents a big step towards realism. The images on the laser disc can either be motion pictures from real life, or computer generated animation. Astron Belt also offers stereo sound and even a vibrating seat that reacts to low frequencies in the audio circuit



TONY LODGE

their original form.

When we discussed flight simulation (see page 201), we noted that all arcade games are simulations of one sort or another, either of a real-life situation, like a game of table tennis or a car race, or a fantasy, such as Space Invaders, PacMan or Frogger. Over the dozen years that arcade games have been in existence, these two broad bases have expanded into parallel streams of development, though the fantasy/space games are much more common.

We looked at the two original games — Pong and Space Invaders — earlier, but it is worthwhile tracing their development. The first of the bat and ball games to set man against machine was Breakout and its variants, in which the player hits a ball against a wall of bricks. Each brick disappears from the screen as it is hit, and the object is to remove them all without losing the ball. Following on from this we have golf, snooker/pool and pinball 'simulations'. Generally speaking the more attention the programmer pays to replicating the forces found in the real world (for example gravity, wind or surface resistance, mis-hitting), the better the game will be.

But these criteria do not apply to fantasy games. Here, the player is actually competing against the person who programmed the machine to play the game in question, and playing entirely by *his* rules, in a universe of *his* creation. The first stage after Space Invaders was to introduce different types of aliens that attack in different ways at random intervals. After this, it was necessary to introduce mobility into the player's 'screen token', which led to the introduction of games like Defender, accepted among the *cognoscenti* as the best of all the games of this type.

Fixed line games, like the original Space Invaders, continued in development, and became games like Missile Command and its derivatives, in which the object is to defend one's base against an attack by Inter-Continental Ballistic Missiles. Maze/chase games such as PacMan evolved from the original car race games, where the objective was to steer a car-shaped token around a course as often as possible in the time allowed, without crashing into the walls. There was no real element of competition in these games — even allowing for the random oil slicks that appeared by magic — and so the next step was to turn the 'time trial' into a chase. The original car track now became a maze, and the tokens changed into fruit, light bulbs and the like.

Car chase games went towards a pseudo three-dimensional representation of the course, viewed from the car, or behind it, with the ever-changing road coming at the player. A very similar method is used in the more realistic of the flight simulation games.

Lastly come the traditional board games, like draughts, chess and backgammon. These are confined to home computer applications, because they typically take a much longer time to play, and the graphic representation takes second place to the games algorithms themselves in the program.

The only real technical departure lies in the method of graphics generation used. All the early games, and most of the current ones, use raster scan graphics, but some, notably Asteroids, use vector scanning methods whereby only the images on the screen — not the dark areas — get scanned by the electron beam.

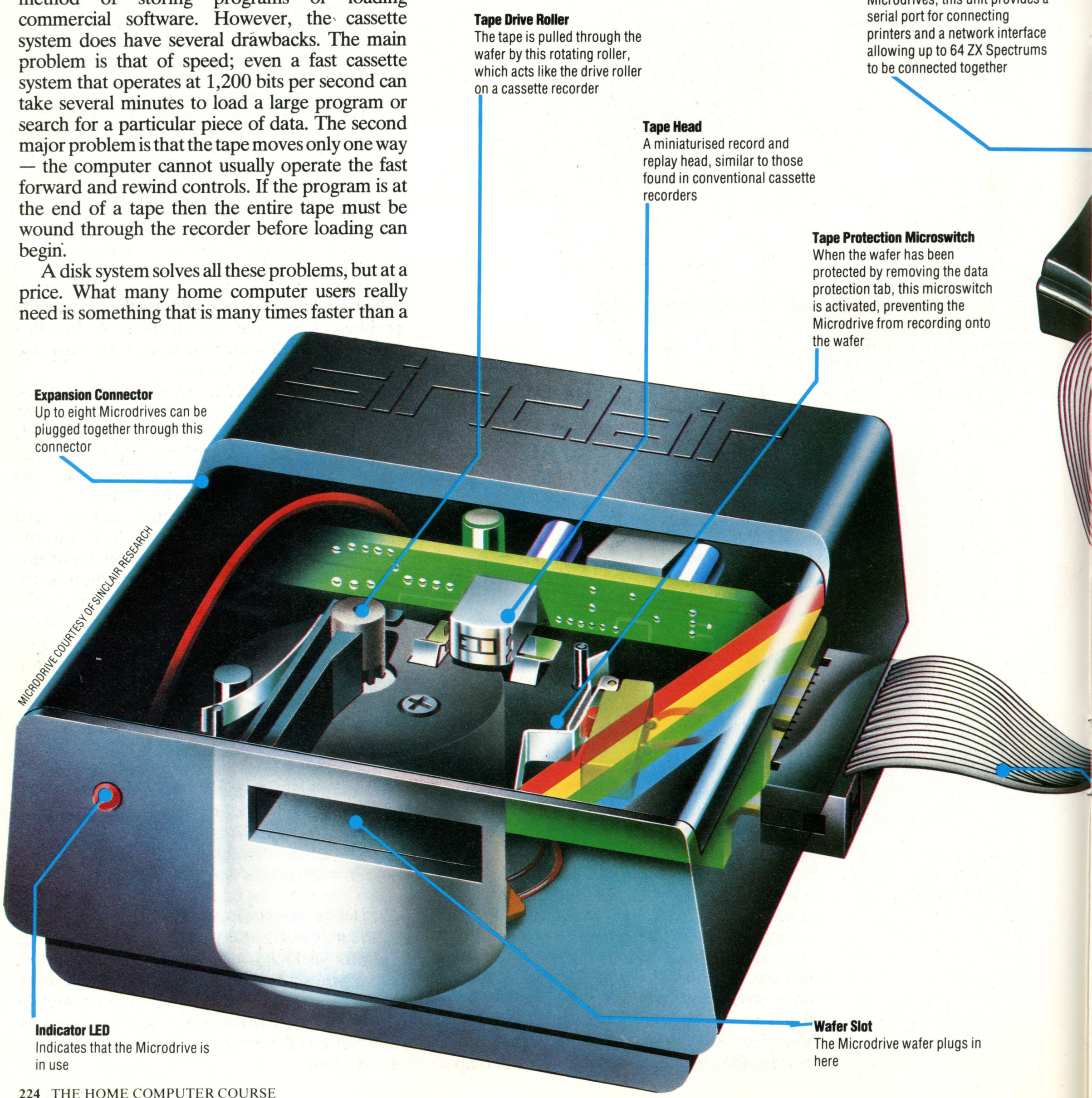
So the next time you pass an amusement arcade crowded with vastly complicated games, or lean against one in a bar, remember that the computer that is inside driving it is probably far more complex and more powerful than any in use in the home or the small office, that it uses many of the same techniques, and is supported by some of the most talented programmers in the world today.

Small Is Beautiful

Sinclair's Microdrive is just one answer to the cost and size problems associated with mass storage. It uses a narrow loop of magnetic tape instead of a floppy disk

For the home computer user the conventional cassette offers a cheap and generally reliable method of storing programs or loading commercial software. However, the cassette system does have several drawbacks. The main problem is that of speed; even a fast cassette system that operates at 1,200 bits per second can take several minutes to load a large program or search for a particular piece of data. The second major problem is that the tape moves only one way — the computer cannot usually operate the fast forward and rewind controls. If the program is at the end of a tape then the entire tape must be wound through the recorder before loading can begin.

A disk system solves all these problems, but at a price. What many home computer users really need is something that is many times faster than a



cassette but much cheaper than a disk. Such devices exist and are known as 'floppy tapes' or 'stringy-floppies'. Originally developed in America for Tandy's TRS-80 Model 1 system by Exactron, the first stringy-floppy used a continuous loop of tape in a cartridge housing; the idea was borrowed from the eight-track audio

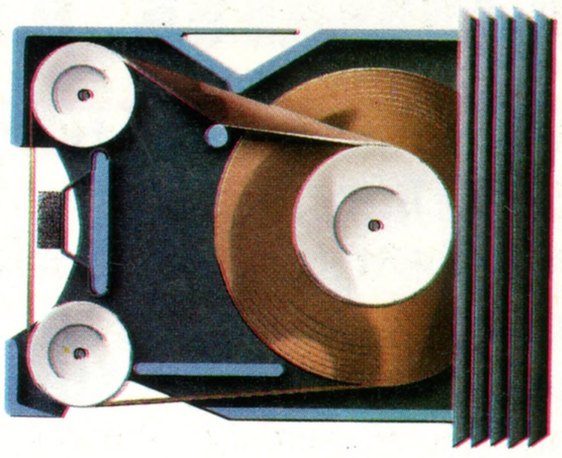
DAVID WEEKS

Interface Lead
This flexible ribbon cable links the first Microdrive to the Interface 1 unit

tape system that was fashionable some years ago. The principle of operation is simple: the tape loop circulates constantly, so the various programs can be found much more quickly. A catalogue of all the programs and files stored on the tape is also kept (just like the directory on a disk), so a list of the contents is always available.

Because the information is recorded digitally rather than by audio methods, the transfer of information can be much faster than with an audio cassette tape — at least five times as fast, and often more. No complicated and expensive interfacing is required: the unit uses a normal parallel port and all the necessary operating software is either built into the drive or comes as a ROM to be plugged into a spare socket inside the computer. The name given to these drives describes their halfway status between a tape and a floppy disk — they use tape but operate like a disk.

Unfortunately, the first units were beset with problems. The mechanisms worked well but the tapes themselves proved unreliable. The major failing was that the tape simply couldn't take the strain of being constantly pulled out of the middle of a coil and wound back on the outside. This was not a problem with the eight-track audio tapes as they were much wider and moved far more slowly. Very few versions of the original stringy-floppy reached the UK; one was produced by a company



The Microdrive Wafer
The tape contained inside the Microdrive wafer is a continuous loop of video tape about 2mm ($^3/_{32}$ ins) wide. Video rather than audio tape is used simply because of its strength and long life. Compared with an audio cassette the tape is both thinner and narrower, making it very easy to damage.

In operation the loop of tape circulates within the Microdrive in approximately seven seconds and information is transferred at around six Kbytes per second, a significant improvement over the 1.5 Kbytes per second of the Spectrum's cassette interface. Any program stored on the wafer can be found and loaded in about 15 seconds. Each tape can hold up to 100 Kbytes of information, but Sinclair guarantees only that 85 Kbytes will be available for use. Each wafer must be prepared for use by 'formatting' the tape, a process activated by a simple BASIC command. Formatting checks which parts of the tape are capable of being used, and skips over any bad patches

called Aculab but it suffered from the tape quality problem. Until the introduction of Sinclair's Microdrive the floppy tape system was regarded as something of a lost cause.

The principles behind the Microdrive are just the same: a loop of tape is constantly rotated past a record and replay head. Furthermore, the tape is smaller than ever, at about 2mm ($^3/_{32}$ins) in width, under half as wide as the original floppy tape. Its reliability, however, is yet to be proven.

One alternative storage system that offers high speed with proven reliability is the digital cassette tape. Professional units have long been available but at high prices. With the introduction of the Philips micro digital cassette system, devices like the Hobbit (see page 94) have started to appear. Although the tape is not in the form of a loop it offers remarkable speed. The directory is kept in the middle of the tape, which can wind in both directions under the control of the operating system. Applications of this type of micro cassette have recently appeared in the Sharp PC-1251 and the Epson HX-20 portable computers.

The only major problem with all three systems of storage is that the software available in their respective formats is limited. The PC-1251 and HX-20 computers have the facility to load programs from a conventional cassette and then save them onto the internal micro cassette. The Hobbit and the Microdrive can be connected to the computer at the same time as a normal cassette recorder, making copying even easier.

In terms of proven reliability the digital micro cassette system has the edge over both stringy-floppy and conventional cassette systems. Whether it will ever replace them is a matter for speculation.

Straight Facts

Which computer do you buy, and how do you improve its performance with accessories and peripherals? Here's an unbiased assessment of the available hardware

Peripherals

When a new computer is announced, the brochure frequently describes a range of peripherals, which may not even be at the design stage. It is therefore important to distinguish between what is currently available and what is merely planned. A machine that has both disk drive and printers available from the manufacturer is one for which useful business software will be developed. For entertainment uses, a well-planned machine will have joysticks, game paddles and light pens available, though in these cases, better value can often be obtained from independent suppliers

> > > Spectrum, BBC Model B, Tandy Color, TI99/4A
> > Atari 400 & 800, Vic-20, Commodore 64, Dragon-32
> Oric-1, Sord M5

Screen

Much attention is given to a computer's graphics, when in fact most of the time you may be viewing text, or program listings. Two factors are important: the number of characters that can be displayed at once (25 rows of 40 characters is an average figure) and the legibility of the characters themselves. Also important are the screen editing facilities. Can alterations be made to a program simply by moving the cursor to the position and typing, or must special commands be used?

> > > BBC Model B, Atari 400 & 800, Commodore 64
> > Oric-1, Sord M5, TI99/4A
> Spectrum, Vic-20, Dragon-32, Tandy Color

Keyboard

Unless you intend only to play games using a joystick, you will be using your keyboard very often. It is therefore important that you are happy with it. Two factors dictate the quality of a keyboard. Firstly there is the physical design or 'feel'. A touch-typist, for example, would probably prefer one of the better quality keyboards as some of the cheaper computers are fitted with solid-state keys, which do not offer 'tactile feedback'. Secondly there is the keyboard layout. A keyboard with a separate key for each function is ideal, as it can be confusing when a key has a number of different possible applications. Programmable function keys are worth having, because they can be used within a program to perform dedicated functions like FIRE or START AGAIN

> > > BBC Model B, Atari 800, Vic-20, Commodore 64
> > Dragon-32, Tandy Color, TI99/ 4A
> Spectrum, Atari 400, Oric-1, Sord M5

Business Software

If your main reason for buying a microcomputer is for running a small business, then you should consider a purpose-designed computer. However, the more expensive home computers can also double-up as business machines provided you are prepared to invest in a disk drive and printer. Applications then include word processing, accounts, databases and spreadsheets. As with games, some machines have a lot of business software available, some none

> > > Atari 800, Commodore 64
> > BBC Model B, Atari 400, Vic-20, TI99/4A
> Spectrum, Dragon-32, Oric-1, Sord M5, Tandy Color

Interfaces

A computer with a large range of interfaces on the back is one that has been designed with future expansion in mind, and this feature is a very good pointer to good design in general. Having standard ports like RS232 and Centronics means that you have a wide choice of devices to hook up to the computer, other than the units supplied by the manufacturer

> > > BBC Model B,Vic-20, Commodore 64
> > Atari 400 & 800, Dragon-32, Oric-1,Sord M5, Tandy Color
> Spectrum, TI99/4A

The Quality Of Basic

Almost all home computers come with a BASIC interpreter built into ROM, but as you will have gathered from our 'Basic Flavours' boxes, the commands available on each machine vary considerably. All BASICs that bear the name Microsoft work in approximately the same way, others may use completely different structures for strings, arrays etc. A good BASIC is one that has lots of friendly 'high level' commands like CIRCLE, DRAW, and PAINT to take advantage of that computer's graphics and sound

> > > BBC Model B, Dragon-32, Spectrum, Atari 400 & 800, Oric-1, Tandy Color, TI99/4A
> > Vic-20, Commodore 64,
> Sord M5

Graphics

The difficulty with comparing the graphics on home computers is that most now offer more than one mode of graphics. Typing MODE 1, for example, might give you access to 16 colours on a grid of 40 × 25 positions, while MODE 7 might give the maximum resolution of 320 × 200 pixels, but with a choice of just two colours. One thing to watch out for is that the maximum resolutions on some machines can only be realised if you use a monitor instead of a television as the screen

> > > BBC Model B, Atari 400 & 800, Commodore 64, Sord M5, TI99/4A
> > Spectrum, Dragon-32, Oric-1, Tandy Color
> Vic-20

Memory

The more RAM a computer has, the more sophisticated the programs it can run in terms of the amount of data they can handle at once. Home computers come with anything from 1K to 64K as standard, though most can be expanded with plug-in modules. However, the computer may well require part of this memory for its own internal use (called the 'system overheads'), leaving less for the program. High-resolution colour graphic displays, in particular, consume large amounts of RAM

> > > Spectrum (48K), BBC Model B, Atari 800, Commodore 64, Dragon-32, Oric-1 (48K)
> > Atari 400, Tandy Color (16K,) TI99/4A
> Vic-20, Sord M5 (4K)

Cassette Recorder

Unless you are prepared to stick exclusively to ROM cartridge or disk-based software, you need a cassette recorder. Only a couple of manufacturers still require you to purchase their own unit — most computers will work with any domestic cassette recorder, though both the speed and the reliability with which programs are saved and retrieved vary considerably. The better systems also permit control over the cassette recorder's motor

> > > Spectrum, BBC Model B
> > Dragon-32, Oric-1, Sord M5, Tandy Color, TI99/4A
> Atari 400 & 800, Vic-20, Commodore 64

Games Software

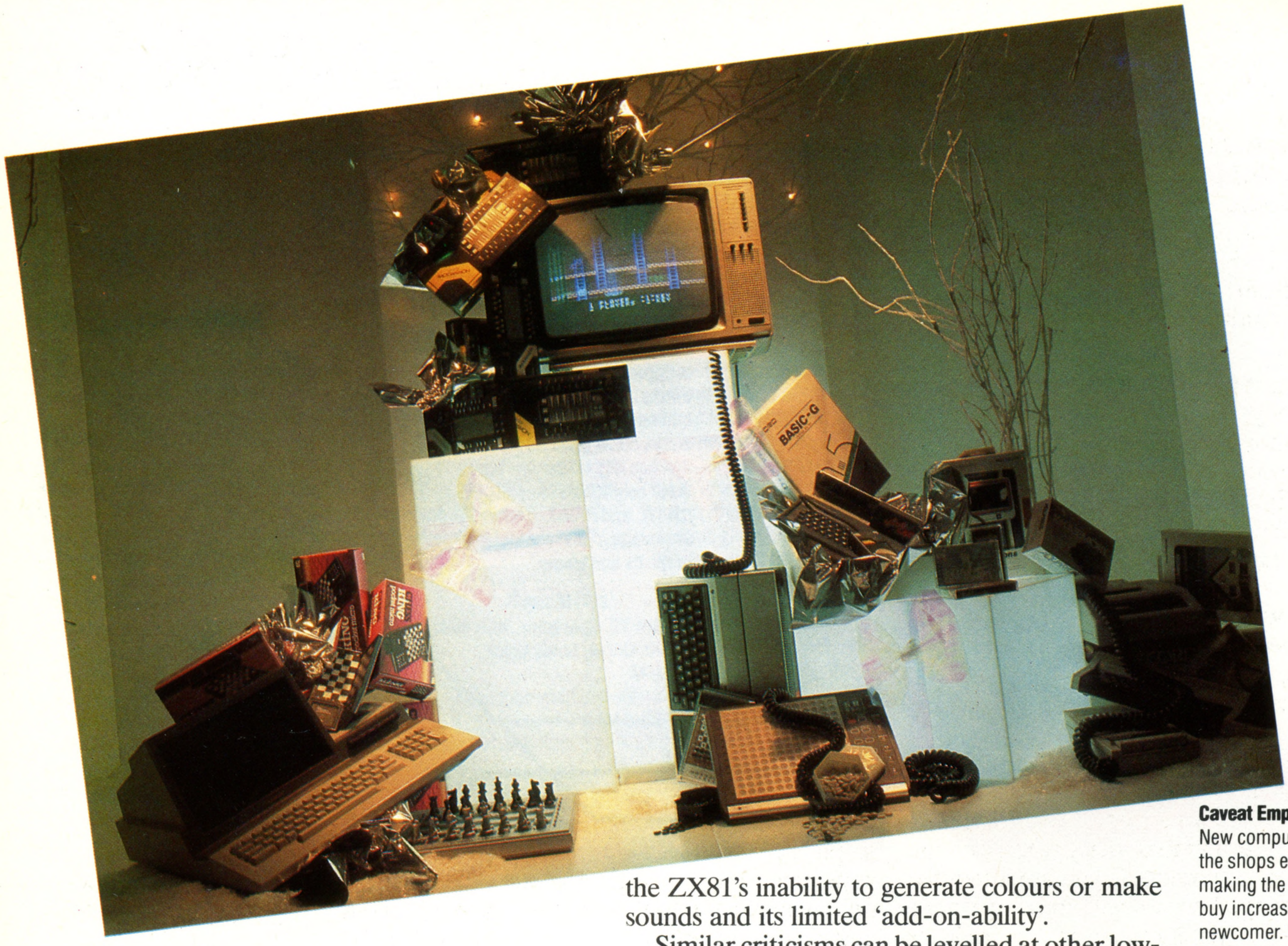
Much of the best games software is available, not from the computer manufacturer, but from specialist software suppliers. What governs both the quantity and quality of games produced for any machine is the number of users of that machine (therefore the size of potential sales) and the computer's technical specifications. Consequently, some computers have a 'games' image, whilst others are geared towards home programming or business use

> > > Spectrum, Atari 400 & 800, Vic-20, TI99/4A
> > BBC Model B, Commodore 64, Dragon-32, Sord M5, Tandy Color
> Oric-1

Sound

Computers that make use of the television speaker can generally produce higher quality sound than those using a built-in speaker, though this is not always true — so ask for a demonstration before you buy. A single-voice system allows only single notes (like playing the piano with one finger), three and four voices permit chords. The 'white noise' generators found on some computers are used for creating explosions and other effects, while those with 'envelope control' can imitate different musical instruments like a mini synthesiser

> > > BBC Model B, Atari 400 & 800, Commodore 64, Oric-1, Sord M5, TI99/4A
> > Vic-20
> Spectrum, Dragon-32



Caveat Emptor
New computers are appearing in the shops every month — making the decision of what to buy increasingly difficult for the newcomer. With so many factors to consider, the old saying 'buyer beware' is more appropriate then ever

Whether you are setting out to buy a home microcomputer for the first time, or you already have one which you now wish to upgrade to a higher level of sophistication, the choice available to you can be quite bewildering. So far in the HOME COMPUTER COURSE we have described in detail some of the home computers on the market, explained basic operating principles and operating methods, talked about some applications of microcomputers to our everyday life and decoded some of the jargon that surrounds them.

There are many factors to be considered when purchasing a microcomputer for the first time, and perhaps the first amongst them is the cost of the machine. Now, it is possible to go to a high street store and buy a Sinclair ZX81 in its simplest form for less than £40. The chief drawback to this is the keyboard quality or 'feel', of the ZX81. In order to reduce the size of its physical package as much as possible, Sinclair adopted a multi-layer membrane design for the keyboard. The result is an absence of sensitivity or 'tactile feedback', which can lead to considerable frustration. Computer programming, when one is a complete beginner, is quite demanding enough, both in terms of intellect and attention to detail, without the added complication of a relatively 'unfriendly' machine.

This might appear a small enough price to pay, considering the saving of £60 or more that one makes, but there are other considerations, such as the ZX81's inability to generate colours or make sounds and its limited 'add-on-ability'.

Similar criticisms can be levelled at other low-cost computers: for example, the Aquarius, which is very competitively priced, but which has a low-quality keyboard of unorthodox layout, and lacks program editing facilities. So we disregard the cheaper machines in this survey.

There is a whole series of features that one should expect in a home micro. These have been detailed on the previous page, and it is vital to make them into a personalised list, in order of importance. If, as many users do, you see your prime requirement as playing computer games, then you will want a broad software base, appropriate peripherals, such as joysticks or trackballs, good graphics and sound generation capabilities. Alternatively, if your interest is in the business/home management field then you will perhaps be more concerned with the number of columns displayed on the screen, the quality of keyboard, mass storage, easy printer connection and, once again, lots of software.

While computers are extremely versatile, some are better suited to a particular task than others. It really is essential to make a comprehensive checklist before you set out to buy your first machine. Put down absolutely anything you can think of, perhaps allocating to each a value from one to ten in order of priority.

For a general-purpose machine, look for a good dialect of BASIC, a comfortable keyboard, wide expandability and a reasonably large memory as standard.

It pays to shop around. You are unlikely to be

able to do a deal at a high street chain store — their prices are generally quite low anyway — but you might be able to persuade the salesperson to throw in a game or two. Don't neglect the secondhand market. Millions of people have bought home computers. Many of them want to upgrade their computing power to the point where it is more sensible to buy a new machine completely, and while trading-in is not unheard of, it is still far from normal. Look in the 'For Sale' pages in the computer press and in your local newspaper.

When you start to think about upgrading your computer, or buying peripherals, it is even more important to study the market carefully. If you are a Spectrum owner, for example, you will be restricted to Sinclair's own peripherals which, while quite effective, are rather limited in their scope. Or you might want to increase the memory capacity of your Commodore Vic-20, when you will be faced with the choice of Commodore's own units, or very similar ones from independent manufacturers, which either offer the same for less, or perhaps slightly enhanced power for the same price.

Perhaps the area of widest choice is printers. Not only is there a huge selection of printer manufacturers, but also some half a dozen different types — dot matrix, daisy wheel and ink jet being the most common. If you are interested in word processing, then you will probably want a letter-quality daisy wheel printer, which produces work that looks like copy from a superior electric typewriter. If you are producing budgets by means of a spreadsheet program, then the maximum line length will be your first criterion.

Next in terms of width of choice comes mass storage. Does one invest in the higher-priced floppy disk drives, or compromise with a stringy-floppy (see page 224)? Once again, the more you spend, the better you are likely to be served.

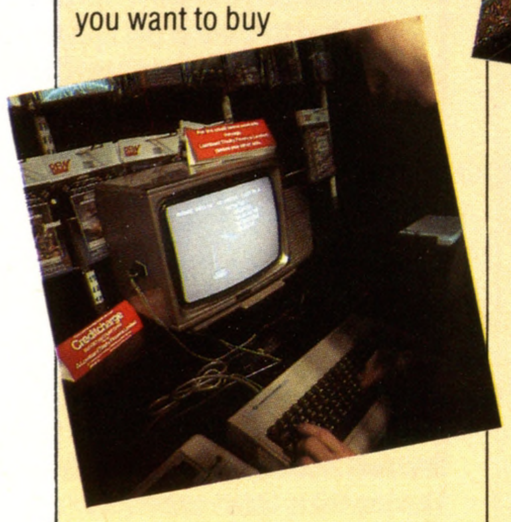
One question that seems to haunt the would-be purchaser is: 'What if I buy now, and then the price drops by ten per cent next week?' All the signs are that this situation will continue for the foreseeable future. By judicious reading of the computer press it is often possible to anticipate price cuts, but in any event, that is no good reason to put off buying a computer. It is much more important to be quite sure that you are buying the right machine for your needs and, as we observed before, the surest way to achieve that end is by thinking carefully about your application first. Always remember that your computer system is much more than the box with a keyboard that connects to your television set: It is more, even, than your software and peripherals. Your computer is a tool that when used to its full capacity can provide hours of entertainment, solve problems and keep concise and easily accessible records.

Software First

Unless you intend to use your computer solely for learning to program, the availability of good applications software will be as important as the computer's specifications, if not more so. The better a computer has sold, the more likely it is that software will be developed for it, and the more software it has, the better it will sell — a true chicken-and-egg situation

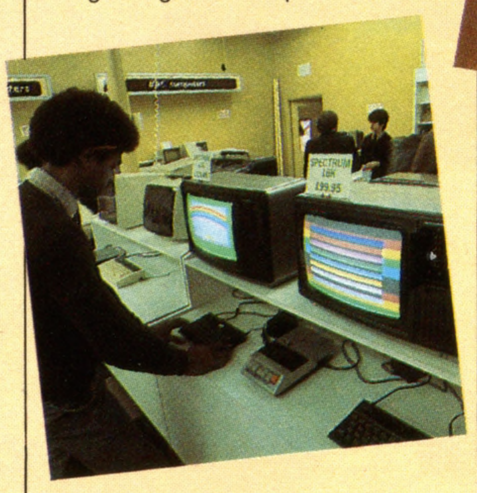
Window Shopping

Discount Store
Shops that trade in discount hi-fi and video can usually offer a good price on microcomputers — if you already know which model you want to buy



High-Street Chain
More and more people now buy their computers from the larger high street department stores, which offer the advantages of bulk buying. At one time the sales staff's computer knowledge was minimal; now many are setting up dedicated computer departments

Computer Dealer
If you buy your computer from a professional dealer, you can be sure of decent after-sales service, and knowledgeable recommendations on software, though you may not be getting the best price



Mail Order
This used to be a popular way of selling microcomputers, and by far the most profitable method for new manufacturers. Long delivery times on promised new computers, however, was the main cause of its demise

Commodore Vic-20

Commodore's smallest machine offers quite sophisticated features to the home user at a reasonable price

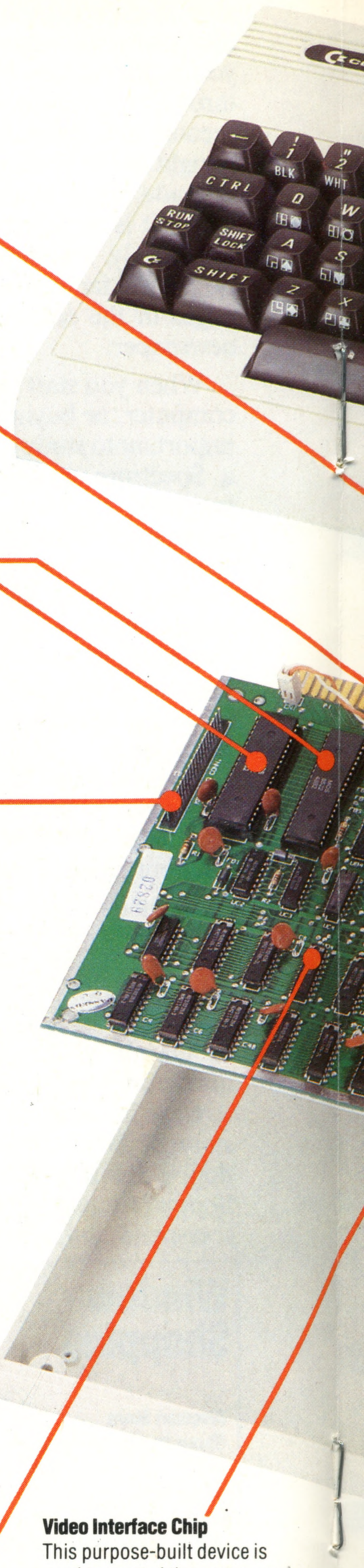
Commodore Business Machines was responsible for one of the first home micros — the Personal Electronic Transactor (PET) — which became available to the public in 1977. In 1981, it launched the Commodore Vic-20, which incorporates many of the same features as the PET. Not only does the Vic use the same 6502 microprocessor, but even the same BASIC in ROM, unfortunately not Commodore's most recent and efficient version.

The most obvious difference between the two machines lies in the Vic's additional graphics capabilities. Its name is taken from the dedicated chip that drives its screen display — the Video Interface Chip. Up to 16 colours are available, though the display is made up of a frame or border, for which eight colours are available; a background, which can be any one of the full set of 16; and the individual characters or symbols, which can be any one of eight.

The character set itself is impressively large, offering both upper and lower case and two sets of graphics characters from 62 keys, with a further four dedicated keys that can be used — shifted or not — to provide eight programmable functions. The design of the keyboard is particularly good, both ergonomically and technically.

The main drawback of the Vic is its small memory capacity — only five Kbytes, which is reduced to 3.5 Kbytes after the operating system has appropriated RAM for the screen and other internal requirements. However, up to 32 Kbytes of memory can be addressed, and additional memory is available from a variety of sources.

Interface ports are provided for paddles/joystick/light pen, games cartridges/memory expansion, printer/disk drive, cassette and television, and there is one that meets the RS232 serial standard, which could be used with a modem or non-Commodore printer. In addition, a comprehensive range of hardware add-ons is available, in common with the more recent Commodore 64 (see page 49).



Cassette Port
The Vic-20, in common with all other Commodore micros, needs a specially made cassette recorder, which plugs in here

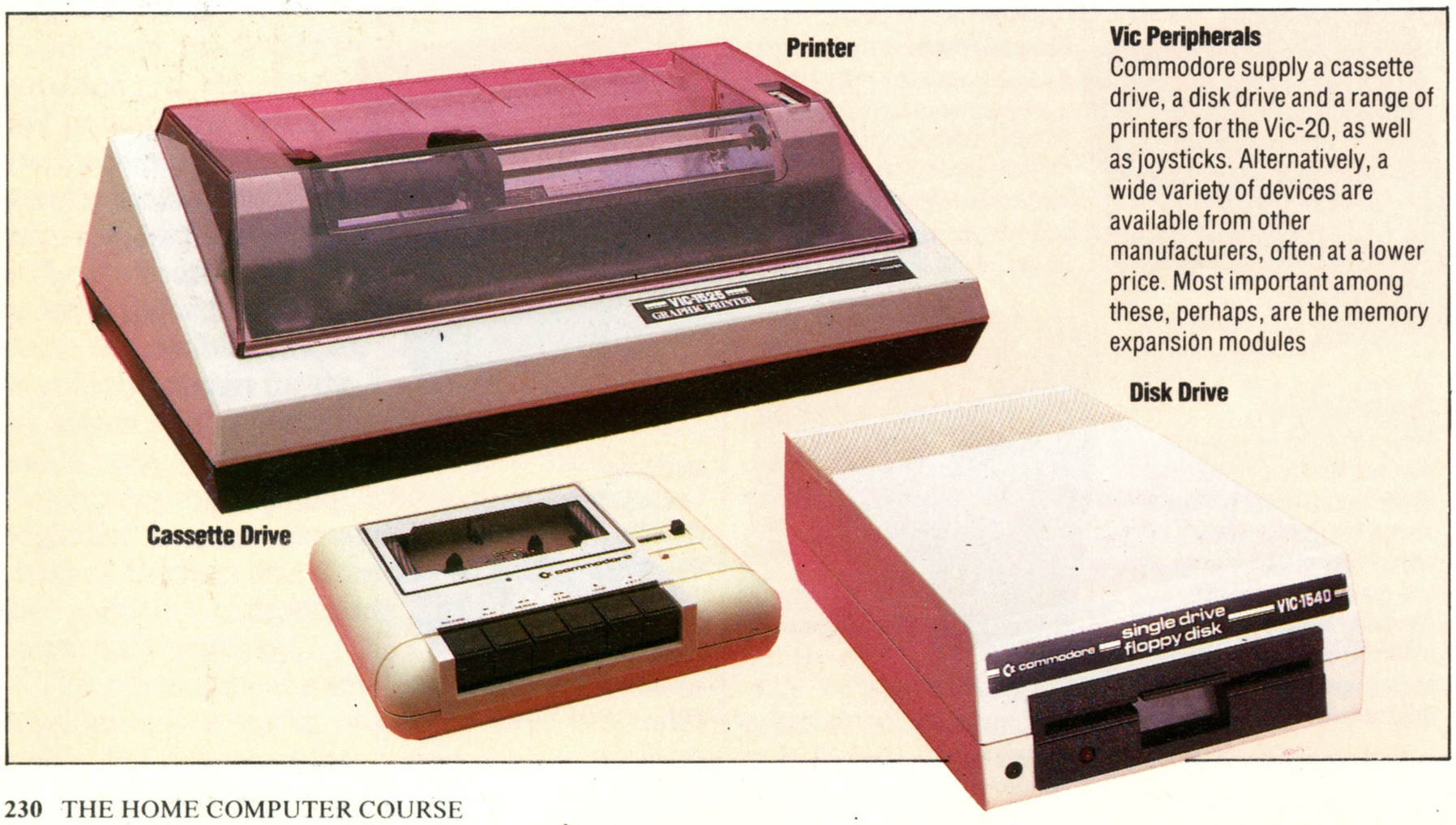
User Port
This 24-pin connector is a serial port, used to drive a variety of additional peripheral devices

Peripheral Interface Adaptor
These chips control all the Vic-20's Input/Output operations, and have some processing capability of their own. They are able, for example, to convert between serial and parallel formats

Keyboard Connector
The keyboard connects here to the peripheral interface adaptor

Video Interface Chip
This purpose-built device is used to control the appearance of the screen, and also the Vic-20's three-voice sound generator

Random Access Memory (RAM)
The Vic-20 is fitted with 5K of RAM, though this can be expanded externally to 32K



Vic Peripherals
Commodore supply a cassette drive, a disk drive and a range of printers for the Vic-20, as well as joysticks. Alternatively, a wide variety of devices are available from other manufacturers, often at a lower price. Most important among these, perhaps, are the memory expansion modules

Keyboard
The full-size keyboard uses 62 keys to provide four complete character sets, and allows the use of up to eight programmed functions from the dedicated keys on the right-hand side

CHRIS STEVENS

Cartridge Slot
This slot will accept ROM software cartridges or memory expansion boards

Power Socket
The external power supply plugs in at this point

On/Off Switch

Games Port
Light pens, joysticks or paddles can be connected at this point

Audio/Video Socket
An external RF modulator plugs in at this point

Serial Port
Most of the Vic's peripherals, the printer and disk drive for instance, are driven from this purpose-designed port, which, unlike most serial interfaces, can address more than one peripheral device at once

Microprocessor
The Vic-20 uses the well-tried and extremely popular MOS Technology 6502 microprocessor

Read-Only Memories (ROM)
These chips contain BASIC, the character set, and the other pieces of information needed to interpret and run programs

COMMODORE VIC-20

PRICE
£149.95 inc. cassette deck

SIZE
404 × 216 × 75mm

WEIGHT
1,820g

CLOCK SPEED
1 MHz

MEMORY
The machine comes, as standard, with 5K of memory, expandable to 32K by means of one 3K, one 8K and one 16K adaptor

VIDEO DISPLAY
23 rows of 22 characters. High resolution graphics allows 184 × 176 pixels. A maximum of 16 colours are available

INTERFACES
Two serial ports, audio/video connector, cassette port, cartridge slot, games port

LANGUAGES SUPPLIED
BASIC

OTHER LANGUAGES AVAILABLE
Assembler and additional BASIC commands

COMES WITH
Power supply, aerial lead, manual

KEYBOARD
Full size typewriter-style keyboard with 62 keys and four dedicated function keys

DOCUMENTATION
One of the areas in which CBM falls down badly is in documentation for its home computers. Business systems like the 8032 come with a copy of Adam Osborne's excellent 'Guide To CBM Computing', but Vic owners are relegated to a much more mundane level. While simply written, and therefore easy to understand, the users' manual barely skims the surface of the machine's capabilities. Happily, there are a large number of alternative publications available

Directory Enquiry

Using all the techniques of Basic programming that we have learnt so far, we now take the first steps towards developing a database program

Now that many of the fundamentals of BASIC have been covered, it is time to put what we have learned to good use by developing a real program. Of course, all the programs we have encountered so far have been 'real' programs in the sense that they performed a specific task, but they were illustrations of how various parts of the BASIC language work rather than the kind of program you might want to use every day. They illustrated how the 'cogs' of BASIC could be put together to make a simple mechanism. Now let's build a whole clock!

One of the most common questions put to computer users by non-users is: 'What can the computer actually be used for?' The question is not as simplistic as it seems. The usual responses tend to go along the lines of: 'Well, you can computerise your recipes' or 'You can create a computerised telephone or address book.' This tactic seldom works because the questioner usually responds with remarks like: 'I can look at my recipe book when I want to cook something and I can look through my address book when I want to find somebody's address, without the trouble of spending hours writing a program to do it.' At what point does a problem become worthy of a computer solution rather than a conventional solution? We'll answer this by working through the specific example of a computerised address book.

An ordinary address book commonly features an alphabetical finger index designed to enable the user to locate, very approximately, the location of any particular name. Names are usually added as and when needed, and not in strict alphabetical order. Your first entry under P might be David Peterson. Later, you might add Brian Peters or Shashi Patel. Although these are not in alphabetical order, they are all grouped together under P so the task of finding any particular surname beginning with P will not be too great. On the other hand, if you did not use any kind of index finding a name would be a nightmare.

The other entries usual in an address book are the person's address and telephone number together, perhaps, with more personal information. A conventional address book, however, couldn't give you a separate list of all the people who live in Birmingham, or who corresponds to the telephone number 258 1194.

Now this may not represent a serious shortcoming, but if you were the owner of a small mail order company it could be a valuable asset if you could obtain specific information about the people on your mailing list. For example, if you had a new line of children's nightwear you could anticipate new orders by informing your clients, but to save on postage it would probably not be worth sending 'mail shots' to customers without children. This is the sort of consideration that must be evaluated before deciding if a problem is worthy of a computer solution or a conventional solution.

If the computer solution is suitable, then the next consideration must be whether to buy commercially available software or not. A glance through the advertisements in the computer magazines would suggest that every possible eventuality has already been thought of by computer programmers. Closer examination, however, may show that a commercially available program may not do exactly what you want, or it may not be available for your model of computer, or it may be too expensive. The cost of a program generally reflects the development costs. A word processor package might cost £350, but writing your own could cost you far more if it takes you six months of dedicated work.

On the positive side, software you write yourself can be made to do exactly what you want it to. The other factor is the sheer satisfaction of successfully writing a major program for yourself and by yourself.

Designing a program involves several stages. The first is a thorough understanding of the problem. This involves a Clear Statement of the Problem — the CSP stage.

The second is to find an approach to the solution of the problem. This involves a description of the expected form of the input and output as a 'first level description' of the problem. The problems and the solutions should be stated in the broadest terms and these should be gradually refined until we are at the stage where we can code into a particular language.

The third stage is the coding itself. We will use BASIC as our high level language, but it could just as well be any other language. Up to the final stage of coding into BASIC, we will use a pseudo-language intermediate between the freedom and flexibility of English and the rigid structures of an actual computer language such as BASIC.

The approach to programming just described is usually called 'top-down' programming. It works from the topmost level — a general statement of the overall objectives, through various levels of

refinement, down to the fine details of the program needed to start coding into the chosen high level language. We shall also try to adhere to the principles of so-called 'structured programming'. These principles will become clear during the course of developing this project.

The steps in the development of the program can be summarised like this:

1. A clear statement of the problem
2. The form of the input and output (first level description)
   2.1 Refinement (second level description)
   2.2 Further refinement (third to nth level description)
3. Coding into high level language

Before embarking on a major software project, it is essential to state the problem clearly. This is a far from trivial exercise. Let's try a few ideas for our computerised address book.

First we'll start with a list of desirable features — later we can decide which of these can be implemented with a reasonable amount of programming effort. We want to be able to:

1. Look up an address, telephone number and notes by entering a name at the keyboard
2. Get a list of names, addresses and telephone numbers by entering only part of a name (perhaps just the surname or first name)
3. Get a list of names, addresses and telephone numbers for a particular town or area
4. Get a listing of all names starting with a particular initial
5. Get a full listing of all names in the address book, sorted alphabetically
6. Add new entries at will
7. Change entries at will
8. Delete entries at will

Assume that the address book program has been written. What form should the input and output take? How would you like the program to work from the user's point of view? Broadly speaking, programs can be 'menu-driven', 'command-driven' or a combination of both. In a menu-driven program, at every point where a decision has to be taken the user is prompted with a list (menu) of options. The selection can usually be made by pressing just one key. With command-driven programs the user types in specific command words or phrases, usually without prompting. Some programs combine both techniques. The advantage of a menu-driven program is that it should be easy for a newcomer to use, making the program more 'user friendly'. A command-driven program should be faster for an experienced user to use. We will opt for a menu-driven approach, though you might decide to implement this program using command routines instead — the choice is yours.

Given that the program will be centred on a list of names, the first thing we must consider is what form those names should take. Will the computer understand all of the following formats, for example?

A. J. P. Taylor
Leonardo da Vinci
Glenda JACKSON
Liz T.
P. O'Toole
e. e. cummings
P Jackson
Twiggy
GROUCHO MARX
Sir Freddie Laker

This may seem like splitting hairs, but consider what would happen if you had entered P Jackson and then you asked the program to search for P. Jackson. Unless you had anticipated the problem, the computer would probably respond with NAME NOT FOUND.

There are two ways the problem could be tackled: we can either have 'fuzzy' input, allowing names to be input in any form, together with clever routines to allow for this when searches are made; or we can insist on names being input in a strictly defined form. Any name that did not conform to the convention would result in an error message such as NAME FORMAT UNACCEPTABLE. The choice is an arbitrary one, but we shall opt for very 'fuzzy' input and let the program worry about converting names to a standard form.

From the point of view of an alphabetic search, names can be thought of as having two parts — surnames and the rest. A surname is relatively easy to define: any string of upper or lower case alphabetic characters terminated by a Carriage Return and preceded by a space (ASCII 32). A problem immediately presents itself: what would happen if the name 'Twiggy' were entered without a space at the front? Presumably the program would reject it as an unacceptable format. We'd better change our definition.

A name comprises one or more parts: a surname or a surname and forename. The name may consist of either upper or lower case characters, full stops, apostrophes and hyphens. It will always start with an alphabetic character and be terminated with a Carriage Return (terminal full stops will not be allowed). If there is a space, the last group of characters — including apostrophes and hyphens — will be counted as a surname and other parts, including the space, will be counted as the forename. If there is no space, the whole name will be counted as the surname.

The surname needs special consideration because in any alphabetic search, it always takes precedence over forenames. Thus Albert Peterson would come after Zoltan Patel. If a name consists of only one group of characters, such as Trevanian, Twiggy or a nickname like Baldy, it can be considered as a surname for the purposes of our program.

In an alphabetic search, which name would come first — A. J. P. Taylor or Alfred Taylor? The decision is arbitrary, but the simplest solution would be to ignore the full stops and spaces before

the final space and to make the names equivalent to AJP TAYLOR and ALFRED TAYLOR. If we were to do this, both AJP and ALFRED could be considered as forenames, and so AJP would come first.

Part of our program would accept as an input a name and produce as an output a name, address and telephone number (note that we have not even begun to consider the meanings of 'address' and 'telephone number'). If we were to accept names with a 'fuzzy' format as input, with internal conversion to a standardised format, would we expect the output to be in the 'standardised' form, or in the same form as the original entry? The most 'user friendly' output would be for the name to be in the original form, but, as we shall see, this will complicate the programming.

As an initial programming task, let's suppose that a name has been assigned to the string variable NAME$ and that we have two other variables, FORENAME$ and SURNAME$. How will we assign the appropriate parts of NAME$ to FORENAME$ and SURNAME$? Ignoring, for the moment, the problem of keeping a record of the original form in which the name was entered (so that it can be retrieved when needed later), a simple statement of the program could be:

Convert all characters to upper case
Eliminate all non-alphabetic characters except the final space
Assign all characters following a final space to SURNAME$
Assign all characters preceding a final space to FORENAME$

Before considering how this problem could be coded into BASIC, we'll see how the process of 'top down programming' can take us from a very broad statement of our objective to the point where coding into a particular programming language becomes possible. You will notice that we are using not only long variable names like SURNAME$, but command words like BEGIN, LOOP and ENDLOOP. These are constructions that we have invented to help us describe our program. At the final stage of development, they will be replaced with equivalent commands from BASIC. We'll explain more about these commands, and why we have indented some of the lines in the next instalment of the course.

1ST STATEMENT OF OBJECTIVES

INPUT
A name (in any format)
OUTPUT
1. A forename
2. A surname

1ST REFINEMENT

1. Read NAME$
2. Convert all letters to upper case
3. Find last space
4. Read SURNAME$
5. Read FORENAME$
6. Discard non-alphabetics from FORENAME$

2ND REFINEMENT

```
1. Read NAME$
2. (Convert all letters to upper case)
   BEGIN
   LOOP while unscanned characters remain in NAME$
      Read out characters from NAME$ in turn
      IF character is lower case
         THEN convert to upper case
         ELSE do nothing
      ENDIF
      Assign character to temporary string variable
   ENDLOOP
   LET NAME$ = temporary string variable
   END
3. (Find last space)
   BEGIN
   LOOP while unscanned characters remain in NAME$
      IF Character = "     "
         THEN note position in a variable
         ELSE do nothing
      ENDIF
   ENDLOOP
   END
4. (Read SURNAME$)
   BEGIN
   Assign characters to right of last space in NAME$
   to SURNAME$
   END
5. (Read FORENAME$)
   BEGIN
   LOOP while unscanned characters remain in NAME$
   up to last space
      SCAN characters
      IF character is not a letter of the alphabet
         THEN do nothing
         ELSE assign character to FORENAME$
      ENDIF
   ENDLOOP
   END
6. (Discard non-alphabetics from FORENAME$)
   (This has been handled in 5 above)
```

This second level refinement is now very near the stage where it could be coded into a programming language. Let's develop 2 (Convert letters to upper case) to a third level of refinement and then code it into BASIC. We've encountered an algorithm for doing this before (see page 212).

3RD REFINEMENT

```
2. (Convert all letters to upper case)
   BEGIN
   READ NAME$
   LOOP
   FOR L = 1 TO length of string
      READ character L
      IF character is lower case
         THEN subtract 32 from ASCII value of
            character
         ELSE do nothing
      ENDIF
      LET TEMPSTRING$ = TEMPSTRING$ + character
   ENDLOOP
   LET NAME$ = TEMPSTRING$
   END
```

This program fragment in pseudo-language is now close enough to a programming language to be coded. Our version of Microsoft BASIC does not allow the names of string variables to be full words, so single letters will be used instead. Thus NAME$ becomes N$.

```
1000 REM UPPER CASE CONVERSION SUBROUTINE
1010 INPUT "INPUT NAME"; N$: REM FOR TESTING
     ONLY
1020 LET P$ = "": REM ENSURE STRING IS EMPTY
1030 FOR L = 1 TO LEN(N$): REM INDEX OF LOOP
1040 LET T$ = MID$(N$,L,1): REM EXTRACTS
     CHARACTER
1050 LET T = ASC(T$): REM FINDS ASCII VALUE OF
     CH.
1060 IF T >= 97 THEN LET T = T - 32: REM U/C
     CONV.
1070 LET T$ = CHR$(T)
1080 LET P$ = P$ + T$: REM P$ IS TEMPSTRING
1090 NEXT L: REM END OF LOOP
1100 LET N$ = P$: REM N$ IS NOW ALL UPPER CASE
2000 PRINT N$: REM FOR TESTING ONLY
2010 END: REM FOR TESTING ONLY. SUBSTITUTE
2020 REM 'RETURN' IN ACTUAL PROGRAM
```

This program fragment would normally be used as a subroutine to be called from a main program. We have written it for testing purposes with an INPUT statement, a PRINT statement, lots of REM statements, and an END. These would be removed before the subroutine was incorporated into a full program.

Basic Flavours

LYNX

The Lynx has a function, UPC$(A$), that converts the letters in A$ to upper case, so the program reduces to:

```
1010 INPUT "input name";N$
1020 LET N$=UPC$(N$)
2000 PRINT N$
2010 REM "RETURN" HERE IN ACTUAL
     PROGRAM
```

SPECTRUM

The Spectrum program in full is:

```
1010 INPUT "INPUT NAME";N$
1020 LET P$=""
1030 FOR L=1 TO  LEN  N$
1040 LET T$=N$(L)
1050 LET T= CODE T$
1060 IF T>=97 THEN LET T=T—32
1070 LET T$=CHR$ T
1080 LET P$=P$+T$
1090 NEXT L
2000 LET N$=P$
2010 PRINT N$
2020 REM "RETURN" HERE
     IN ACTUAL PROGRAM
```

DRAGON 32

This program will run on the Dragon, but lower case characters are reserved for the printer, so the problem doesn't really arise

BBC Micro

Replace line 1010 with:

```
1010 INPUT "input name", N$
```

Exercises

■ Refine all the stages above to the point where they could be turned into a BASIC program. The 'pseudo-language' you use does not need to be the same as ours, but it is a good idea to follow the convention of using upper case letters for keywords that will probably correspond to statement words in the target language (e.g. LOOP, IF, LET etc.). Use small letters for operations that will need to be stated more explicitly when finally coded. These can be in ordinary English.

■ Having developed the programs to a satisfactory level of refinement, convert them into program modules (subroutines) in BASIC. Test them individually using dummy inputs and print statements that can be removed later if they work properly.

Revision Exercises

These exercises demonstrate most of the commonly used statements and functions in BASIC. There are no 'trick' questions and nothing new is being introduced. If you can do all or most of these exercises without difficulty you may consider yourself on the way to becoming a fully-fledged BASIC programmer.

■ Write a program to accept two numbers input from the keyboard, to add them and to print out the result.

■ Assign two words (character strings) to two string variables and then create a third string variable that concatenates (joins together) the two original words. Print the third variable.

■ Write a program that allows you to type in any word on the keyboard and then prints the length of the string in the messageTHE WORD YOU TYPED HAS * CHARACTERS (* stands for a number).

■ Write a program that accepts a single character typed in at the keyboard and then tells you what the ASCII value of the character (in decimal) is.

■ Write a program that prompts you with the message TYPE IN A WORD and then answers with the message THE LAST LETTER OF THE WORD WAS *(* stands for a letter).

■ Write a program that prompts you to TYPE IN A PERSON'S NAME and then responds with THE SPACE WAS THE *TH CHARACTER.

■ How would you modify the program above to print 2ND instead of 2TH if the space was in the second position?

■ Write a program that prompts you to TYPE IN A SENTENCE and then responds with the message THE SENTENCE YOU TYPED HAS * WORDS (assume that there will be one word more than there are spaces in the sentence).

■ Test your computer to see if characters (or special graphics) have been assigned to ASCII values from 128 to 255 (use a loop and the CHR$(X) function).

Waiting Room

Computers transfer information at a rate much faster than mechanical devices like printers can handle. This problem is solved by using a short-term memory called a buffer

The buffers used by trains are designed to cushion impact by absorbing energy in springs or damped pistons. Computers have buffers, too, and in some ways they function like the train's buffers by helping different parts of the computer system to 'get on together'.

The term is used somewhat loosely in the computer world and covers two quite distinct things. To the programmer 'buffer' means a specialised use of computer memory, while to the circuit designer it means a type of electrical signal amplifier. The second type, which we shall call signal buffers, is dealt with in the panel.

Memory Buffers

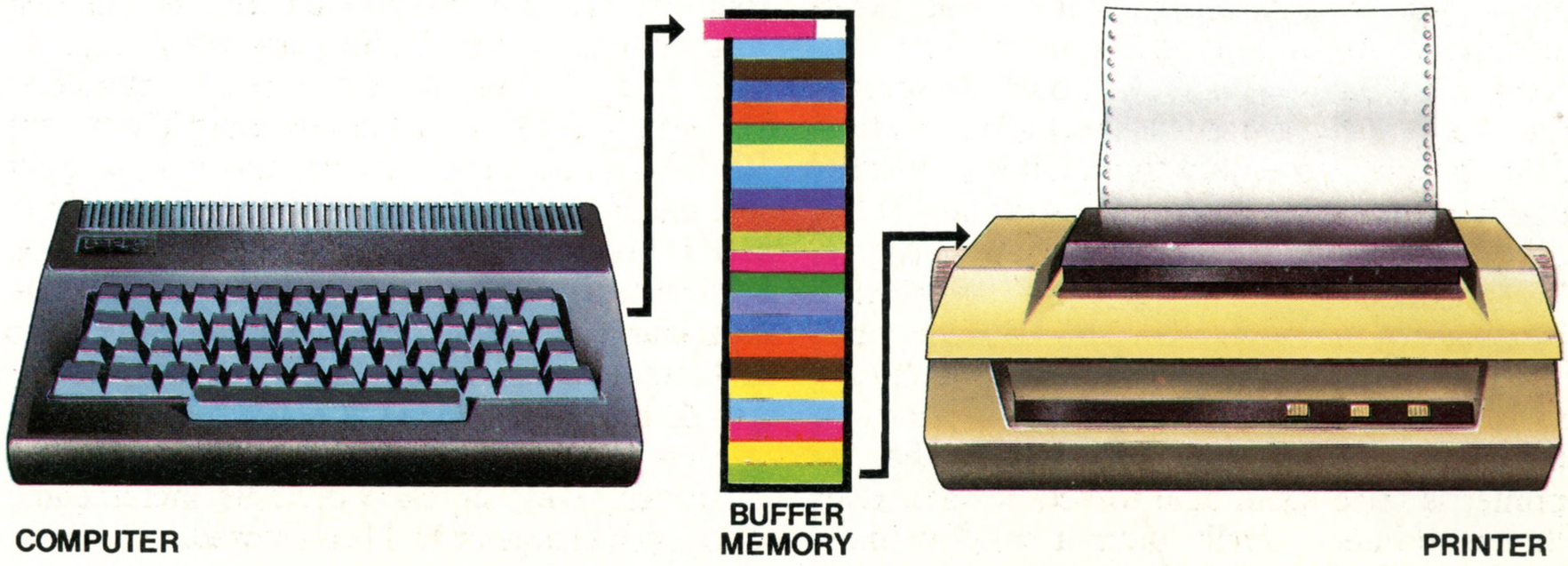
Imagine a word processor program that, among other things, can move a block of text from one part of a 'document' in the computer's memory to another part. The text consists of printable characters and spaces and certain 'unprintable' characters such as the Carriage Return. All these are represented in the computer's memory as ASCII codes in binary. One byte of memory is needed for each character. To move the characters in the block from their old positions in memory to their new ones means that another part of the computer's memory has to be set aside as a temporary text storage area. Such an area of memory specially set aside for a specific task is known as a buffer.

As a second example, consider the problem of printing a document created on a word processor. The document might consist of 15,000 separate characters, but they clearly could not all be sent to the printer for printing at once — most printers cannot print faster than about 80 characters a second. To overcome this, part of the computer's memory will be set aside, under the control of the word processor software, as a 'print buffer'. The software will first fill up this buffer with characters to be printed, and then send them out to be printed at a speed appropriate for the printer.

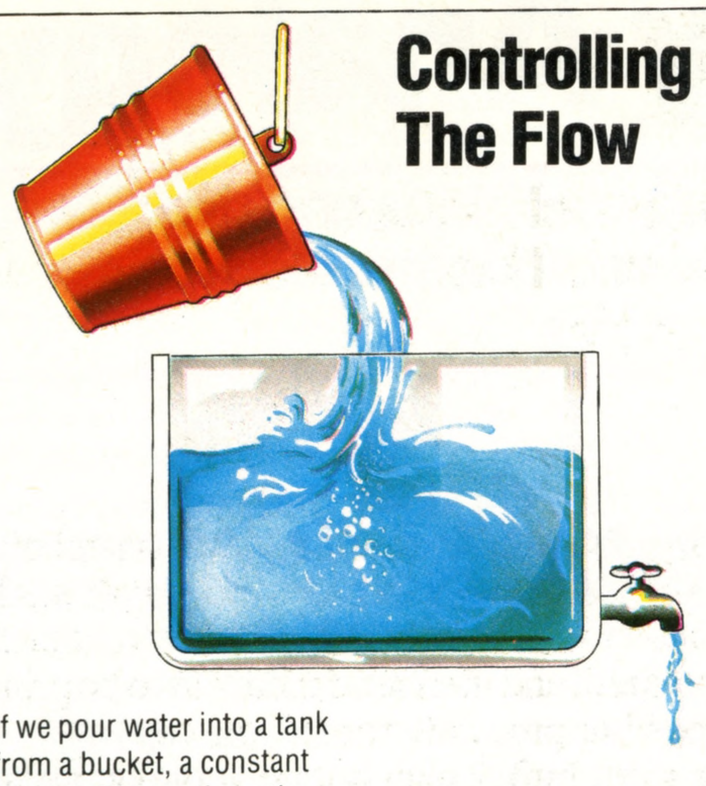
The print buffer may not be very big, perhaps only 128 or 256 bytes in capacity, but the principle remains the same however big it is. First a 'block' of ASCII characters is written into it, and then these are despatched out again, one byte at a time. The first byte to be written into the buffer will also be the first byte to be read out from it (obviously we will want the characters to be printed in the same order they were typed in). This type of buffer is called a 'First In First Out', or FIFO buffer. When all the characters have been read out from the buffer, the software will fill it with the next block of characters destined for the printer.

FIFO buffers of this type are an extremely common feature of most types of computer software. They are used wherever incompatibilities of speed exist, not only between computers and printers, but also between computers and floppy disk drives and between computers and computer keyboards. Although the phenomenal processing speed of computers means that they can usually identify which keys have been pressed faster than typists can type them, there may be times when the computer cannot identify the keys and display the corresponding characters fast enough. This can happen if the computer is momentarily busy doing something else (accessing a disk, for example). When this happens, it is common to incorporate into the computer's operating system a 'type ahead buffer'. This buffer 'remembers' which keys have been pressed and the computer displays them as soon as it can. Normally, the user will not notice

Temporary Stopover
One of the most common uses for a buffer is between the computer and a printer because the printer cannot output characters at the same speed that the computer sends them. Characters are consequently stored in the temporary memory until this buffer is full, whereupon a 'busy' signal is sent to the computer to stop it transmitting. The contents of the buffer memory are then sent to the printer in the same order that they were received, but at a much slower rate. When this has finished, the process begins again until the whole text has been printed



Controlling The Flow



If we pour water into a tank from a bucket, a constant trickle will come from the outlet pipe. In just the same way, the computer 'pours' data into the buffer, which releases it at the much slower speed at which the printer, for example, works

this happening, but with certain disk operating systems, or with certain types of applications software (involving a lot of processing of information) there may be a slight delay between a key being pressed and its appearance on the screen. A few operating systems allow the type ahead buffer to be turned on or off, or even allow the size of the buffer to be altered by the user.

The precise way in which the software is organised to handle buffers varies depending on what the buffer is supposed to do, but generally it will be necessary to set aside a few bytes to be used as counters and flags. It will be necessary to know how many bytes have been read out from a full buffer before more are written in, otherwise important data may be destroyed before it has been used.

Hardware Memory Buffers

Readers with printers may have noticed how slow they seem to be, especially when printing out a long program listing or document. Most computer operating systems are not able to do anything else while the printer is in use, so if it takes a long time to finish printing, the user will just have to sit at the screen waiting for the printout to end. Many manufacturers now offer add-on print buffers, usually in the form of a box that connects between the computer and the printer. These boxes effectively speed up the printer from the computer's point of view. Although the printer does not actually print any faster, the box contains extra, dedicated memory (sometimes as much as 16 Kbytes) with its own built-in software. To the computer it appears exactly like a faster printer. When the computer has to print a file, it sends bytes out to the printer until it receives a 'busy' signal, meaning that the printer cannot accept any more bytes. The computer then has to wait until the 'busy' line goes 'false', indicating that the printer is once again able to accept data. Even though printers usually have a small memory buffer built in, this is often not more than two Kbytes and it will not allow the computer to send more data until this buffer is empty. The add-on hardware memory buffers contain more memory, so they can accept much more data before sending a 'busy' signal to the computer. If the buffer is big enough, it may be able to hold all the data to be printed in one go, so the computer will be able to get on with other tasks while the buffer sends the data at a slower rate to the printer.

Memory is often used rather like a large trunk or pigeon-hole rack to store programs and data, but *it* may instead *be* organised into stacks or buffers. Stacks are 'Last In First Out' or LIFO structures, whereas buffers are 'First In First Out' of FIFO structures. The analogy often made for stacks is that of the pile of plates supported by a spring occasionally seen at self-service counters. Plates are piled on the stack and the last one placed there will be the first one removed. Like buffers, stacks are also temporary memory areas and differ from buffers only in the order in which the data is input and retrieved. Stacks are used 'internally' in high level languages (in BASIC interpreters, for example) when information needs to be stored temporarily and recalled later. Consider this BASIC program fragment:

```
FOR X = 1 TO 10
PRINT "X = ";X
FOR Y = 1 TO 10
GOSUB SCAN
NEXT Y
PRINT "C$ = ";C$
NEXT X
```

This is an example of nested FOR. . .NEXT loops. When the BASIC interpreter gets to the second FOR statement, it needs to remember which variable is used by the previous FOR (X in this case), and so it 'pushes' the information about the first FOR onto a stack. When the inner loop has been completed, it 'pops' the information from the top of the stack and knows that the current FOR uses variable X. Since FOR. . .NEXT loops can be nested as deeply as required, it may need to push information for several FORs onto the stack. When it pops the information from the stack, it clearly needs to have the information in the inverse order from that in which it was pushed.

Buffers, on the other hand, organise memory so that the first information entered is the first information output. Buffers are often used in input/output routines and are used as 'interfaces' between routines or devices working in different units or at different speeds. For example, an input routine in BASIC might work in units of lines, terminated by a Carriage Return <CR>, but the interpreter may work on the lines in units of one character. Buffers usually need a 'pointer' to indicate where in the buffer the next character should be written. The pointer would be a byte or several bytes containing the address of that character. The address would be incremented after each character had been stored.

Strong Signals

The logic inside your computer works at TTL levels. Transistor-Transistor-Logic signifies a binary 1 with five volts, and 0 with zero volts. However, though devices such as the CPU are capable of producing these voltages, they can't generate enough current to drive all the other chips that might be connected to each pin. Signal buffers are therefore connected to the CPU's output lines to magnify the amount of current available. Signal buffers are themselves small chips, each typically capable of acting as a buffer for six signals

Computing By Analogy

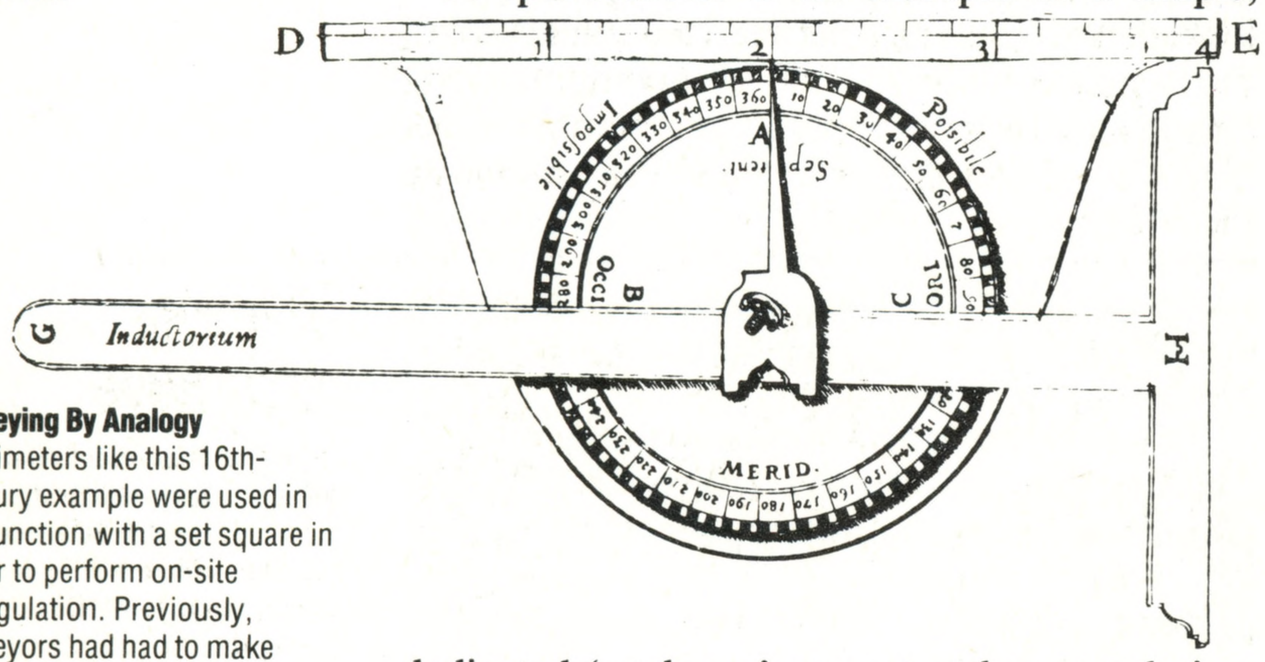
Analogue computers, used for controlling machines and processes, react directly to changes in the real world without the need to translate information into digital form

COURTESY OF PHYSICAL & ELECTRONICS LABS LTD

A Patchboard
Analogue computers do not use languages such as BASIC. They are 'programrned' by wiring together various electrical components. The components are attached to the back of a circuit board. On the front are plugs and sockets that allow selected components to be connected

There are two quite different families of computers, and so far we have been looking only at one of them: the digital computer, so named because all the instructions in a program and all the data are represented using binary 'digits'. The other, older family comprises analogue computers.

A speedometer is an example of a simple, dedicated, 'analogue' computer; the name derives from the 'analogous' behaviour of the speed of the car to the position of the needle on the dial, the former being directly proportional to the latter. Modern analogue computers can perform many tasks and are based on the type of electrical components that are commonly found in domestic appliances — transistors, capacitors, resistors and magnetic inductances. Early on in the development of electronics it was discovered that the behaviour of electrical components resembled that of mechanical devices. For example, electrical engineers discovered that the oscillations of electric current that can result when a magnetic inductance and a capacitor are connected together closely mimic the oscillations of a weight hanging on a spring. In fact, the mathematical description of both systems was identical — the basis for 'computing by analogy'.



Surveying By Analogy
Planimeters like this 16th-century example were used in conjunction with a set square in order to perform on-site triangulation. Previously, surveyors had had to make measurements and then do mathematical calculations in order to determine the length of the remote side of their triangle, but this simple analogue instrument made their work considerably easier and more accurate. Similarly-named instruments are still in use today, for measuring the area of irregular planes such as animal hides

Some analogue devices look very similar to the systems they model — for example, the model of an aeroplane used in a wind tunnel experiment is an exact copy, scaled down, of the airframe. Other models appear very different. A 'model' of a real life situation may be just a list of mathematical formulae, or it may be an electrical circuit that mimics the flow of water over a weir, for example.

Calculating machines that use analogue principles first showed their importance with the invention of the slide rule in 1630 by William Oughtred. Numbers were arranged on two rules in such a way (they were logarithmically spaced) that the movement of one alongside the other is equivalent to multiplication, and the answer can simply be read off the scale.

At the end of the 19th century Lord Kelvin designed an ingenious hand-cranked mechanical device that could be used to calculate the high and low tides at a port throughout the year.

Then in 1930 an electromechanical machine was built in America that could solve general differential equations (the type of equation that crops up in almost any mathematical representation of the real world), rather than the specific set of differential equations that Kelvin had succeeded in solving. It was invented by Vannevar Bush and called a differential analyser.

Early Analogue Computers

The first fully electronic analogue computers came into operation in 1947 just after the first digital computers were born. We have seen how digital computers do arithmetical calculations using a combination of logical gates (see page 68). The analogue computer can perform mathematical calculations simply by utilising the nature of electricity. If there is, say, an electric current of five amps flowing in one wire and a current of four amps flowing in another and the two wires are joined together to merge into a single wire, then the current in the new wire will be nine amps — the sum of the two currents. So the

behaviour and properties of the flow of current automatically make an 'adder'.

Even very complicated mathematical functions often find very simple solutions in elementary circuits (an example is 'integration' — finding the area under a curve). Analogue computers aren't 'programmed' like digital machines; instead, a circuit has to be constructed that models the program to be solved. All the available components are mounted on the back of a 'patchboard', on the front of which are plugs and sockets to wire the components together. A patchboard looks rather like an old-fashioned telephone switchboard.

In an analogue computer the varying voltages or currents are used to represent physical quantities such as force or velocity, and the 'values' of electrical components represent things like the mass of a car, or the strength of its springs. But in a digital computer all data is represented with strings of pulses, five volts for binary 1 and zero volts for binary 0. Here lies a further distinction between analogue and digital computers: in one information can be stored in a way that caters for continuously changing quantities but in the other data is stored in 'discrete', or individual, units.

Some home computers (for example, the BBC Micro) have a socket for analogue input. In general, when digital computers deal with analogue information, such as a temperature or force, they first have to convert the data into digital form.

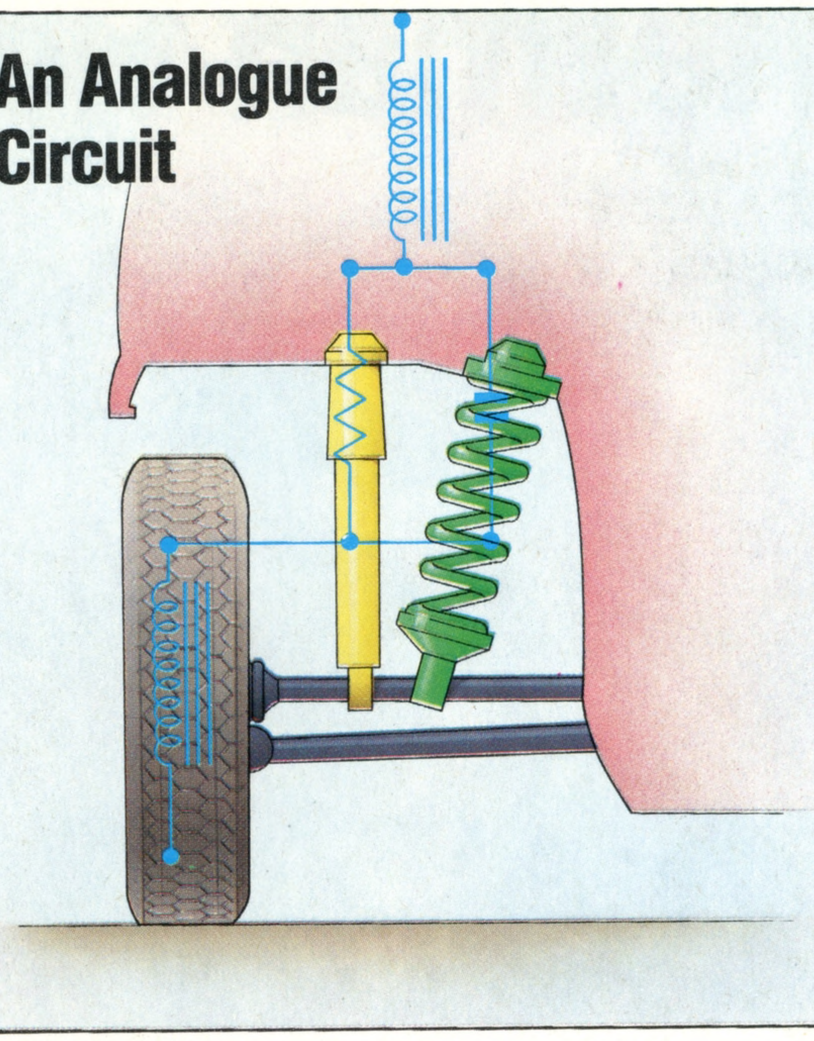
The big advantage of digital systems is that information can be processed or transmitted without loss of quality. If a five volt pulse is put through an electrical circuit it may well be affected by the distortion that is inherent in any circuit and come out, perhaps, as 4.9 volts. In an analogue system, in which voltage fluctuations represent changes in information, this might mean the difference between a singer's voice being an A or an A flat. But in a digital system, in which there are only two possible signals, five volts or zero, any signal close to five (for example 4.9) is automatically recognised and regenerated as a pure five volts. So errors are corrected and do not accumulate. By contrast, the accumulation of errors as the signal passes through successive circuits is one of the disadvantages of the analogue computer.

However, the analogue computer does reap an advantage from representing quantities as varying values of a voltage or current. It means that an input condition can change rapidly and the system will reflect the consequent changes immediately. No time is needed for coding the data into binary pulses, processing it and finally decoding it again to give the output.

This characteristic is very important in applications where rapid responses are essential. For example, an automatic pilot has to respond to a sudden gust of wind during a landing, when there is no time to do lengthy computations, even at the speed with which modern digital computers operate. Sensors detect the sudden gust, generating a relatively small output voltage. The autopilot circuit responds instantly with a relatively large output voltage change, which automatically operates the wing flaps to keep the plane in trim.

Analogue computers are used in many areas of industrial control where complex systems have to be delicately managed with fine and continuous adjustment, as in an industrial chemical plant. But they remain less well known than their digital counterparts. Though simple analogue computers are sometimes used in schools for teaching purposes, the type of application to which they are suited means that we are unlikely ever to see an analogue home computer. Thirty years ago it looked like a close race between the analogue and the digital computer. Analogue computers will always have their uses but as digital computers get bigger and faster they will increasingly dominate the market.

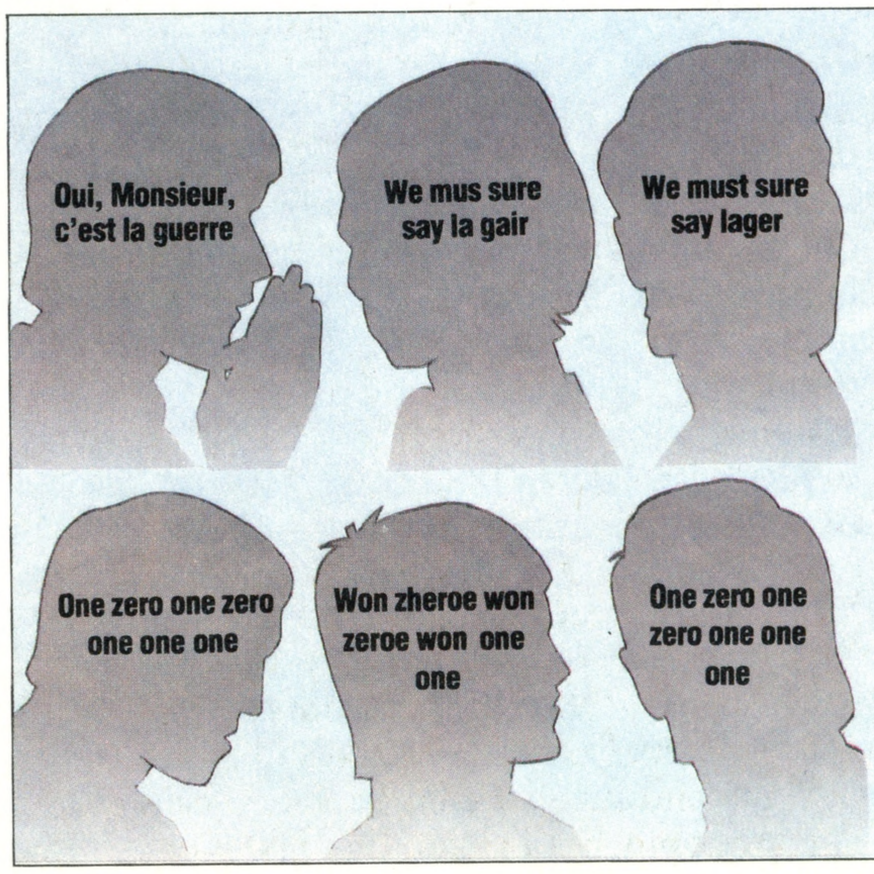
An Analogue Circuit



Car suspension systems consist of a spring and a damper. The spring absorbs the sudden shock of hitting a bump and then the oscillations of the spring are dissipated by the damper. The engineers have to select the best size of spring and damper to give the most comfortable ride over different road surfaces at different speeds. An electrical circuit can be used to represent the arrangement of spring, damper, car body and wheel. A fluctuating voltage is applied at one end to represent the bumps on the road and the size of the electrical components is varied until the output voltage is as smooth as possible

KEVIN JONES

Chinese Whispers



The way in which a message gets distorted in a game of Chinese Whispers has a strong parallel with the cumulative errors in an analogue circuit

In a digital circuit, only a limited number of messages (effectively 1 or 0) can be passed. So if any distortion does arise, it can easily be identified and eliminated at the next stage

Herman Hollerith

1860
Born in Buffalo, New York State

1879
Graduates from Columbia University and becomes an assistant at the US National Census Office

1882
Enrols at Massachusetts Institute of Technology to do further research

1883
Works in Washington for the Patent Office

1884
Applies for his first patents for representing information on continuous punched paper tape

1887
His system is adopted for processing mortality statistics in Baltimore, Maryland

1889
System installed in Surgeon General's office for organising army medical statistics. Files a patent for the concept of the individual punched card

1890
Wins competition for supplying equipment to process the 1890 census. Gains a PhD from Columbia University for work on information processing

1900
Introduces a new generation of improved equipment

1901
New equipment used in the census of agriculture

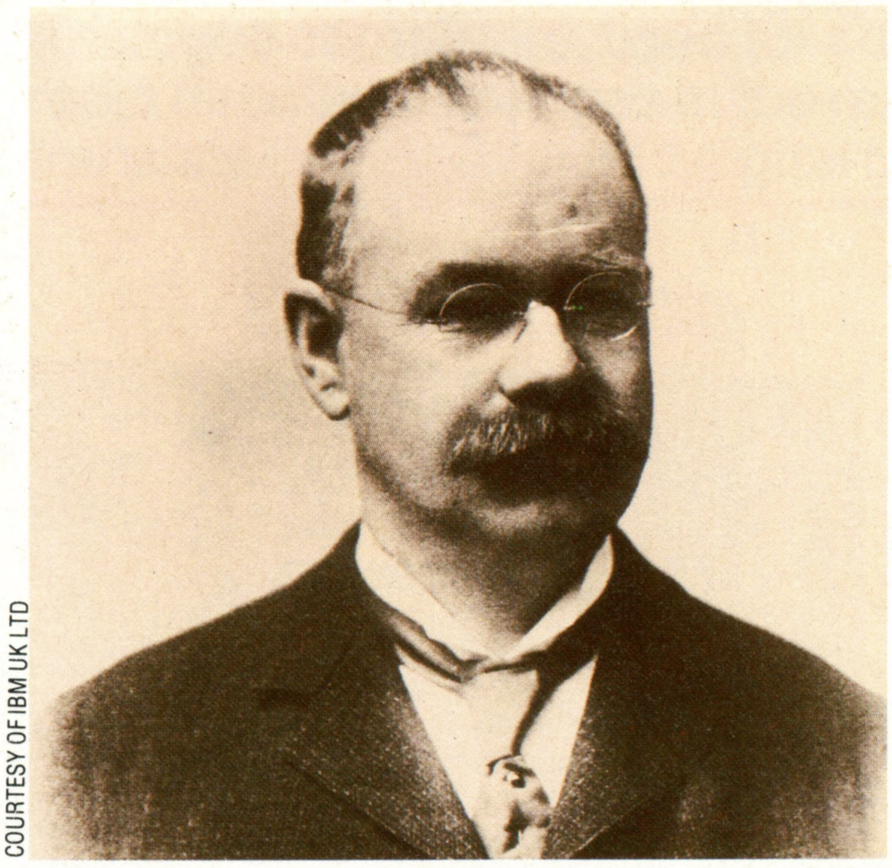
1905
First patents start to expire and competition begins against his monopoly

1911
He forms a holding company, the Tabulating Recording Company

1914
The famous business administrator Thomas J Watson takes over the running of the company

1924
Company changes its name to International Business Machines (IBM)

1929
Dies in Washington DC



COURTESY OF IBM UK LTD

The inventor who put America's population figures on punched cards and went on to found the world's largest computer company

Hollerith was born in America in 1860. After graduating from Columbia University he became an assistant at the US National Census Office, helping to compile statistics from the census of 1880. The work was all done by hand and was painstakingly slow — so slow, in fact, that when the time came for the next census, 10 years later, the Office was still tabulating the results. Hollerith knew his strength lay in invention; in order to train himself as an inventor and develop his creative skills he left the National Census Office and joined the Patent Office in Washington.

Hollerith's first idea was to code information onto paper tape. The paper tape was marked with ink into 'fields'. Each field represented different categories — say male or female, or black or white. The presence of a hole in the male/female field meant that the subject was male, while its absence denoted a female, and so on. These holes could later be 'read' by machine. His first patents came out in 1884, and over the next few years he improved his system. He began by processing information on health statistics for the fast developing American cities and the army administration.

Five years later in 1889 he improved on the idea of punched paper tape by using separate cards for each individual. The cards were the size of a dollar bill — partly, it is said, because the only equipment that could be adapted had been built for handling money. The holes were originally round and made with a bus conductor's punch, but later special punches were built to cut a 6mm (¼in) square hole. In this way a great deal of information could be included on a single card.

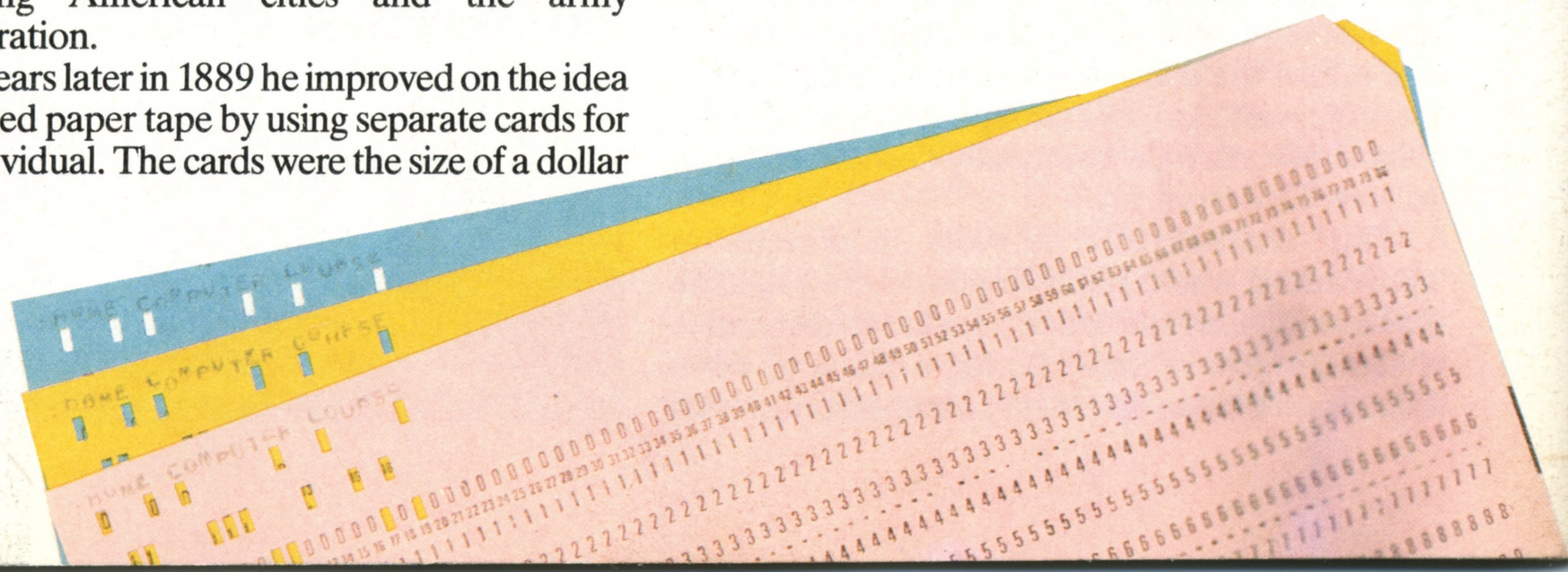
The advantage of individual cards over continuous tape is that information can be sorted as well as totalled. For example, you might want to find out the number of white women aged 80 living in New York City. All the cards would be sorted through and any with holes punched in these three fields could be mechanically separated from the rest. These early machines could produce only a running total but later on, in order to improve the efficiency of his machines, Hollerith introduced addition and other simple arithmetic operations, proving that machines could replace the human hand.

Commercial success came in 1889 when the Bureau of Censuses announced a competition to supply a system of equipment to process the coming year's census. Systems were tested by retabulating data from the previous census. Hollerith's equipment won. His machines were by then all protected by patent and he took advantage of his monopoly by charging the Government 65 cents to process every thousand cards. Though there was an individual card for every person in America, Hollerith took only two years to finish. He announced that the population stood at 56 million, and invoiced the Government accordingly.

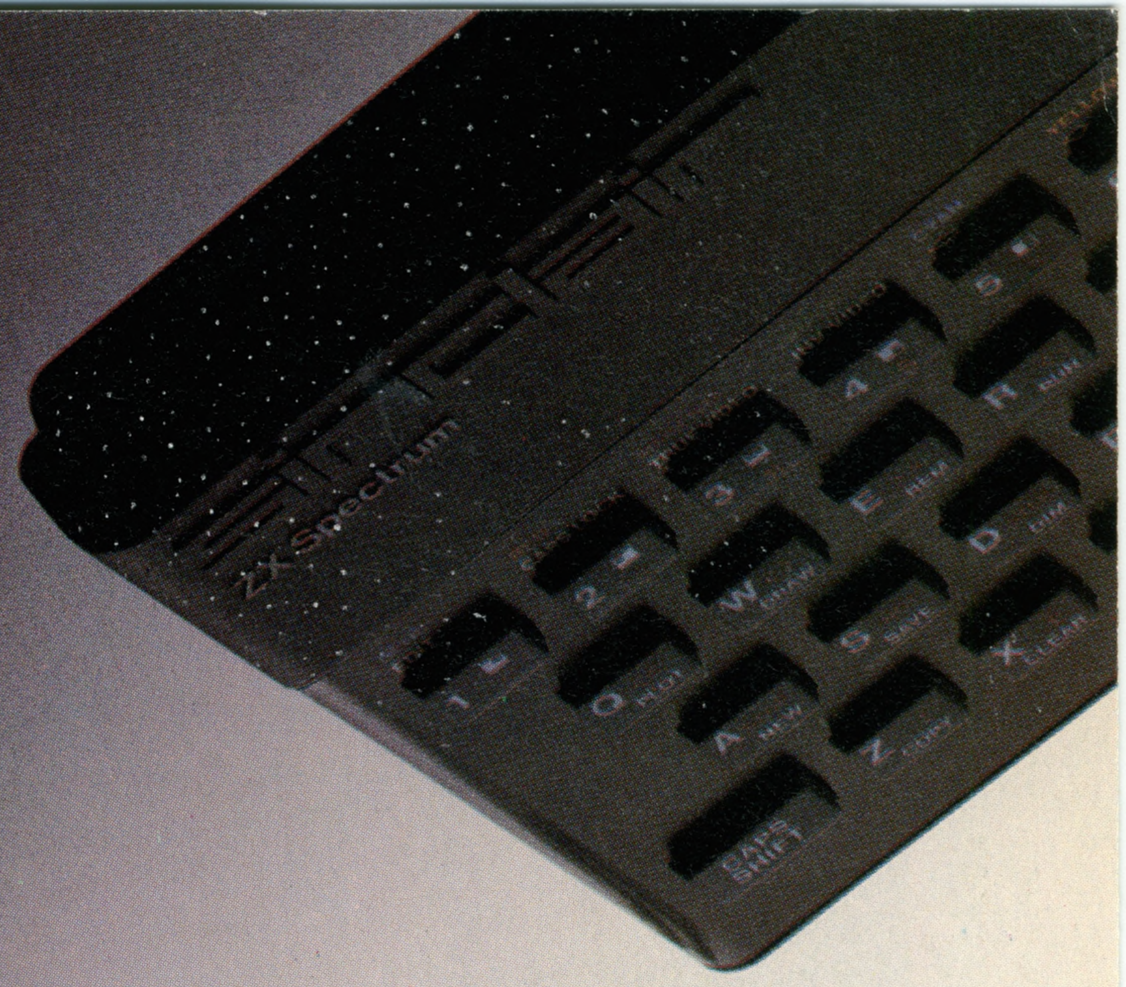
By the time of the 1900 census he had developed far more efficient machinery but he refused to lower his charge. When his patents ran out the Government looked to other companies but Hollerith overcame the competition by forming his own company, which later became International Business Machines. Today IBM is the world's largest computer manufacturer with an annual turnover of 20 billion dollars.

Card Games

Hollerith's original method of representing information is still in use today, a century later, though the format of the cards has changed. Modern punched cards have 12 rows of 80 columns. Tabulating machines used the decimal system, and so each card could store 80 numbers. Alphabetic characters were created by 'multi-punching' — making more than one hole in a column. Computers can also accept cards that are punched in the binary system



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

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