

**Sinclair**

# QL USER

**Unlocking Sinclair's Supermicro**

## **GETTING TOUGH WITH THE QL**

*Our 'no-holds-barred' test tells all.*

## **AND WHAT ABOUT THE SOFTWARE?**

*Abacus, Archive and Quill - are they any good?*

## **C PROGRAMMING WITH THE QL**

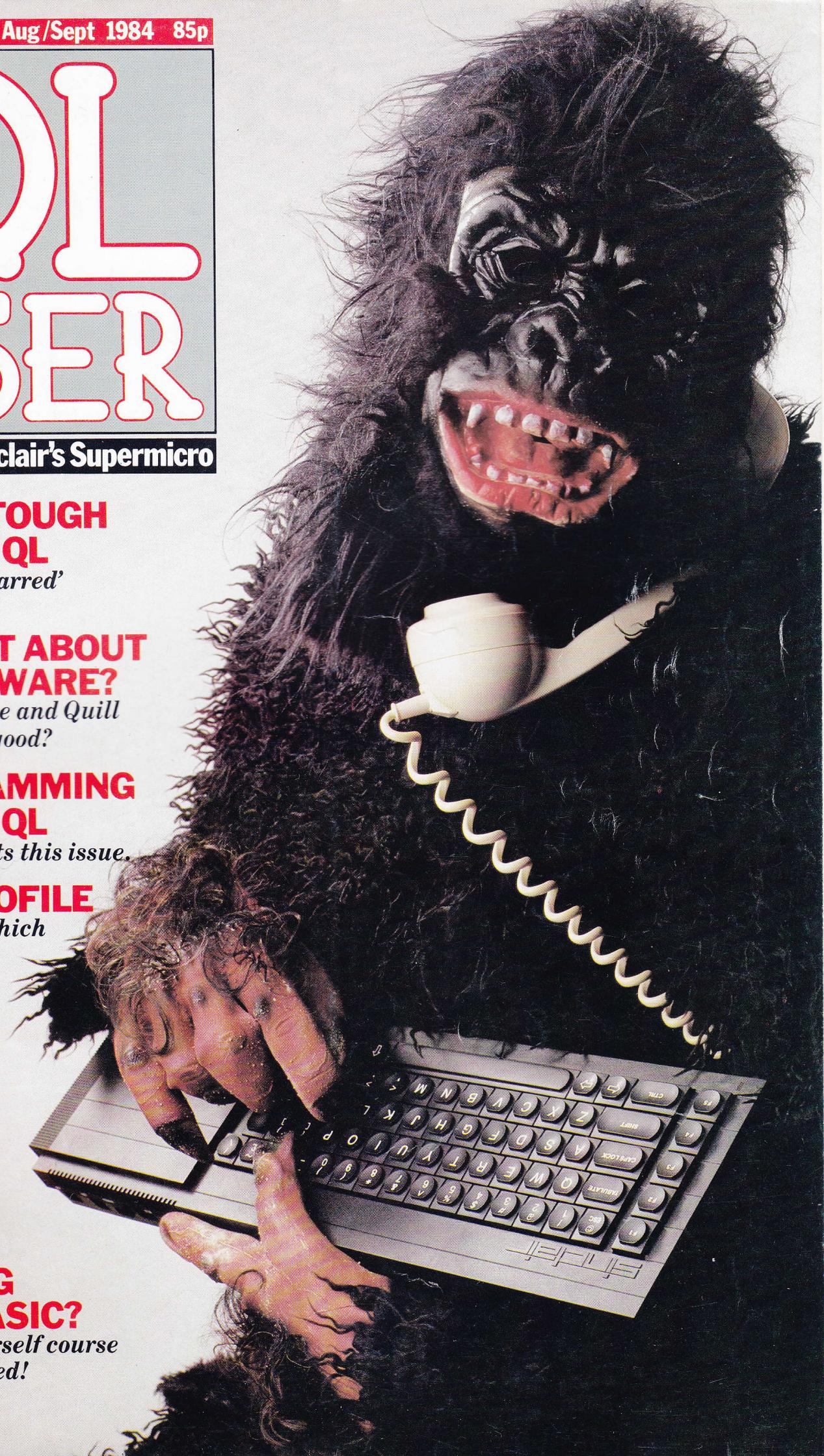
*New series starts this issue.*

## **PSION PROFILE**

*The company which wrote the QL software.*

## **LEARNING SUPERBASIC?**

*Our teach-yourself course is what you need!*



# EVEN THE PRICE WILL KEEP YOU IN THE BLACK

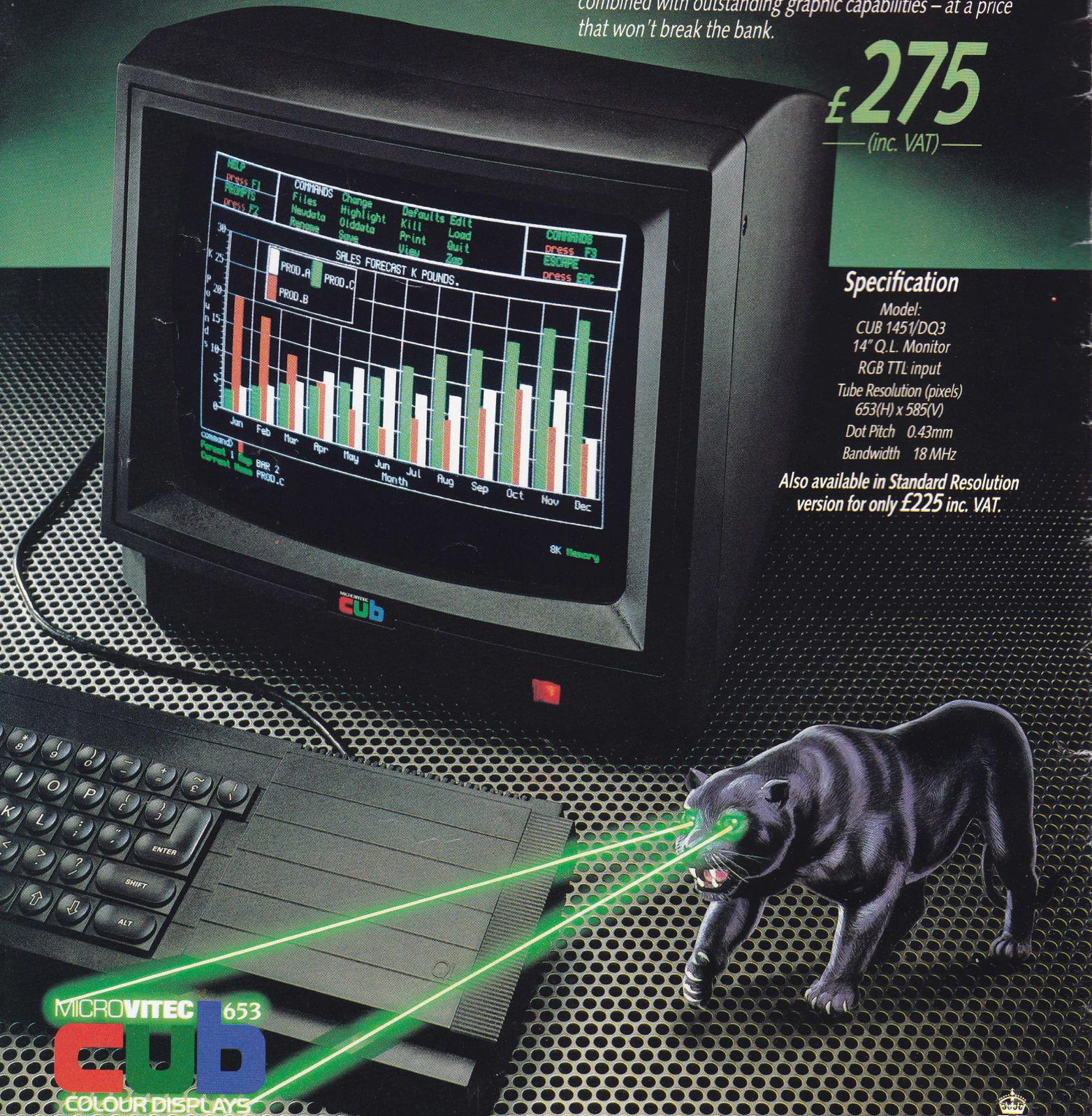
This sleek, black Microvitec CUB medium resolution colour monitor has been designed to be totally compatible with the Sinclair Q.L. An ability to display 85 column text is combined with outstanding graphic capabilities – at a price that won't break the bank.

£275  
(inc. VAT)

## Specification

Model:  
CUB 1451/DQ3  
14" Q.L. Monitor  
RGB TTL input  
Tube Resolution (pixels)  
653(H) x 585(V)  
Dot Pitch 0.43mm  
Bandwidth 18 MHz

Also available in Standard Resolution  
version for only £225 inc. VAT.



MICROVITEC 653  
**CUB**  
COLOUR DISPLAYS

Microvitec PLC, Futures Way, Bolling Road, Bradford BD4 7TU, West Yorkshire. Tel: (0274) 390011. Telex: 517717

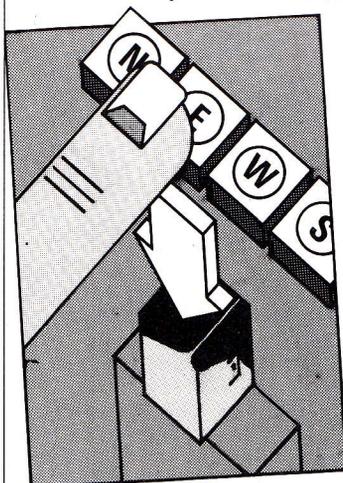


# QL USER

## CONTENTS

**2**  
**Editorial** The boss sounds off about Sinclair's marketing policies.

**3**  
**News** Not a lot of it around in the QL world right now, but what there is you'll find here!



**6**  
**How many bits in your micro?**  
When is a 32-bit micro not a 32-bit micro – and does it really matter?

**9**  
**Letters** Our readers in their own write.

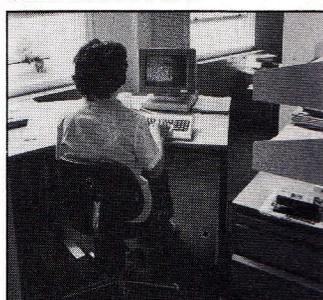
**10**



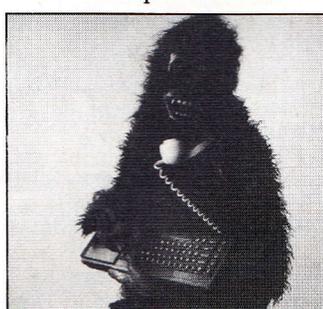
**NEW SERIES: Programming in C** Peter Rodwell kicks off a new teach-yourself series on the C programming language, promised for the QL.

**14**  
**SuperBasic course**  
Meanwhile, Adam Denning continues his tutorial on SuperBasic, the language which comes with the QL.

**17**  
**A peek at Psion** Maggie Burton profiles the company which wrote the QL's bundled software.



**24**  
**QL review** We put an early production machine under the microscope.

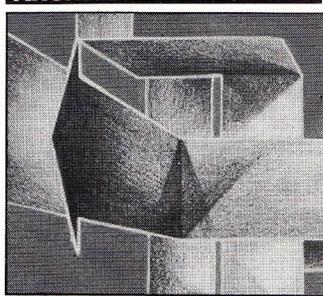


**33**  
**That software reviewed** The three main QL packages come under close scrutiny from our team of reviewers: Quill, Abacus and Archive.

**34**



**39**



**44**

**Editor**  
Peter Rodwell  
**Art director**  
Paul Carpenter  
**Contributors to this issue**  
Maggie Burton, Adam Denning, Duncan Doenitz, Gareth Jefferson, Barry Miles, Sid Smith.

**Cover photography**  
Chris Stevens

**Advertising**  
Philip Baker  
**Publisher**  
Alfred Rolington



Scriptor Court  
155 Farringdon Road  
London EC1R 3AD  
Telex: 32157 EMAPPB.G  
QL User is published and distributed by EMAP Business and Computer Publications Ltd  
Typesetting by Crawley Composition Ltd  
Printed by Riverside Press  
© Copyright QL User 1984

**Editorial: 01-370 2573**

**Advertising: 01-278 46995/8**

## Have We All Been Conned?

Somehow we all hoped that this time it would be different. Surely, we thought, having messed up the launches of his previous two machines, Clive Sinclair would take care to get it right this time. Particularly as he was going after the more serious business buyer.

But no. Once again we have the old saga of a machine being launched *and advertised as available* long before the product is in a fit state for selling. Sinclair, it seems, takes a long time to learn his lessons.

Let's get a few things straight, though, rather than condemn the whole business out of hand.

If you're developing a new computer, you have to plan ahead. You have to buy allocations of chips from the manufacturers many months ahead. If you don't own your own factory, you have to book production time in somebody else's, again months ahead. And of course you won't sell anything unless you tell people about it, which means booking advertising space in magazines and newspapers.

It's this last aspect which causes the most resentment when things go wrong, because that's what the public sees. Now most people are unaware that a typical micro magazine has a lead time of about two months. This means that if you launch your product today and buy lots of advertising space, nobody will know about it for another two months. The art is in choosing your launch date so that by the time the news and the adverts hit the streets, production is just under way. In turn, this means that by launch date, you'll only have a few prototypes - you can't have a factory churning out thousands of machines during those two months.

This is the bit which Sinclair consistently gets wrong. With the ZX81, the Spectrum and the QL, he has managed to launch and book advertising space long before the product was ready. The result has been chaos each time, with thousands of people sending off money in response to those infamous 'delivery within 28 days' adverts and then waiting for months before the goods turned up.

Of course Sinclair is by no means the



only company to behave in this way. Acorn kept some people waiting for 11 months for the BBC Micro and there have been plenty of other, less spectacular cases in the last few years. But Sinclair seems to be more public than most companies and to have done it more often.

There can be little doubt, though, that Sinclair knew full well before QL launch that the product would not be ready in time. It seems reasonable to conjecture that the machine was launched some six months before it should have been. But why?

One obvious answer is that it was a swipe at Acorn, a company which has irritated Sinclair (and a lot of other people) by getting rich with little effort on the BBC Micro. The QL launch was clearly timed to strike hard at Acorn's chances of renewing its BBC contract - and the QL price is clearly no coincidence either when you take into account the BBC factor.

In fact the QL's price is very interesting indeed. We have been reliably

informed by production engineers who have studied the machine that Sinclair could be paying a factory price as high as £340 on each unit. If true, then the £59 margin gives little or no room for profit. This contrasts wildly with the Spectrum (estimated cost £15 for the 48k model) and with general industry practice, in which a business machine typically retails for about three times its factory price.

Meanwhile, Sir Clive Sinclair himself has kept a remarkably low profile on the QL saga. People close to him say he's no longer interested in the computer side and prefers to concentrate on his pet projects, the electric car, the flat-screen TV and the Metalab.

The flat-screen TV was launched a long time ago and has never seen the light of day. With monochrome 80 column x 25 line liquid crystal displays now appearing on production machines and colour versions likely within the next 18 months, the flat-screen concept seems a dead duck for the computer business. And the Japanese have developed liquid crystal TV screens...

The electric car is more promising, even though General Motors abandoned the concept years ago as impractical after spending \$110 million on development. Sinclair seems pretty determined to go ahead with it: one of our spies recently reported seeing a £220,000 bodywork mould for the vehicle being made in Portugal.

The point of all this is that, with these and other ambitious projects in the pipeline, Sinclair will need its computer business to provide finance. In turn, QL customers will need to be treated rather better than they have been so far if they are not to give up in disgust and buy other machines.

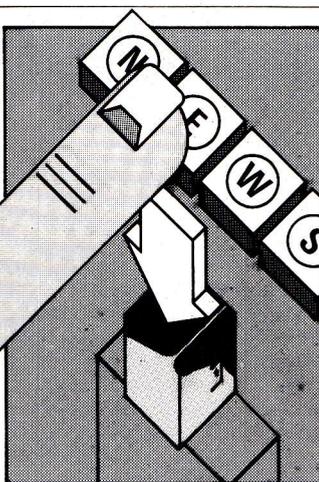
So have we been conned? On the whole, no. But people who in good faith part with their money and then wait months, only to receive a bug-ridden machine with a bodge on the back and semi-functioning software could be forgiven for feeling that they have.

Over to you, Sinclair...

Peter Rodwell

A run-down of new products, people and events in the QL market.

Alex White



## The rivals suffer

Sinclair may not be having the best of times getting its act together with the QL. What's certain, however, is that its rivals are having a much worse time of it. Two of its UK rivals, Computers and Dragon Data, are the latest companies to suffer financially at the hands of the UK marketplace.

Computers, which manufactures the Lynx, has creditors meeting in an effort to find a suitable formula for keeping the company afloat. At the time of going to press the company had not ceased trading. Dragon Data in Wales has gone into receivership, with the directors and shareholders hoping that a new buyer can be found for the company. There's talk that this could be Tandy, whose Color Computer was the model for the Dragon 32. Dragon made a name for itself with the 32 but has never quite managed to translate that into massive sales.

The Lynx, which comes in 48k, 96k and 128k versions – the latter recently having been rechristened the Laureate – has been a good seller on the continent, especially in France. It has never managed to stake too much of a claim to the hearts, minds and wallets of computer buyers in this country. The top-of-the-range Laureate, is the new addition to the family, has the type of specification that could make it interesting as a rival to the QL in certain markets.

With 128k of memory, an optional floppy disk drive, the admittedly boring Z80 processor and the admittedly equally boring CP/M operating system, it has the capability of taking a useful slice of the new small business/home user professional market that Sinclair has identified as a major QL target. The Z80 and CP/M may be boring and are certainly old but they

## Alternative OS for QL?

The tangled story of the QL's operating system receives another twist with the announcement that Cambridgeshire-based GST is to offer its K/OS (yes, that's right!) operating system for the machine to end users.

GST was commissioned by Sinclair in early 1983 to write the operating system for a new micro and at the QL launch in January it was GST software which was responsible for the stunning demonstration of multi-tasking and multiple windowing.

But this GST operating system was dropped by Sinclair Research and the QL is now issued with firmware written by the company itself. The QDOS system is generally conceded to be inferior to the GST material, containing (at least in its earliest manifestation) numerous bugs and requiring a higher degree of programming skill from its operator.

The most notable example of this heavier demand on the user is in respect of the multi-tasking windowing facility. The GST operating system contained built-in screen controllers which automatically allocated areas of the display to programs being run simultaneously. Although Sinclair's QDOS does permit multi-windowing, the programmer is responsible for giving each of his programs clear instructions as to which areas of the

screen are available.

However, GST has announced that an enlarged version of K/OS could be made available to QL purchasers who wanted more powerful firmware in their machines. The company coyly described the likely price as 'in the low hundreds of dollars'.

But GST is expecting more takers in the OEM sector (ie, among companies which buy parts supplied by other manufacturers and assemble them into new products). It cites companies making terminals and control equipment as its likeliest first customers.

Among the reasons given for possible reluctance among end users to invest in the K/OS firmware is its lack of a Basic. The 32k operating system leaves little room for an on-board command language like Basic and GST is reluctant to invest in more software unless convinced of the demand.

However, *QL User* has seen a QL running K/OS and using a command language loaded from Microdrive. Basic, or some other language, could also be plugged into the external ROM socket. And Sinclair Research is known to be planning sales of the QL's advanced main board to the same OEM market as GST's operating system. It's probable that the two products will eventually meet up after all.

*Sid Smith*

bring to the Laureate a wealth of working applications software, something that the new marketplace will need and which the QL arguably does not have.

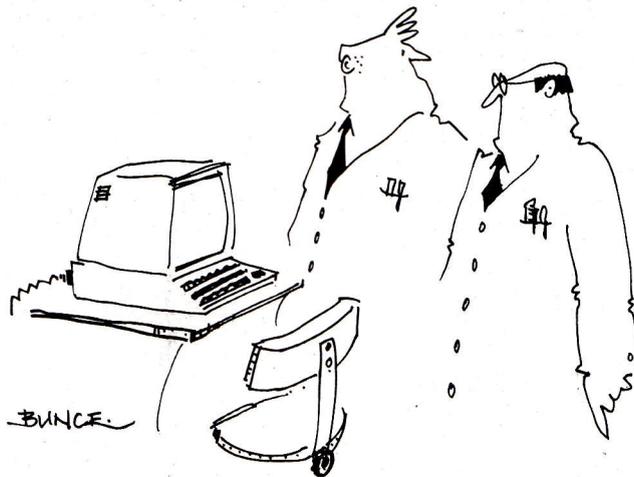
Dragon has succumbed and moved into receivership with a view to locating a new purchaser for the company. It has been refinanced several times already since being saved from the failed Mettoy toy company by Prutech, the high-tech arm of Prudential Insurance.

Although it built up a coterie of dedicated fans, the Dragon 32 never became a massive seller and was, with its 32k memory and Motorola 6809 processor, overtaken by the mainstream of events. The improved version, the Dragon 64, did little to change the company's fortunes.

Ironically, Dragon's failure comes at a time when it could have made a more significant, if temporary, mark on the scene. It had just announced a professional machine, equipped with micro disk drives, and had plans to produce an MSX machine, a move which could have saved it by putting the company back into the mainstream of small computing. The MSX standard is going to be aggressively exploited by its main protagonists, the Japanese manufacturers. These have so far conspicuously failed to take the home computer market by storm, mainly because they have been unable to get any worthwhile systems or applications software on their existing hardware.

The collaboration with US software house Microsoft to produce the MSX standard gives them their best chance. It also gives any other company an equally good chance and the Japanese would have welcomed an indigenous company like Dragon championing their cause for them. In the short term the move would be an advantage for the company, but in the long term, there are many observers who feel it would simply open the door to a flood of Japanese machines which would then wipe out any direct, local rivals in the traditional Japanese way.

*MB*



*Dobson's made the quantum leap – he discovered he could earn more money elsewhere.*

## Put an Apple in your pocket — well, nearly

Portability seems to be the flavour of the month in small computing at the moment, with everybody having a go. Among the latest is Apple, which has taken an old idea, repackaged it in a compact portable form and combined it with a mass of working, proven applications packages.

The old idea is the Apple II, the computer that probably wins the prize as the oldest piece of kit still being sold today. That basic machine, following its reworking two years ago into the Apple IIc, has now been reworked once more with CMOS devices and a lot of repackaging. Now it can be had as a portable machine — the Apple IIc.

Though this might seem to many to be a rather poor scam on Apple's part — putting old technology in a new package and remarketing it — the move looks quite sensible in practice. For one thing, many of the professional and even quasi-professional people that Sinclair is targeting for the QL are going to be interested in the possibilities that come with portability. And with the Apple II there is a fund of directly runnable applications software that is second only to that available under CP/M. Software. For many of those target users, this could be the key to their hearts.

MB

## Clone QL competitor

You may think that cloning is a purely biological trick much loved by science fiction writers and food scientists. In the world of computers, however, it has another meaning: the production of almost endless versions of the IBM Personal Computer, produced by other manufacturers.

These PC clones have been developed in the main to try to take a direct slice of the IBM market from under the Blue Giant's nose. Others, however, have been aimed at

different markets, and one of the latest is aimed at the same market as the Sinclair QL.

This is the Advance Technology 86a, a machine which offers a claimed IBM compatibility with a unit price which will be familiar — £399. Advance Technology has signed to supply the machine, together with its bigger brother the 86b, to WH Smith, which is now selling them through selected stores. This is something of a coup for Advance for it brings it directly into one of the major high street retailers, though Smiths will be starting by selling the machine through selected stores only.

Apart from their IBM compatibility, the two machines are in many completely different. The bigger 86b is much more the direct competitor to the IBM PC itself, with twin disk drives, up to 768k of memory and bundled software from Perfect. The 86a is something different, however.

Both share the same processor, the true 16-bit 8086 from Intel, but the 86a has a standard 128k of memory, expandable up to 256k, a 16k video RAM, 64k of ROM which includes Microsoft's GW Basic, and a separate, good quality keyboard.

It has interfaces for a joystick, light pen and Centronics-compatible printer. It also has a port for an audio cassette. This is the odd quirk about a machine that only needs a disk port and optional disk drive to make it something that could potentially run the QL a very good race. Instead, it has the worst of all possible compromises, a high-powered processor coupled to a lousy storage system.

Though the GW Basic incorporated in the machine has been modified to be compatible with that used in the IBM PC, the fact that it is cassette-based rather than disk-based would seem to negate its original attraction. This omission has been made with, presumably, the objective of protecting the market for the bigger 86b, which retails for a startling price of £1,500, the same total as a fully upgraded 86a would cost. Like the QL itself, however,

it seems as though a good idea has been diminished by the lack of that most obvious utility, access to disk storage.

MB

## HP laps up software

It's amazing just how big the little 'uns are getting and what sort of facilities they can offer. Take the new HP110 from Hewlett-Packard. This was recently pre-announced by the company, together with the suggestion that it will be available in the UK towards the end of this year.

Be that as it may, the package specification shows just what all this technology stuff can accomplish. Starting with a CMOS 8086 processor, which means its power consumption is low, it goes on to include 272k of RAM, 384k of ROM and a flip-up 80 × 16 LCD display. All of this is in a lap-portable machine which runs off batteries.

It's not going to be a cheapo little box, that's for sure. The quoted US price is going to be around \$3000, which means that it's above the majority of QL owners' wallets. That doesn't mean it can be ignored, though. For example, it comes complete with bundled applications software and one of those packages could be of great interest to the type of business user that Sinclair expects to see making up part of the QL market. That package is the now famous Lotus 1-2-3, which integrates a variety of business-oriented tasks into a single program. This is complemented by the Memomaker word-processor

package and HP's own Personal Applications Manager. A significant feature of all this software, coupled with the exploitation of technology, is that they are all mounted in that massive 384k of ROM.

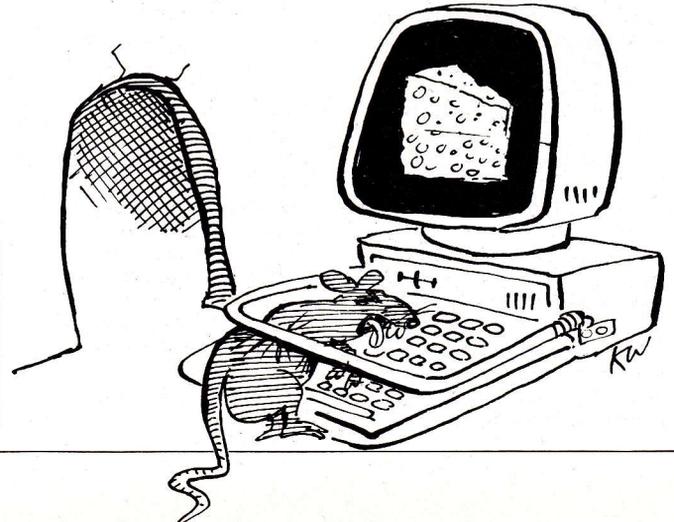
Though the HP110 represents the top of the pile when it comes to lap-portable computers, there are others around that offer similar facilities for less money. There are the existing 'note book' machines such as the Tandy 100 and the Epson HX20, which offer considerable performance for a low price. They lack the type of performance that a QL user could come to expect, though they are gaining a useful niche, especially the Epson, as specialised remote data entry tools for other machines.

Epson has another lap machine, the PX-8, which it has recently announced. This, like the HP machine, makes much of technology's advances in semiconductor technology and, in particular, ROM technology. It also includes bundled applications software mounted in extensive ROM as a standard item. This is a capability that has long been predicted but is now becoming a reality. It will give all small computers both the performance of existing desk-top business machines with considerably easier operation by the user — no swapping disks, cassettes, Microdrive cartridges or anything else to load the applications programs.

MB

## Squeezing it in

Sinclair managing director



Nigel Searle has said that the external EPROM is to be fitted to the first 13,000 QLs to be shipped. These machines will be referred to as Version Zero. Version 1 of the QL will follow, containing EPROMs fitted internally and at some later date these would be replaced by full production ROMs.

At the time of writing, it is not clear how big these internal chips will be. *QL User* understands that the current size of the machine's system software is around 42k. Of this, 32k occupies two 16k internal EPROMs while a further 10k is contained in the external chip.

But 42k is an inconvenient size in computing terms. Sinclair is currently deciding whether to expand this figure to 48k, allowing it to build in extra features but requiring chip sizes of 32k plus 16k. Alternatively, it could try and restrict the ROM size to 40k and thereby require only an 8k chip along with the 32k one.

Sinclair's struggle with ROM sizes has been at the heart of the QL delays. The company originally intended that the QL should have a 32k ROM and commissioned Cambridge systems house GST to write a 21k operating system for the machine. Meanwhile, Sinclair would write its 11k SuperBasic.

Accounts differ over what happened next. According to some sources, the GST operating system was too big or did not fulfil the Sinclair specifications; Sinclair's programmers had to heavily adapt it, thus causing the delays.

However, sources close to GST itself insist that the company fulfilled the specifications laid down by Sinclair both as to size and specification. They claim that the fault lay with the Sinclair SuperBasic, which emerged as double the specified size. Asked for confirmation of GST's innocence, one Sinclair informant said tersely, 'Well, they got paid.'  
*Sid Smith*

## QL modem soon?

The QL modem, promised at the machine's launch as one of the first add-ons to be released, is being

manufactured and designed by OE Ltd of Cumbria.

OE makes the VTX5000 modem for the Spectrum, sold for use with the Micronet 800 database on Prestel. Sinclair is known to have been impressed by the VTX5000, which recently won the British Microcomputing Peripheral of the Year award. MD Nigel Searle is on record as saying that the high specification of the OE modem had prevented Sinclair from releasing its own device for the Spectrum.

The priority given to the QL modem reflects the growing recognition given by micro manufacturers to telecommunications links between computers. Acorn has already released a modem for the BBC Micro and Commodore is planning an independent Prestel-style database.

The Sinclair deal with OE is supposedly confidential. Neither of the two companies would comment about the price or availability of the new device.  
*Sid Smith*

## Osborne encores ... again

Maybe a bit outside the price range of an average QL user (should such a thing exist) but the new Encore from Osborne could be up the street of some of Sir Clive's projected business-oriented marketplace. At around £1,500, perhaps a bit more, the new machine combines the two most sexy elements of personal computing known: portability and IBM PC compatibility.

Ah! Osborne – now there's a name to conjure with. Adam Osborne's company shot from nothing to megabuck corporation status in double-quick time. It repeated the trick, in reverse, last year even quicker. Now it has reappeared in a new, slimmed-down form with a new machine, aptly named Encore.

But this is no second coming for Adam himself, who has left the fold. Instead, the machine is the brainchild of other Osborne people and is being manufactured for it under contract. It sports 128k of



One more time from Osborne.

RAM, CMOS 8086 processor, an 80 character by 16 line LCD display and a single 360k floppy disk as its standard equipment. It comes in an interesting-looking package that is oriented vertically, rather than horizontally.

By going for PC compatibility, Osborne is taking on a number of competitors head-on, including – so rumour has it – the forthcoming new micro from ACT. The blood-letting in this area could be quite dramatic in the not-too-distant future. The potential pickings are, however, large to whoever corners the market. There is going to be a lot of interest in portable PC clones from professional people, especially those with pretensions towards being serious users.  
*MB*

## Commodore now a rival

Having been a head-on competitor of Sinclair's in the home computer market, Commodore has seen Sir Clive Leap Quantumly into the realms of extra-high-tech-computing-at-a-low-price and in the process target the quasi-business professional (and there's a contradiction in terms) as a major potential marketplace for the QL.

The response from Big C has just arrived. Or to be more specific, has been waved around a bit prior to being actually launched. Or not as the case may be. Commodore is exceptionally

good at almost producing a lot of machines it appeared to have launched at one time or another. The rival to the QL, if that's the correct title for it, is the Plus/4, which was one of two machines introduced by the company recently. The second, called the C16, is a games-player primarily, with just 16k of memory. The Plus/4 with 64k, while not over-endowed compared to the QL, is intended for this new target market of the home business person.

This can be deduced from the four applications packages that come bundled in ROM with the machine. These – word processing, database management, spreadsheet and business graphics – are what gives the Plus/4 its name: computer 'plus four', geddit?

With the logic prevalent in the home computer business, the Plus/4 is incompatible with the software already available for Commodore's hugely successful 64 machine. While primarily a games player, this has attracted a number of business applications programmers and, consequently, some of the early members of that target market all manufacturers are identifying. The change will oblige them to throw away their existing software and start again. Alternatively, they could throw it away and buy something else instead, which is no doubt what Sinclair earnestly hopes.  
*MB*

*Sid Smith is news editor of Micronet 800.*

# How Many Bits

*Peter Rodwell ponders the  
significance of Sinclair's  
32-bit QL claim.*

## **We haven't really gained all that much in overall efficiency.**

Did you buy your QL because Sinclair describes it as a '32-bit' computer? Does that make it twice as good as a 16-bit machine and four times better than an 8-bit micro? Does it really matter anyway?

For answers to all these questions, we need to take a look inside a micro and see what goes on in there.

A microcomputer is a computer in which the central processing unit or CPU is built into one single component, a microprocessor chip. Add some memory and some input/output capability, and you have a computer.

Inside the microprocessor there are a number of memory cells called *registers* which are used to hold data being processed by the CPU. The more registers there are, the more data can be stored there, where the CPU can get at it quickly without having to waste time getting it from the main system memory. When work on a piece of data has been completed, the result can be sent out of the CPU chip along the *data bus*, a series of wires which carry the information to a location in main memory. Similarly, when the CPU needs to get information from memory, it sends out a series of signals which cause a copy of the data item to be sent along the data bus and into the CPU chip, where it is placed in a register.

Consider a CPU with one or more 8-bit registers and an 8-bit data bus, as shown in Figure 1. There are four registers, A, B, C, and D, of which A is the main one, used for all arithmetical operations. Usually, this is called the *accumulator*. Suppose we want to

add together two numbers, each represented by eight binary digits, or a single *byte*. We must get one number from memory and store it in the accumulator; then we get the second number and store it in one of the registers, say B. Then we execute an instruction which adds the number in B to that in the accumulator and leaves the result in A. Finally, we send the result out on the data bus, usually to another memory location.

Although this sounds tedious, it works quite well until we want to handle larger numbers than can be represented in a single byte. If we want to add, say, two 16-bit numbers, we find ourselves involved in quite a few memory transfer operations. The obvious solution is to devise a CPU with 16-bit registers, like Figure 2. This actually shows part of the guts of the Z80 processor, the most widely-used 8-bit CPU. As you can see, we still have an 8-bit accumulator but at least we now have some 16-bit registers for storing either several 16-bit numbers or many more 8-bit numbers.

But of course there's a bottleneck; when we want a 16-bit number, we still have to move it to and from memory as two bytes because we still have only an 8-bit data bus. Thus, although we can work more efficiently inside the CPU when we handle large numbers, we haven't really gained all that much in overall efficiency.

## **Getting bigger**

The next step, then, is to make the data bus wider, and, while we're at it, we might as well give the CPU a 16-bit accumulator. But we run

into problems of a different nature. If we have 16 data lines, we end up with an undesirably large CPU chip, for we want another 20 or so lines for the address signals (which tell the memory which location the CPU is reading from or writing to) and we need a number of other control signals too.

There are ways around this, though. For instance, the 8086 CPU chip is built with a combined data/address bus and with 16-bit registers inside. This sort of design requires memory to be '16-bits wide', though, and it needs additional circuitry to separate the data and address signals. The end result - at the time the 8086 first appeared - was an expensive system.

To overcome this problem, a 'sawn-off' version of the 8086 was developed, the 8088. This is internally virtually identical to the 8086 but has only an 8-bit data bus. This may seem somewhat retrogressive, but the theory was that what was lost through spending more time shuffling data to and from the CPU would be made up by increased power within the CPU itself. The 8088 was quickly adopted in the IBM PC and the Sirius, and because it has a 16-bit CPU, these machines were described as 16-bit computers.

The swings-and-roundabouts theory didn't work out, though. *Byte* magazine conducted some extensive benchmark comparisons of the 8088 and 8086 and found that the '88 was between 10 and 40 percent slower than the '86. A debate then arose as to whether or not the 8088 could truly be called a 16-bit processor.

A similar situation now exists

# S in Your Micro?

---

---

## The majority of users will never notice the difference.

---

---

over the 68008 chip in the QL. The 68008 is a derivative of the powerful Motorola 68000 chip, which has 32-bit registers internally but a 16-bit data bus. The 68008 is internally the same as the 68000 but has an 8-bit data bus, producing the same bottleneck effect as the 8088.

In this case, should the QL rightly be called a 32-bit computer? And in fact should any machine using a 68000 family processor be so designated? If not, then how do we describe these machines?

Clearly, it is no longer accurate to talk in terms either of accumulator width or data bus width alone. I feel that a new description of these CPUs is called for, which shows clearly what the situation is. Currently, the only method which seems to make sense is to quote both figures, in the form accumulator/data bus. This would give us the following table for the popular processors:

Z80	8/8
8088	16/8
8086	16/16
68008	32/8
68000	32/16

This nomenclature is pretty clumsy, of course, but so far I have yet to come across some other, neater and more meaningful system.

---

### Does it matter anyway?

---

The obvious question to ask is whether the user will notice the effects of having a wider CPU. As most microcomputers spend nearly all of their time sitting waiting for

something to be typed in at the keyboard, the answer must be that the majority of users will never notice the difference. After all, most people buy computers for their functionality rather than for technical considerations. Processing speed, for the majority of current applications, is quite irrelevant as far as most users are concerned.

But there's another side to the coin, one which is rarely considered. While a processor like the 68008 may not provide massive speed improvements over an 8-bit machine (and to be fair, it does provide some improvement), the real advantage of the bigger processors is in the amount of memory which can be attached to them.

With an 8/8 processor like the Z80, a maximum of 64 kbytes of memory could be used, unless additional and complex circuitry was employed. The bigger processors can, in some cases, handle several megabytes of memory.

Now a popular theme of some 'old hand' programmers is to ramble on at length about how, back in the 'good old days', they wrote an entire payroll program on some ancient beast which took up only 1k of memory; today, they assert, we're just spoilt by all those kilobytes of RAM.

What they overlook, of course, is that the computers they worked with were big and expensive and there was no question of one being provided for the personal use of everybody in a large company. Those computers required a team of highly-trained specialists both to program and to operate them.

Today, things are different. Personal computers are cheap and

plentiful and are appearing on desk tops throughout industry, commerce and education. But these new machines are ceasing to be computers - they're becoming items of office equipment, business tools, in fact, much like typewriters and photocopiers.

Now you don't need a degree course to operate a photocopier because the manufacturers have made them simple to operate - just show someone where to put the original and which button to press and they're away.

Thus it is with computers, too. People neither want nor should be required to know anything about computers in order to use one as part of their daily work, just as you don't necessarily need to know how a TV works to watch *Crossroads*. The old idea of computer literacy for all is in fact spurious - computers must be made 'people literate' instead.

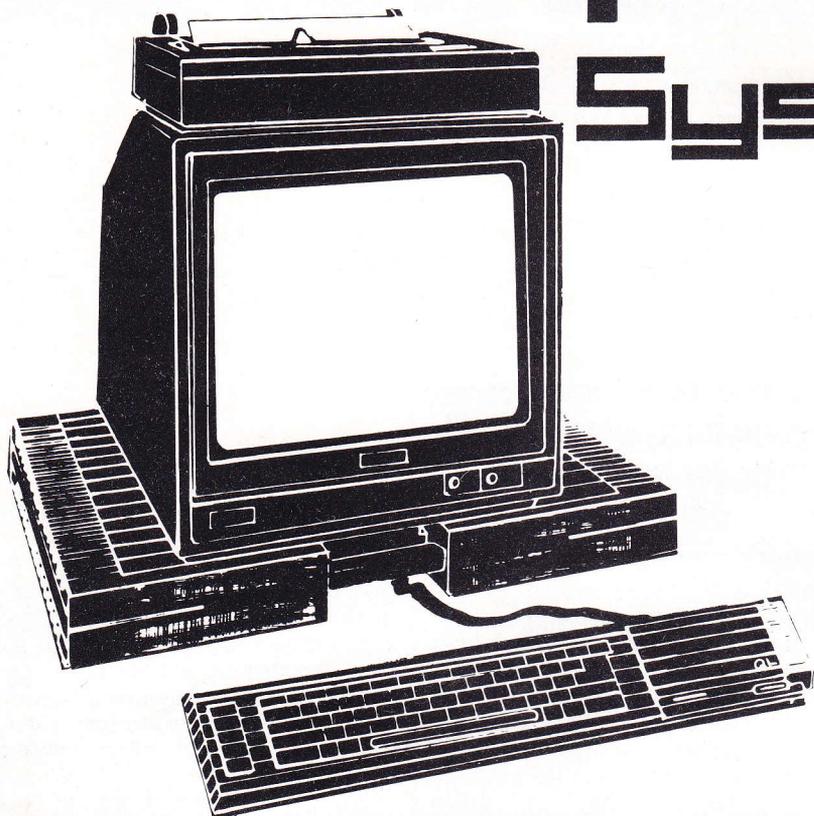
In turn this means that much more effort has to be put into making the machines 'user friendly' and this requires massive amounts of memory. User interfaces such as windows, colour, and high-resolution graphics are all memory-intensive and could not take place without, firstly, cheaper memory chips and secondly, processors which can handle the extra memory.

Enter the bigger processors. If and when Sinclair produces the promised 512k RAM expansion, you'll see some really slick and smart programs on the QL. Not that the existing software isn't smart already, but we have a long, long way to go yet and on the way we'll be gobbling up all the memory we can get...

# NEW - 4 THE QL

## Systems 4

### YOUR QL

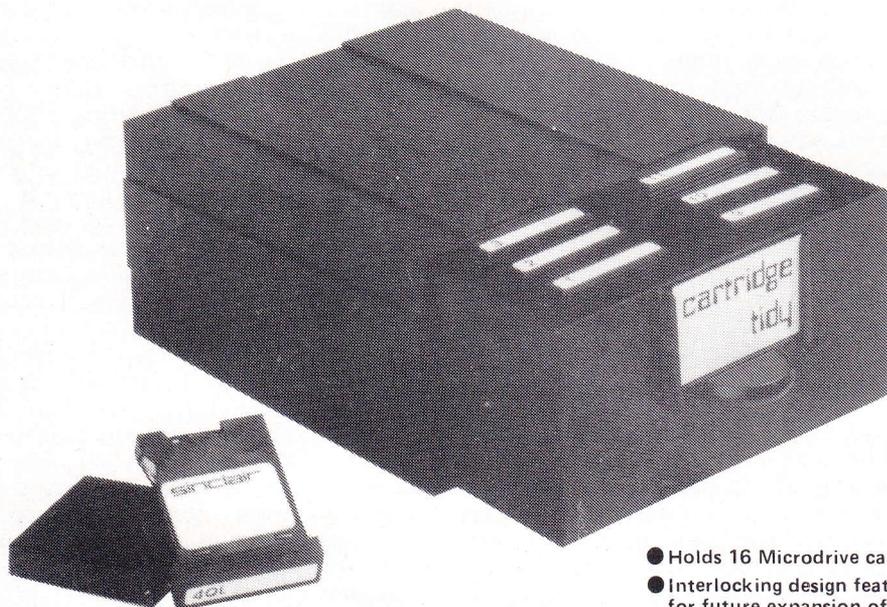


SOLE U.K. DISTRIBUTORS

Exclusive systems  
for your computer.

Send for details.

## NEW - CARTRIDGE TIDY



- 4 Printer
- 4 Monitors
- 4 Terminal
- 4 Cartridge Tidy
- 4 Software
- 4 Link
- 4 Modem

EXCLUSIVE SYSTEMS  
4 YOUR  
**sinclair**<sup>®</sup>

- Holds 16 Microdrive cartridges
- Interlocking design feature allows for future expansion of your filing system

## 4 Systems

Please send me . . . . . (Qty) Cartridge Tidy's,  ZX  QL,  Mr/Mrs/Miss \_\_\_\_\_

at £6.99 each, inclusive. \_\_\_\_\_

I enclose a cheque for £ . . . . . \_\_\_\_\_

Make cheques payable to: \_\_\_\_\_

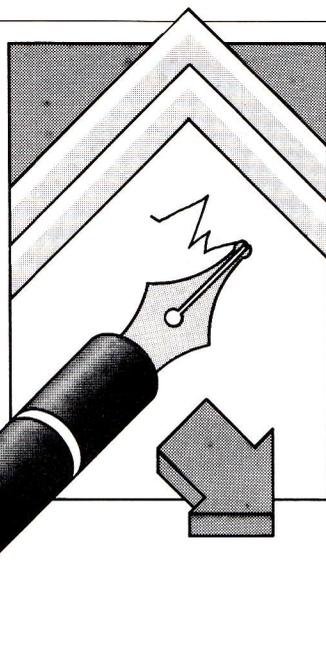
Quantum Leap Systems, 4QL House, 68 Foxwood Close, Feltham, \_\_\_\_\_

Middx. TW13 7DL Tel: 01-844 1399 \_\_\_\_\_

Sinclair is a registered trade mark of Sinclair Research Ltd. \_\_\_\_\_

# LETTERS

We welcome letters from our readers but do try to keep them short and to the point – long, boring letters will be cut! Send letters to: The Editor, QL User, Scriptor Court, 155 Farringdon Road, London EC1R 3AD.



## To buy or not to buy

Currently I own a Dragon 32, tape recorder, joysticks and a Shiwa CP80 dot matrix printer, all of which I am very happy with. I am now very tempted by the QL but before forking out £400-plus, I would like to clear up a few doubts.

There is always speculation with new micros regarding teething problems and these apply to the QL so I am not sure whether to wait until the QL is established and readily available or just to go ahead and order now. My main fear is that after I have invested £400, Sinclair will decide to iron out any problems.

The Shiwa printer has a parallel interface but the QL doesn't have a Centronics port. Will it be better to buy the add-on parallel port for the QL when it's available or to buy the optional RS232 interface for the Shiwa? Also, where would I obtain the appropriate lead to match the QL?  
K G Sime, Nuneaton

*A careful read of this issue will reveal that there are problems with the QL. Clearly these will in due course be cleaned up and the machine will then live up to its excellent spec – but we still don't know when this will be. Whether or not you should buy now or wait for the final version depends on, firstly, whether or not the problems in current machines affect the sort of work you want to do, and, secondly, on how*

*Sinclair proposes to upgrade machines sold before the final version becomes available, something else which we don't know at the moment.*

*On balance, we'd suggest that unless you're desperate for a QL now, you'd be better off waiting for the finalised, full production versions; we hope that these are only a short time away but there's no guarantee of that!*

*Regarding the printer, it's probably easier to convert the printer to RS232 – again, we just don't know when the parallel interface will be available. See our comments to the letter below regarding leads; hopefully Sinclair will offer a proper RS232 lead some time... Ed.*

## QL ousts NewBrain

I am thinking of buying a QL because it would be a considerable upgrade on my present machine – a NewBrain. I find I need considerably more memory and it would seem better to buy a QL rather than spend the money on extra memory for the NewBrain.

One problem which concerns me is how to transfer the large amount of data currently stored on tape. This is stored in the form of string arrays. Is there any way of transferring it without retyping it all?  
I M Kennedy, Northampton

*As both machines are equipped with RS232 interfaces, the easy way (in theory, at least) to transfer it would be electronically. You*

*will need to write two programs, one on the NewBrain to send the data, an element at a time, out through the RS232 port, and one on the QL to receive the data and put it back into an array in the same order. It could then be saved onto a Microdrive cartridge.*

*You will also have to make up an RS232 lead, as both machines have non-standard connectors. The NewBrain connectors are sold by Kuma Computers Ltd, Unit 12, Horseshoe Park, Horseshoe Road, Pangbourne, Berks RG8 7JW and cost around £8 each. We haven't yet found a source for QL connectors, but we'll keep you posted. Ed.*

## What about some games?

I have never had a computer but am thinking of buying a QL. Please could you tell me if there is, or will be, as wide a range of software as for the ZX Spectrum? I am asking you this as I am told there are no games for the QL as it is solely a business computer.  
Sunil Mehra, London NW2

*I don't think the QL will be regarded as a business-only machine, so I'm sure we'll see plenty of games available for it in due course. But software houses can only write games when they've got their hands on the bug-free version of the machine, and at the time of writing, few had. Hopefully, though, once the machine is around in larger quantities, we'll see some really good games, ones which exploit the*

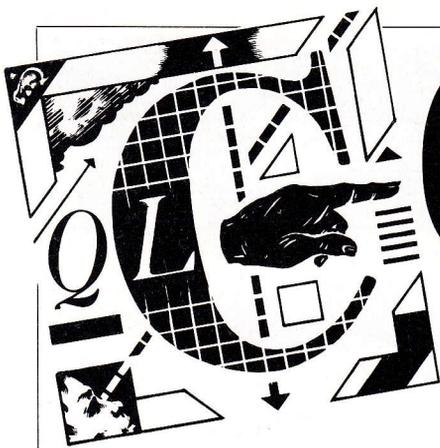
*QL's very superior facilities. Ed.*

## QL for BBC?

I have been told that the QL might become the new BBC computer at some time. As I was thinking of buying a BBC Micro, would it be better to wait and get a QL instead?  
J Williams, London NW11

*Obviously Sinclair hoped that, by launching the QL so early and pricing it at £399 it would get the BBC contract, which is up for renewal this summer. But hoping is a long way from getting and there seems little doubt now that the BBC will renew its deal with Acorn, especially considering the vast number of machines now sold and the large amount of BBC software available. If Sinclair did ever have a tie-up with the BBC, it would probably be as a second 'official' machine, rather than as an exclusive – but this is pure speculation and the BBC just doesn't want to comment right now.*

*In any case, you don't say why the BBC tag is so important. Of course, any machine with BBC on it will sell well (because it effectively gets free advertising on the BBC) and there will be plenty of software. But there are other considerations to be taken into account when you buy a computer: what you want to do with it is the most important; value for money counts too, and the QL undoubtedly beats the BBC machine in this respect, or will do when it's debugged. Ed.*



# C SERIES

---

*Peter Rodwell launches a  
major new QL User series on  
the C programming  
language.*

---

Remember those glossy brochures which Sinclair produced at the launch of the QL? The ones containing full-colour photos and long, eulogistic paragraphs about the glories of the QL? The ones with the now infamous '28 days' order form?

If you sent off your money there and then, you're probably intimately familiar with that brochure as it's the nearest most people were able to get to a QL for several months. Even now, we suspect, there are plenty of people around wondering if '28 days' really meant '28 weeks'.

The brochure featured, among other things, a list of hardware and software enhancements which, Sinclair claimed, were on their way to make your QL even better. Now we could be forgiven for taking that list with a shovelful of salt in view of Sinclair's performance – or, rather, lack of it – since then. But on that list was one very interesting item – a compiler for the C programming language.

Now many people would immediately ask why, when the QL comes with a very comprehensive programming language – SuperBasic – built into (mostly) the machine, anyone would want to pay more for another language.

To understand why SuperBasic isn't necessarily the answer to everyone's prayers, we need to understand a little about the relationship of programming languages to computers and how it is that there are now several dozen programming languages in use.

## **Programming languages**

Computers work by manipulating a series of electric signals. To all intents and purposes, these can be represented as a series of 1s and 0s. Now you'll readily appreciate that something like 01100110 might mean a lot to a computer but

nothing whatsoever to a human. Yet early computers required humans to write programs in just this form; a difficult, tedious and very error-prone task.

It was quickly realised that a more efficient way of programming could be devised; this took the form of a very simple language, which was intelligible to humans but of course was meaningless to the computer. The trick which made this useful was a special computer program which 'translated' or 'compiled' this language into the computer-understandable 1s and 0s. Programming languages were born.

Early programming languages were pretty crude; they involved a set of very cryptic commands and their effective use required an intimate understanding of the computer's innards. This in turn meant that a program written for one computer could not be used on another, different machine unless it was completely re-written for that second computer.

So a further, 'higher level' of languages was devised, which at least in theory allowed a programmer to write independently of the computer's hardware. And a series of different languages was devised, each designed for a particular type of application: for general commercial work, for industrial and military control, and for writing the language 'compilers' themselves. Basic was originally invented as an easy-to-learn, general-purpose language which would allow people like engineers and scientists to get useful results from a computer without first undertaking the lengthy programming training that some other languages require.

## **The problem with Basic**

Generally speaking, Basic fulfils its intended purpose quite well. There's no doubt that millions of

people are now writing computer programs in Basic for fun or serious work who otherwise might not have been able to get to grips with computers at all. Basic has made the computer accessible to these people and it has become *the* language of the microcomputer world.

However, while Basic has certainly been good to the micro industry, the industry has not been very good to Basic. One problem is that the original Basic was fairly crude in some areas. Specifically, it contained no provision for handling the features we now expect on our micros – sound, colour, graphics, etc. Most manufacturers solved this problem by adding their own extensions to the basic language and this has led to a considerable problem: there are now lots of different versions of Basic around. While they all share a common core, they are usually completely incompatible in every other respect, so that there's little or no guarantee that a Basic program written on one machine will work on another make of computer without extensive alterations.

The obvious answer would be to standardise Basic in some way. In fact the American National Standards Institute has produced a powerful standard Basic – ANSI Basic – but few manufacturers have bothered to implement it. Microsoft, a well-known US software house, has tried to introduce a standard Basic called MSX Basic but this has severe drawbacks, not the least of which is a requirement that a computer must contain certain specific components to run it. Virtually the only people to adopt the MSX standard are the Japanese, as others seem to see little use for a standard which restricts hardware development in such a fast-moving industry.

Sinclair's Basics have always been very non-standard, but the company takes the attitude that if you sell enough machines then you set some kind of a standard. This is

rather debatable, some think.

Another problem with Basic is that it is an *interpreted* language. Inside your QL is a large amount of software devoted to translating each line of a SuperBasic program into the 1s and 0s that the machine can 'understand'. Each time you run a program, this translation process must take place, which slows down the computer considerably.

Most other languages are *compiled* - you write your program (usually with a word processor or text editor) and then a special piece of software called a compiler turns it into 'machine code'. Each time you run your program, this machine code is executed directly, with no intermediate translation process, resulting in faster, more efficient programs.

The Penalty is paid during the development stage, as writing, compiling, testing, re-writing, re-compiling, etc can be a tedious business. With a Basic interpreter, you have a high degree of *interactivity* which allows you to write a section and test it without any compilation process. You can, incidentally, buy Basic compilers for some machines (but not yet for the QL) which give you the best of both worlds - you write the program using the interpreter and once it's finished and thoroughly tested, you compile it. For reasons which we'll look at as we discuss C, this isn't a widely-used way of writing commercial software.

## The structuring debate

If you are programming for serious reasons (as opposed to personal amusement), there are several major considerations to be borne in mind.

You need to be able to write a program as quickly as possible and you need to be able to test it and to find and correct any errors with the minimum possible time and effort. Unless you are writing a one-off program, you probably want it to work on a wide variety of different computers with the minimum possible adaptation - it must be 'portable'. And you need to be able to return to the program months or even years later to amend it and be able to find your way about it easily. Ideally, the program should be 'self-documenting' so that merely by looking at it you know what's happening. A family of programming languages has evolved which allows all of this. They are called structured languages and C is one of them; Basic isn't.

To give you an idea of what I mean, here are two fragments of a program, one in Basic and the



Illustrations by Will Hill

other in a hypothetical structured language. Let us suppose that we want to multiply a variable called X by another variable called Y and put the result back into X; we want to carry out this operation several times, and print the value of X each time. Suppose also that the multiplication and the printing are both carried out by subroutines which exist elsewhere in the program. Here's what the Basic program might look like:

```
100 X=50
110 Y=2
120 GOSUB 5000
130 GOSUB 8050
140 X=27
150 Y=2.7
160 GOSUB 5000
170 GOSUB 8050
etc
```

The problem here is that there is no indication as to what the two subroutines are doing. If it's a really long program, we have to ferret around trying to find the subroutines and figure out what each one does before we can understand what's going on. You could of course include plenty of REM statements to remind you what's happening but these take up memory

space and are tedious to write. Compare it with this structured example:

```
x=multiply(50,2)
print(x)
x=multiply(27,2.7)
print(x)
```

It is perfectly clear from this what is going on. We have two subroutines, 'multiply' and 'print', whose names tell us precisely what they do. When we call each subroutine, we give it the information it requires - in this case by enclosing numbers or variable names in brackets - and in the case of 'multiply', the subroutine returns the result which we can place straight into the variable. Study this example carefully and make sure you understand what's going on - you are looking at what is almost a piece of a program written in C.

But there's a lot more to structured languages than just giving subroutines meaningful names instead of line numbers. When you write with a structured language you break the task down into its component parts and analyse them carefully. Wherever you find an operation being carried out more

**YOU CAN DO ALL THIS ON AN  
APPLE IIe  
PERSONAL COMPUTER?**



- Word processing • Financial models • Budget planning
- Graphics presentations • Data base file management
- Accounting • Sales forecasting • Electronic mail

**UNBEATABLE PRICES?  
TRY US!**

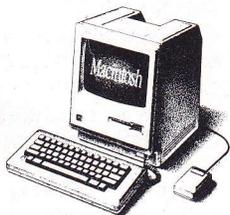
★ **The Apple IIc** ★

**A new portable, compact micro**



Authorised Dealer

**Chaotic Desk?  
Macintosh  
is YOUR answer!**



**The complete Desktop Manager on a screen.  
Call 0942-892818 NOW for full details  
on Macintosh and all the  
Apple family and peripherals.**

**MICRO COMPUTER  
CONSULTANTS LTD**

Ascott House, 227 Elliott Street,  
Tyldesley, Manchester  
M29 8DG.

Tel: 0942-892818

**WE SPECIALISE  
IN OVERSEAS  
ORDERS**

# Computer Enterprises



## PERSONAL COMPUTERS

IBM PC, 64K, 2 DS drives + mono + print ad. + DOS	£1950
IBM PC, 64K, 2 DS drives + colour + print ad. + DOS	£3195
IBM XT, 128K, DS drive + mono + print ad. + DOS	£3689
IBM XT, 128K, DS drive + colour + print ad. + DOS	£3995
IBM Portable	£C ALL
APRICOT, double SS drives + 256K + all free software + monitor	£1349
APRICOT, double DD drives + 256K + all free software + monitor	£1496
APRICOT Xi, 5MB + 256K + mon + SD + all free software	£2029
APRICOT Xi, 10MB + 256K + mon + MD + all free software	£2249
SIRIUS, 1.2MB drives, 128K	£1759
SIRIUS, 2.4MB drives, 256K	£2349
SIRIUS, 256K + 10MB + DS drive	£2996
COMP AQ, 256K RAM, 2 drives + MSDOS	£1995
COMP AQ, 256K RAM, 10MB hard disk + drive	£3795
CHAMELEON, 9" screen, twin drives, £1500 worth software	£1995
OLIVETTI M24, 128K, double drives	£1850
OLIVETTI M24, 128K, 10MB hard disk	£3595
SANYO 555, includes £1000 worth software	£2849
EPSON QX10, 192K RAM, dual DD, free software	£1495
EPSON PX-8	£799
HYPERION, 256K RAM disk, 2 drives	£1995
MACINTOSH Basic + Macwrite + Macpaint	£1699
APPLE IIe HOME PACK: DD controller, TV mod.	£490
APPLE IIc, 128K RAM, internal disk drive	£799
KAYPRO 10 with 10MB HD and free software	£2395
HEWLETT PACKARD HP150	£2395
TELEVIDEO TELE-PC 1605 128K, double disk, mono	£1995
NEC APC, mono, dual drive, 128K, 2MB, 8" disks	£1890
NEC APC (same as above but colour)	£2295
NEC PC-8000, monitor, printer, CPM, 8" drives	£3995
NEC PC-8000, colour, printer, 2 drives, CPM	£995

(Please ring for other NEC combinations)

## HOBBY MICROCOMPUTERS

SPECTRUM 48K	£399
SINCLAIR QL	£339
BBC model B	£349
COMMODORE 64	£159
ACORN Electron	£169
ORIC Atmos	£139
ATARI	£99

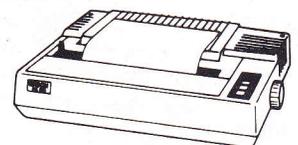
Please ring for software and add-ons

## PRINTERS

	Sale price	RRP
DOT MATRIX		
Canon PW1080A (NLQ)	£289	£329
Canon PW1156A (NLQ)	£359	£499
Canon PJ1080A (ink jet)	£369	£433
Kaga Taxan KP10 (NLQ)	£239	£299
Kaga Taxan KP910 (NLQ)	£349	£395
Epson RX80T	£199	£249
Epson RX80FT	£219	£285
Epson FX80	£329	£438
Epson RX100FT	£339	£450
Epson MX100FT	£356	£475
Epson FX100FT	£456	£569
Epson LQ1500	£179	£299
Shinwa CTI-CP80	£179	£299
Shinwa 40 (colour)	£115	£225
Prism 80S	£895	£1100
Prism 132S (colour)	£1195	£1595
Microprism FT (S&P)	£299	£399
Mannesmann Tally MT80	£195	£260
Mannesmann Tally MT160	£479	£599
Mannesmann Tally MT180	£599	£749
MT Pixy Plotter	£479	£599
Seikosha GP700A (colour)	£349	£399
Seikosha GP250X	£199	£235
Seikosha GP550A	£219	£260
Seikosha GP100A	£149	£174
TEC 1550 (parallel)	£459	£550
TEC 1550 (serial)	£489	£600
Microline 82A	£239	£299
Microline 92P	£359	£449
Microline 93P	£468	£595
Microline 83P	£445	£495
Microline 84P	£659	£799
Microline 2410P	£1590	£1985
Microline Tractors	£45	£55
Star Gemini 10X FT (120cps)	£199	£249
Star Delta 10FT (80col/160cps)	£289	£359
Stat Radix 10FT (80col/200cps)	£459	£578
Toshiba (192cps/100cps)	£1249	£1575
Texas Instruments TI 810	£1190	£1435
Texas Instruments TI 85T	£795	£965
Anadex DP9725 (240cps)	£1195	£1347
Anadex WP6000 (330cps)	£1795	£2199
Anadex DP6500 (500cps)	£1990	£2475
DRE-Newbury DRI 8931	£1745	£1890
Hermes 612B	£1690	£2250
Anadex DP9500 (180cps)	£850	£1095
Anadex DP9520 (240cps)	£985	£1175
Siemens ink jet	£499	£599
Diablo colour ink jet	£890	£995

	Sale price	RRP
TERMINALS		
Qume QVT 102	£495	£595
Qume QVT 108	£629	£756
Qume QVT 211GX (Tektronics)	£856	£1195
Qume QVT 103 (DEC VT100)	£729	£910
Kokusai KDS (TV1925, WS)	£449	£595
Kokusai KDS (STD)	£425	£545
Hazeltine Esprit I	£449	£495
Hazeltine Esprit II	£455	£525
Hazeltine Esprit III (TV1950)	£825	£995

	Sale price	RRP
PLottERS		
HP 7470	£795	£893
Watanabe MP1000	£699	£795
Watanabe WX4636	£2565	£2880
MT Pixy Plotter	£475	£599
Act Writer 80	£528	£595
Act Writer 81	£695	£749
Gould Bryans DP7	£1255	£1495
Rowland DXY800	£475	£595



## DAISYWHEEL PRINTERS

	Sale price	RRP
Brother EP22	£139	£169
Brother P44 KSR	£195	£245
Brother HR5	£129	£169
Brother HR1	£446	£595
Brother HR15	£339	£449
Brother HR25	£559	£759
Brother HR35	£695	£925
Juki 6100	£339	£399
QUME 845 FFP	£1495	£1890
QUME 9/55	£1850	£2350
QUME 11/40	£1195	£1400
QUME 11/55	£1259	£1577
NEC 3510 (S or P)	£1390	£1595
NEC 2010 (S or P)	£599	£795
NEC 2050 (for IBM)	£699	£890
Diablo 620 RO	£339	£395
Diablo 630	£1560	£1950
Ricoh 1300	£990	£1295
Ricoh 2300RP	£1400	£1695
Ricoh 2600RP	£1750	£1995
TEC F10-40	£895	£1295
TEC F10-50	£1250	£1695
Smith Corona TP1	£189	£229
Silver Reed EXP (1209)	£295	£329
Olympia ESW103 KSR	£799	£998

## AUTO SHEET FEEDERS

Genesis (for Juki/Tec/Ricoh/Diablo)	£299
Ruhshauer Mechanical	£395
Ruhshauer electro mechanical	£479
Tractor feeders	£159



## MONITOR AND VDU

	Sale price	RRP
MONOCHROME		
Sanyo SM12N (green, 15mhz)	£89	£89
Sanyo DM112 CX (10mhz)	£89	£129
Philips 12" (green)	£79	£95
Kaga 12G (green)	£98	£119
Kaga 12A (amber)	£110	£137
BMC 12" green	£79	£89
BMC 12" high res	£99	£119
Novex (amber)	£99	£125
Yanjen (green/amber/tilt/Sw)	£85	£99
Swivel and tilt monitor stand	£19	£25

	Sale price	RRP
COLOUR		
Kaga K12R1	£199	£225
Kaga K12R1X, RGB/PAL	£239	£295
novex 14" RGB (super res, 800 dot)	£229	£395
Luxor 14" (super res, 800 dot)	£495	£598
Dyneer 14CM, 640 x 200	£399	£450
Dyneer 14CH, 720 x 350	£575	£650
Sanyo CD3125N (360 dot)	£169	£199
Sanyo CD3117M (620 dot)	£295	£369
Sanyo CD3115H (720 dot)	£399	£499
Fidelity CM14, 12mhz, RGB and COMP	£179	£199
Novex high res, 14"	£199	£235

## ACCESS

HOT LINES FOR INFORMATION AND ORDERS

01-543 6866 01-542 4850 TLX: 8813271

SHOWROOM

85-85A QUICKS ROAD, WIMBLEDON, LONDON SW19 1EX

INTERNATIONAL, EDUCATIONAL, DEALER, GOVERNMENTAL,

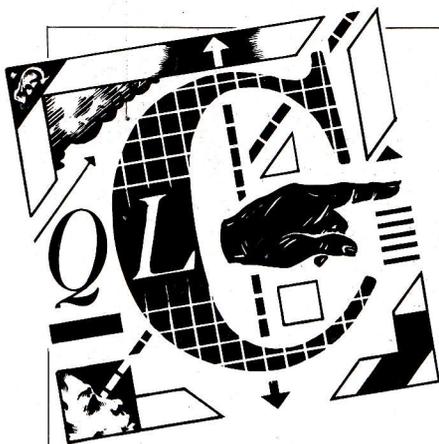
CONSULTANT, LEASE, RENT, PART EXCHANGE ENQUIRIES

WELCOME

All items new and carry manufacturers' guarantee.

Prices are exclusive VAT and delivery





than once, you write a subroutine to do it. Thus, your program is built up with a large number of these blocks, each of which can be thought of as a 'black box' – you put one or more pieces of information into a box and the result is handed back to you.

Once you have made certain that the box is operating correctly, you need no longer concern yourself with what goes on inside it. Your main program thus becomes a progression of subroutine calls with, maybe, just a few pieces of actual program here and there when it's not worth writing a subroutine.

A structured program is a joy both to write and to read, even long after you've forgotten all its intricacies. It is possible to emulate structured programming to a certain degree with SuperBasic but as this series progresses, I think you'll see why many people don't go this route and why structured languages are so enthusiastically promoted by many programmers.

I should at this point make it clear that I have no intention of becoming involved in another 'Basic versus structured languages' debate. The subject has been flogged to death now and there's little point in going over old ground. Some people get very upset about it, which mystifies me as I feel there are far more important things in life to get worked up about than programming languages. So, for the record, I'll state that I consider Basic to be fine in many respects and that the purpose of this series is not to bury Basic but to introduce you to wider programming possibilities; no correspondence on this subject will be entered into!

In this series I am assuming that you have at least a working knowledge of Basic, but not necessarily of SuperBasic. Thus, knowledge of elementary programming concepts such as variables, loops, subroutines, etc, will be assumed. If you find the series heavy going because of this, try reading Adam Denning's SuperBasic course and then come back to this later. C has

been described as a 'sports car' language – you need to know what you're doing to use it well but the results can be exciting. In the next issue of *QL User* we'll get stuck into some real C and you can draw your own conclusions. To finish this introduction, here's the history of C and an idea of where it sits in the hierarchy of programming languages.

## The C story

C owes its origins to a British language called BCPL, which Acorn recently released for the BBC Micro. It was developed in the Bell Telephone laboratories in the USA, as part of a project which also resulted in the development of the Unix operating system.

At that time, minicomputers were only just appearing and they were pretty unfriendly beasts, requiring considerable computing expertise. At Bell, it was decided to devise an operating system which would provide computer programmers with plenty of useful 'tools' for developing software quickly and efficiently. And that's exactly what Unix is – a marvellous programmer's tool, written for minicomputers (usually multi-user systems) and never intended as an applications environment for use by the 'computer-naive' end user.

C developed, apparently, in parallel with the Unix work and Unix itself is now mostly written in C. C thus became intimately associated with Unix and it is only comparatively recently that it has come into widespread use with other computers and operating systems. There is some debate as to whether the next language in the series should be called D or P.

If we consider the computer itself to be at the bottom of a hierarchy with the human programmer at the top, as in Figure 1, we can draw in several layers of programming languages between the two. Just above the hardware is the binary code – the 1s and 0s – which the computer 'understands'. Slightly above this is assembler language, in which each of the binary instructions is replaced by one terse mnemonic phrase. Just underneath the programmer are the high-level programming languages like Basic, which use a sort of abbreviated human language (usually English, although attempts have been made to use other languages). In between the high-level languages and assembler is C.

Assembler language gives you absolute control over the computer, but you must know your computer very well before you can use it. The resulting programs are difficult to read and debug but use memory space very efficiently and execute

very quickly indeed. The major drawback to assembler, apart from the length of time required to write programs with it, is that it is *non-portable*: each processor has a different assembler language and moving a program from one processor to another usually means at the least using a special translator program followed by hand-tuning or at worst a total re-write.

High-level languages require much more memory but are more easily understood, written and debugged. They tend to execute rather more slowly than assembler; as mentioned earlier, Basic is particularly slow because it has to be translated line-by-line as the program is running. As memory prices are falling continuously and processors are becoming more powerful, these drawbacks to high-level languages are becoming less important. However, a high-level language doesn't permit anything like the degree of control which assembler gives.

Most high-level languages are reasonably portable – some more than others – but Basic, as we have seen, is the least portable of them all unless a program is confined to the very common core and makes no use of any computer's extra facilities; this generally makes for exceedingly boring programs indeed.

C occupies a unique position by offering all of the advantages of a high-level language yet allowing very nearly the same degree of control as assembler provides. At the same time it is probably the most portable of the high-level languages.

It would appear that C is the ultimate language, then. Unfortunately this is not so, because different languages are suited to different applications. For instance, Basic is a reasonable compromise; it's easy to learn while still allowing a wide range of general-purpose activities but not quite doing anything outstandingly well. You could write a stock control program in Basic but you couldn't easily write a word processor or an operating system with it. Cobol was devised for general business use, for which it is excellent, but again it's by no means ideal for many other jobs.

C is good for many applications, especially when efficient memory use and high execution speed are important criteria. Generally speaking, a C program will be little larger than, and execute almost as quickly as, an equivalent assembler program. But C is weak in certain areas and it is not the best language to use for, say, a payroll program.

*Next issue: First steps in C*

---

*Adam Denning continues his teach-yourself course on SuperBasic, the QL's built-in programming language.*

---

# Learning SuperBasic

In the last issue we wrote a program to print different messages depending on your age. We found that we had a problem setting limits on the SELECT ranges, so the final program gave a wrong result if an age over 18.9 but less

than exactly 19 was entered.

We promised a way to cure it. The only possible way is to cheat a little and make all ages entered whole numbers. This means that if an age of 18.5 was entered, the program would treat it as 18.

To do this we make use of what are known as *integer variables*, special variables which hold only whole numbers. On the QL, an integer variable can be any whole number between -32768 and 32768. Nobody's likely to be *that* old!

An integer variable is signified on the QL by postfixing the variable name with a percent sign. So, 'age%' is an integer variable holding the age.

We have to alter the program a little to take this into account but we'd still like people to be able to enter any sensible age they wish. In other words, the variable to which the age is input must still be a *floating point* variable. This is a type of variable which can hold any value, whether whole, fractional or a mixture, between limits specified by the machine. On the QL this range is gigantic and quite unlikely to be attained in any calculations we are ever likely to use.

We can now change the program so that it *will* work under all circumstances – as long as a realistic age is input!

First, change line 40 to read:

```
40 SElect ON age%
```

and add a line like this:

```
35 age% = INT(age)
```

Now, by changing every occurrence of 'age' to 'age%' from line 50 on, everything works just as we expected.

Although our algorithm is not quite as elegant as it might be, the program code has been written to accommodate its flaws. As a point of style, it's worth noting that the SElect statement we have used can be made a lot easier to type in. As we have already defined the variable on which we are going to SElect – 'age%' – there's no need for all this

```
ON age% = x TO y
```

stuff. All we need is an equals sign followed by the range, like this:

```
= x TO y
```

or

```
= x
```

---

## **Procedures and functions**

---

Whenever we use certain SuperBasic keywords like PRINT or LIST, we're actually using inbuilt procedures – short sections of program within the SuperBasic software which perform these operations whenever we want to carry them out.

Likewise, things like INT, SIN and RND are functions. It's important to see the difference between functions and procedures.

Both are blocks of program which perform specific acts but procedures differ from functions in that they do not specifically return a result. In other words, you can assign a function to a variable, such as:

```
answer = DEG(PI/2)
```

but it wouldn't make sense in the case of a procedure. You can hardly say:

```
answer = PRINT "Silly"
```

Apart from the inbuilt procedures and functions, SuperBasic lets you add your own. This is called *defining* your own functions, and it's a simple matter.

First, though, we need to discuss *arguments* or *parameters*. A function (and a procedure, too) is generally used with some collection of values which are passed to it, so that it can act on these and produce its effect/result accordingly. The number of these parameters can vary from zero to whatever your machine allows – which isn't defined on the QL, so there is no theoretical limit.

When a procedure or function is defined, the definition includes the parameters: these are called the *formal parameters* – they're there just to tell the computer to expect parameters when the function or procedure is used.

The *actual parameters* which are passed to the function or procedure can be constants, variables or expressions. A constant is simply a number line, like 2, or a string like "A string constant". We already know what a variable is, but we haven't met expressions before. It's simply a mixture of constants and/or variables, such as:

```
a + 4 - RND(2 TO 4)
```

or

```
4 * 2 - 3
```

or

```
RND(a TO 3) - LEN(a$)
```

or

```
"string 1" & "string 2"
```

As you can see, an expression is extremely versatile.

## Defining a procedure

Let's look at a procedure definition so we can see how they are formulated and gain a greater understanding of formal and actual parameters.

Type NEW and press the ENTER key. Now type AUTO. You will be presented with 100 at the bottom of the screen, followed by a flashing cursor. Now we can type program lines directly into the QL; the AUTO facility will automatically give us a new line number each time we have finished typing a line and have pressed ENTER at the end of it.

We can use a shortened form of each keyword as we enter them and the QL will know what we mean and expand them as appropriate. The short form is the part of the keyword in capitals in the KEYWORDS section of the User Guide.

It also makes no difference if we enter the words in lower case ('abcd') or in upper case ('ABCD'). For clarity, we'll show them in upper case here.

```
100 DEF PROC pretty (papcol,
inkcol, number)
110 LOC times
120 INK inkcol : PAPER papcol :
MODE 8
130 CLS
140 FOR times = 1 TO number
150 CIRCLE RND(30 TO 200),
RND(50 TO 150),RND(1 TO 50)
160 PAUSE 25
170 END FOR times
180 END DEF
```

This very simple procedure has just one purpose – it draws a number of circles in a specified colour on a specified background. The first line names the procedure as 'pretty' and defines it as having three formal parameters – 'papcol', 'inkcol' and 'number' – to specify these quantities. When you call the procedure, you don't need to use variables called 'papcol', 'inkcol' and 'number' – they're just used as a way of defining what is to be done with each parameter within the function. So, if you called this procedure with a line like `pretty 2,6,20`

it would draw 20 circles in cyan ink on red paper. The parameters 2, 6 and 20 are the actual parameters.

Now let's look at the body of the procedure and see how it does what it does.

The next line declares a variable called 'times' as being *local* to this procedure. This means that it has no value outside of the procedure – as far as the rest of the program is concerned, it simply doesn't exist and if you use a variable with the same name elsewhere, its value won't be changed by this procedure. It's always best to declare as many of the procedure variables as being *local* as is practical. Although this takes up more memory, it's neater and altogether preferable as it eliminates the chance of a procedure accidentally changing the value of a variable being used by some other part of the program. And memory is hardly at a premium in the QL.

The next line of the procedure sets the ink and paper to the values passed as parameters and then sets low resolution mode to ensure that all colours can be displayed. The screen is then cleared by the next line.

Now we enter what's known as a FOR loop. These loops execute everything between the line containing the FOR and the line containing the END FOR a certain number of times. The number of times round the loop is dictated by the programmer using the FOR control variable – in this case 'times'. We have asked the machine to do the loop starting from 1 and going up to the third parameter, 'number'.

The loop merely consists of drawing a random circle and then waiting for half a second. It will do this 'number' times. We won't go into how the RND function works just yet.

Notice how the end of the procedure definition is marked with a line saying END DEF (which comes out as END DEFine). Without this, the QL wouldn't know where the procedure ended, so it *must* be included!

Now that we have defined a procedure, we can use it at any

# Last Issue's Programs

time just by typing its name. Naturally we must also write the parameters or else the procedure won't work. So,

**pretty 6,0,10**

will draw 10 random circles on the screen in black ink on yellow paper, and

**pretty 0,7,200**

will draw 200 random circles in white on black.

## Defining functions

A function looks very similar to a procedure but it returns a result, defined with the RETURN keyword and followed by the expression to be returned.

Although we haven't introduced number systems yet, *hexadecimal* numbers (ie, numbers based on 16 as opposed to the more usual base 10) are very important in computing. A hexadecimal or hex number consists of a series of digits, from 0 up to a digit signifying the quantity 15. The ordinary numbers, 0, 1, 2 etc, are used until we reach 10 and then we switch to letters, as shown in Figure 1.

Decimal	Hex	Decimal	Hex
0	0	8	8
1	1	9	9
2	2	10	A
3	3	11	B
4	4	12	C
5	5	13	D
6	6	14	E
7	7	15	F

Figure 1

As hex numbers cannot be entered into the QL directly, it's very useful to have a function to convert a hex string into a decimal number. The hex string is the parameter and the decimal number is the result returned by the function.

Note that the hex number *must* be a string, as the QL won't recognise hex digits as being numbers. Here's our conversion function:

```
32000 DEFine FuNction hex(hex$)
32010 LOCAl a, b, dec
32020 dec = 0
32030 FOR a = 1 TO LEN(hex$)
32040 b = (hex$(a) INSTR
      "01234567889ABCDEF") - 1
32050 IF b <> -1
32060 dec = dec * 16 + b
32070 ELSE dec = 0 : EXIT a
32080 END IF
32090 END FOR a
32100 RETurn dec
32110 END DEF
```

What happens here is that we look at each character of the hex string and convert it into its decimal equivalent. The function LEN(hex\$) returns a number equivalent to the number of characters in the string and the function hex\$(a) returns the a'th character of hex\$.

INSTR is rather more complicated. It finds the position of the first string (in this case hex\$(a)) in the second string (a list of all the permitted hex digits). If it isn't found, then INSTR returns zero, so the complicated bit simply returns the position -1 of each character in the hex digit string.

This is made to be -1 if the character is not found and this causes the function to finish prematurely with a result of zero, thereby trapping strings with 'illegal' characters in them.

If the character is valid then 'b' holds the decimal value of the character and this is added to 16 times the accumulated total, 'dec'. Finally, we get the decimal equivalent out of this.

Next month we'll describe some useful functions and procedures and then we'll delve into the world of SuperBasic REPEAT loops.

We received a number of complaints from early QL buyers that the programs published in the first instalment of the SuperBasic course don't work.

The programs were written using the latest possible Sinclair documentation available and conformed to that documentation's description of the way the Basic should work. It does appear, though, that some of the very early machines contain bugs which prevent this from happening.

We have now tested the programs on the latest release of machine and found that they do work properly. If you have one of the first machines, there's not much we can do to help you at the moment. Obviously, we have to conform to the latest version of SuperBasic, but we'll try to include a bug report in the next issue to help early users to run the programs while they're waiting for their upgrades.

# A Peek Inside Psion

*Maggie Burton visits Psion, the company which wrote the QL software packages.*

Psion Limited, (formed in October 1980) shot to almost overnight fame with the classic program Hungry Horace, launched on the ZX Spectrum in 1982. Since then it has produced several more programs for the Spectrum, all known for their quality and inventiveness.

Vu-3D, Vu-file and Vu-calc pushed the Spectrum hard – the way programs should but mostly didn't at the time. Psion, best known among the heavily competitive companies battling it out in the Spectrum arena, was probably instrumental in giving the Spectrum a name for classy software.

Psion is known for the quality of its Spectrum software more than for what it did with the ZX81, not that the ZX81 presented anyone with much of a computing challenge. Versions of some Psion programs for the ZX81 are said to be 'selling well' by managing director David Potter – in particular Vu-file. The Vu-suite can also be found on the BBC computer. But the Spectrum brought Psion the biggest success of all with the Flight Simulator. This program is one of few which will go on selling for as long as the Spectrum exists – a true classic of the games software charts.

A less well-known game has been put out on the BBC computer – Saloon Sally. Another excellent Spectrum program, Computer Scrabble, was based on an original version for the Apple II, written by Little Genius. Psion, it would seem, bought the rights to the Spectrum version from that company and Computer Scrabble (the subject of heavy licensing from Spear and Son, which owns the board game) was considerably improved in conversion for the Spectrum.

The funny thing about Psion is that it normally keeps so quiet. One expects a company so well known to have quite a hefty public

*Window on a cool, collected David Potter.*



image. Instead, the company has previously been known mostly under the Sinclair umbrella – so much so that quiet ties and exclusive agreements between Psion and Sinclair have often been hinted at. Hardly surprising when you look through your back copies of various magazines to find a Psion ad which is separate from Sinclair's.

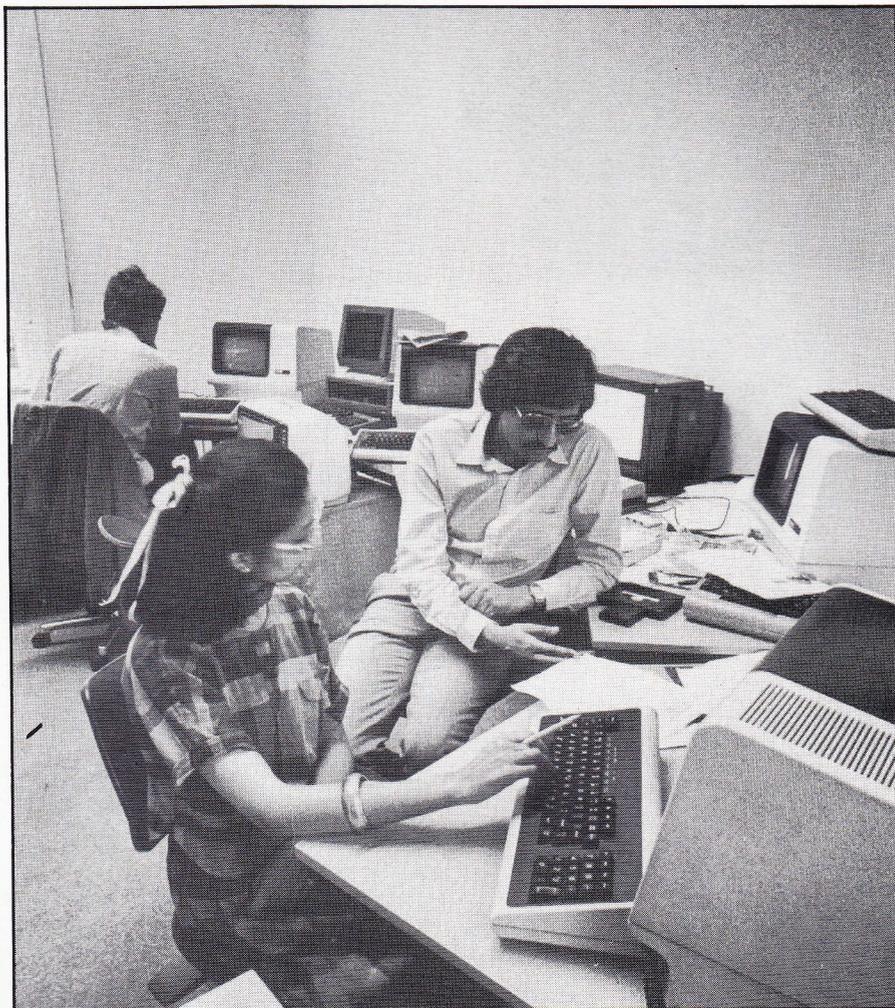
Imagine the surprise when Psion actually leaps out from the bushes and invites members of the press to an at-home with the top brass. The reason, says PR man Robin Kinnear: 'We realise we've been a bit quiet until now, but we've been steadily growing and building ourselves up. We'd like to get to know the press a little better.' The real reason, everyone is sure, is to show off about the QL. In fact the visit turned out to be an ideal opportunity to find out a little more about what makes the company tick.

The first burning question had to be, 'What made Psion choose the Spectrum for its grand plan?'. The answer: forecasting. 'A software house has to look around and then produce for hardware which is going to sell. We saw Sinclair as a big volume seller and that was a correct judgement,' explained Potter. He then goes on to enlighten us as to the relationship between Psion and Sinclair. 'We're not tied to Sinclair, we're totally independent – totally separate companies.' But – and this is the crunch – 'we do have a very good relationship with them. Sinclair respects Psion's capabilities.' Potter does not, however, see its position as privileged.

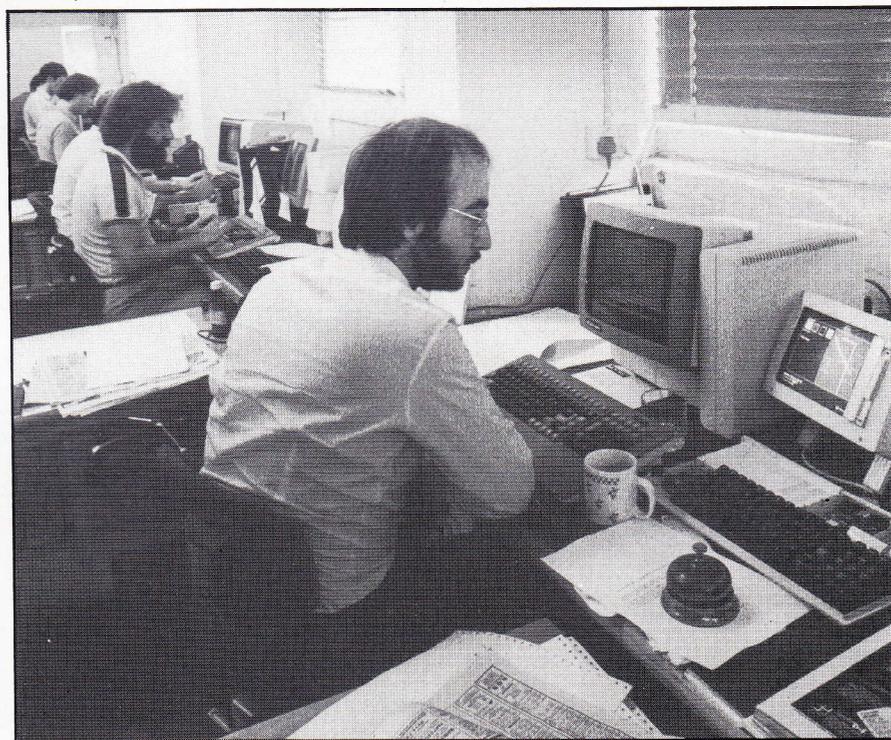
Psion's arrangement with Sinclair is one which might naturally lead a casual observer to believe that an exclusive agreement over Psion's products and ideas has been reached. In fact Psion sells the marketing rights to many of its products to Sinclair while still retaining the intellectual rights for itself.

This concept of intellectual rights is at the heart of Psion's philosophy. David Potter puts it succinctly: 'Software is an abstract, intangible thing and the law is very complex around it. It has to be understood that it's not ink but ideas which are important.' In other words, Psion retains the copyright to all the programs Sinclair has the right to sell in volume. This naturally includes all the programs which are bundled with the QL. 'But,' adds Potter, 'we guard our rights jealously.'

The agreement between Sinclair and Psion on, for example, the QL software, relates only to the Microdrives. Further negotiations would have to take place before



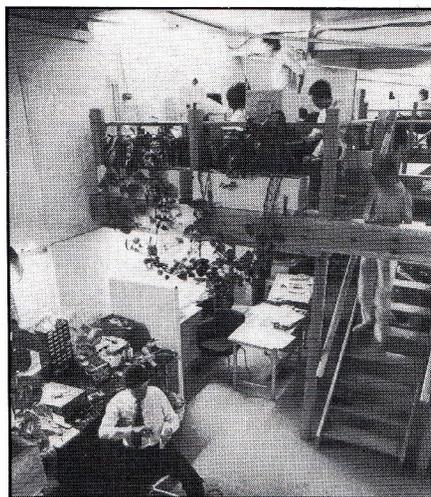
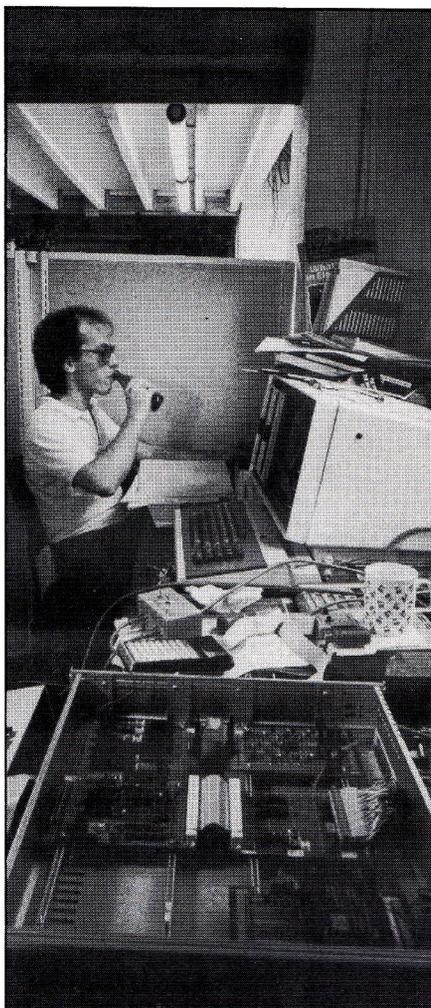
*A brief look around the Psion facilities.*



Sinclair could supply these programs on another storage medium. The intrinsic spirit of independence with which Psion approaches Sinclair is apparent when David Potter goes down in computing history by saying 'We can't comment on the future but we'll be seeking opportunities with Acorn and IBM.' Products are also in the

pipeline for the Commodore 64.

Another reason for not marketing its Sinclair range by itself is that, according to Potter, 'it makes sense that software – particularly applications and volume-market software – should be sold through the same channels.' And to achieve its aims – those of developing high-quality software products and



of becoming Europe's leading software house – Potter feels that 'we have to concentrate on penetration and volume. If licensing (ie to Sinclair) is a way to achieve greater mass than there's an argument in favour of it.' What is lost in revenue per unit in this kind of arrangement is made up for in volume.

The end result, in the case of the QL, is that Psion would be free to sell the programs on any machine it likes but wouldn't bother with 8-bit computers.

Having dealt with some of Psion's more interesting opinions, a look at the internal machine is now appropriate before reaching a conclusion. Psion is not one but

four companies. It was formed by David Potter (as Potter Scientific Investments) to 'recognise and exploit opportunities in the growing home and business computer market.'

A closer look reveals Psion as a rather diverse company, headed by a determined Potter. Indeed, on meeting the man face-to-face and on finding out how the company started, it is apparent that Potter has headed the company doggedly from its birth.

Psion Limited acts mainly as a holding company and a 'central source of expertise.' Psion Computers acts as a distributor in South Africa, selling machines from Acorn, ACT and Sinclair. This company produced £500,000 worth of profits in the fiscal year ending February 1984. This is 50 percent owned by Psion Limited, the rest being held by 'local partners.' Psion Systems writes the applications software and Psion Processors (both are 100 percent owned by Psion Limited) is working on 'a number of hardware developments.' Psion personnel were rather cagey about this. Basically, no clue was forthcoming as to what these developments are, every question drawing a 'no' answer. It is worthy of note that a large number of different computers were scattered about the research and development wing (a separate office) and that the Sirius figured strongly among them.

Psion appears to be a strongly hierarchical company, each of its key personnel being given definite roles. At the same time there is an indefinably familiar air about it, aided perhaps by the fact that Colly Myers, head of one of Psion's programming teams, was formerly General Manager of Psion in South Africa and is also Potter's half-brother. Technical Director Charles Davies took his PhD in computational plasma physics under David Potter. As with many successful companies, there is a close-knit group at the top.

Nor are Psion's successes confined to this small island. Home computer products are available in 30 countries. Psion works with distributors in those countries to translate versions of its programs regards ease of translation as an objective in the design stages of many of its programs.

From Psion's potted company profile (from which any unattributed quotes are taken), one of the company's objectives is to produce quality, useable and *bug free* programs.

In the light of Psion's excellent releases for the Spectrum in recent years, one can only sit back and guess as to why such disastrously bug-ridden versions of the much-

vaunted QL software have been released. Two main possibilities spring to mind. The first is that Psion did not receive that elusive hardware soon enough. During the press visit to Psion in March, it was apparent that the QL software was developed on a VAX that, under a full speed simulation, 'thinks it's a QL'. Psion also works a great deal in C, reckoning the language improves productivity, reliability and portability – all qualities for which C is well known.

In spite of the VAX, technical director Charles Davies admitted that it took 24 hours to 'get the programs going' on a QL when one finally arrived. 'We can do all the development without ever seeing the computer' claimed Davis, before admitting to that fact . . . Nevertheless, while at Psion the press was treated to a quick run-through of the Psion software actually on a QL. At the time this was quite an experience and we were made to believe that the software was practically ready.

The other main reason could be that Psion has not yet changed its attitude enough to cope with writing complex applications of this nature. Though the company is now nearly four years old, it is in the past two years that growth and success have really knocked at its door. In the financial year between November 1982 and November 1983, Psion's turnover was £10 million. Growth has been fast. While visiting Psion we were informed that further premises have been acquired – an old printing works near Marylebone station – new premises were rented in November 1983 as well. Developing the printing works will cost £1 million and will provide 10,000 square feet of space. Psion is set for further expansion, but is it ready for the changes that developing applications software – as opposed to games – will entail?

Undoubtedly the ideas which Psion so religiously guards are good ones – Sinclair itself is a company which survives on good ideas. But good ideas must be followed through reliably – ie they have to work. Without doubt Psion has the ideas people there to set innovation in motion. But does it have the weight and strength to cope with the inevitable support and maintenance any professional software suite must entail?

The climate of opinion regarding this enigmatic company is that it has a lot of work to do. A visit to its offices presented one with a self-assured profile, a company with a bit of panache and more than a bit of cash. Undoubtedly the QL catch is its most momentous one yet, and it might yet be the one that didn't fit the frying pan . . .



ORDER TODAY -  
PRINT TOMORROW!  
24 HOUR DELIVERY

## SCI(UK) SETTING NEW STANDARDS IN CUSTOMER SERVICE

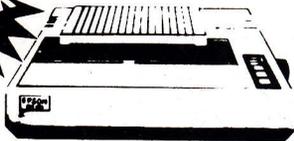
ORDER TODAY -  
PRINT TOMORROW!  
24 HOUR DELIVERY



**0730 68521 ANY DAY INCLUDING SUNDAY**

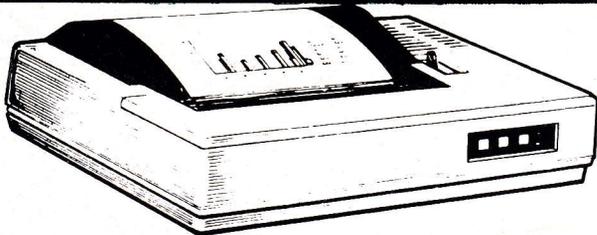
### EPSON LOW PRICE SPECIALS FROM £199

**7 DAYS  
A WEEK**



EPSON RX80 (DOT MATRIX) . . . . .	<del>£249</del>	<b>£199</b>	+ VAT = £228.85
EPSON RX80FT (DOT MATRIX) . . . . .	<del>£285</del>	<b>£239</b>	+ VAT = £274.85
EPSON FX80 (DOT MATRIX) . . . . .	<del>£438</del>	<b>£324</b>	+ VAT = £372.60
EPSON MX100 (DOT MATRIX) . . . . .	<del>£475</del>	<b>£365</b>	+ VAT = £419.75
EPSON RX100 (DOT MATRIX) . . . . .	<del>£450</del>	<b>£385</b>	+ VAT = £442.75
EPSON FX100 (DOT MATRIX) . . . . .	<del>£569</del>	<b>£499</b>	+ VAT = £573.85

### NEW! Canon PW-1080A £269.00



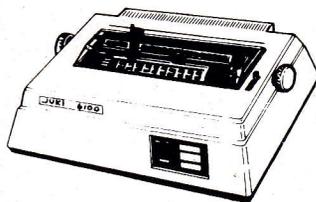
+ VAT = £309.35

80 cols; High speed printing, 160 CPS; bi-directional logic seeking; fantastic 27 CPS near letter quality; 23 x 18 matrix; very quiet - less than 60 dB; 4, 5, 6, 8 10, 12, 17 CPI; down loading for user-optional characters; high resolution graphics; handles various forms, roll paper, fan fold, single sheet and multipart copy paper.

ALSO AVAILABLE THE CANON PW1156A as above but 156 cols **£369.00** + VAT = £424.35

### PHONE 0730 68521 INCLUDING SUNDAY!

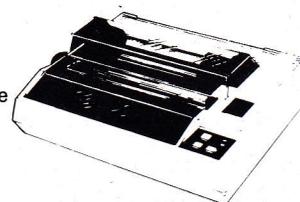
**JUKI 6100 just £349** + VAT = £401.35



20CPS: Bidirectional & Logic 10, 12, 15 & Proportional Spacing; Wordstar compatible 2K Buffer; 13 inch Platen Underline; backspace & lots more Centronics Interface Standard

**OPTIONAL RS232 TRACTOR AND SHEET FEEDER**

**SHINWA CP80 £179.00** + VAT = £205.58



Friction and tractor feed as standard. 80cps. Bi-directional logic seeking 13 x 9 dot matrix giving true descenders, sub and superscripts Italic printing and auto underlining Condensed, emphasised, expanded and double strike (can be mixed in a line). Parallel interface fitted as standard

### WE WILL MATCH ANY GENUINE PRICE ADVERTISED SCI(UK) IS NEVER BEATEN ON PRICE

MANY MORE PRINTERS AVAILABLE 1,000s OF BARGAINS SEND NOW FOR THE FAMOUS SCI(UK) CATALOGUE



FIDELITY 14" COLOUR MONITOR & COMPOSITE VIDEO



£189.00 + VAT = £217.35

We have interfaces and cables for all types of computers, including CMB 64, Vic 20, APPLE, TRS 80, IBM, BBC, SPECTRUM, DRAGON, TANDY, SINCLAIR, APRICOT, SIRIUS, MONOTECH, QL, ADVANCE, TEXAS ETC



24 hour nationwide delivery by Securicor £9.50 + vat. Bankers Orders; Building Society Cheques; Postal Orders; - same day despatch. All orders covered by the Mail Order Protection Scheme. Nationwide maintenance contracts arranged. Educational discounts very welcome.

**SCI(UK)**  
0730 68521

FREEPOST  
PETERSFIELD  
HANTS GU32 2BR  
TELEX 86626 MYNEWS G

DEALER ENQUIRIES  
WELCOME WRITE  
FOR DETAILS

CALLERS WELCOME SEVEN DAYS A WEEK TO OUR SHOWROOM AT 12 HIGH ST., PETERSFIELD, HANTS

**URGENT ORDER**  
PLEASE RUSH ME

NAME \_\_\_\_\_  
ADDRESS \_\_\_\_\_  
CREDIT CARD NUMBER \_\_\_\_\_  
SCIUUK FREE POST  
PETERSFIELD, HANTS GU32 2BR

Recommended  
by QL-User magazine

# At Last! THE SERIES THAT MAKES THE QUANTUM LEAP



*"I am certain that these books will add enormously to the enjoyment and practical use which QL users will get from their computers."*

**NIGEL SEARLE**

Managing Director, Sinclair Research Limited

**5 books to help you get the most from the QL, whether you are still waiting or are lucky enough already to have yours.**

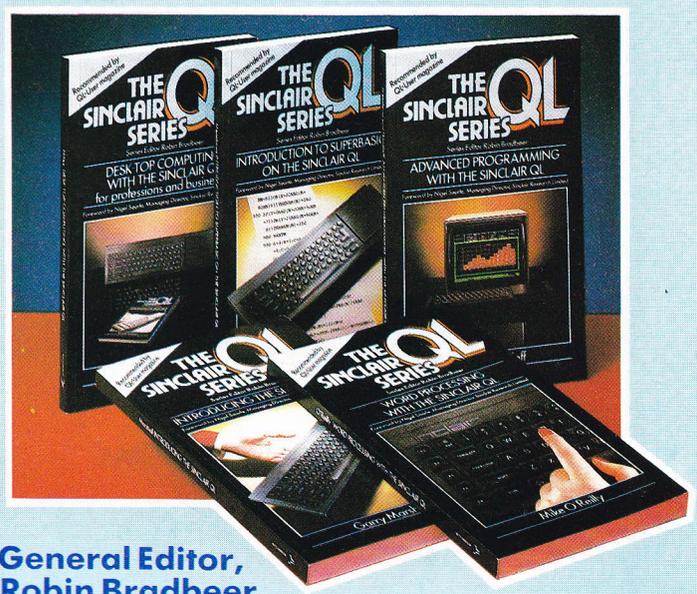
**Introducing the Sinclair QL** explains how the QL works and what you can do with it.

**Introduction to SuperBASIC on the QL** explains SuperBASIC and introduces its special features and qualities enabling you to master programming quickly.

**Advanced Programming with the Sinclair QL** is an essential reference work for users who really want to get to grips with the Sinclair QL. The book includes such topics as program logic representation, types of commercial program, programming techniques and document design.

**Desk-top Computing with the Sinclair QL** shows just what can be achieved in business computing using the Sinclair QL and how to get the best out of the four QL software packages: word processing, spreadsheets, database management and business graphics.

**Word Processing with the Sinclair QL** has been written to explain both the concepts behind the uses of word processing and how the QL word processing package operates and what it can do.



**General Editor, Robin Bradbeer and his team of authors received the help and co-operation from both Sinclair and Psion Software – creators of the QL software – to ensure that these books really are the ultimate handbooks for QL users.**

**Order now** – you won't have to wait long for your books! Just fill in the order form and return it to us with your cheque or money order. You can even charge your Access or Barclaycard account. We will send you your books POST FREE within 28 days.

**ORDER FORM** To: TBS, 38 Hockerill Street, Bishop's Stortford, Hertfordshire. QL

- Please send me:
- ..... copies of **Introducing the Sinclair QL** (at £6.95 each)
  - ..... copies of **Introduction to SuperBASIC on the QL** (at £6.95 each)
  - ..... copies of **Advanced Programming with the Sinclair QL** (at £6.95 each)
  - ..... copies of **Desk-top Computing with the Sinclair QL** (at £6.95 each)
  - ..... copies of **Word Processing with the Sinclair QL** (at £6.95 each)

I enclose my cheque/money order made payable to TBS for £ \_\_\_\_\_

Please charge my Barclaycard/Access account (delete as appropriate) number \_\_\_\_\_

My name and address is \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_ Signed \_\_\_\_\_

Dealer enquiries to: Doug Fox, Hutchinson, 17-21 Conway Street, London W1



Q.....Q.....Q.....Q.....Q.....Q

There will be no Q for our Quality packages for QL Users

Competitively priced monitor and printer packages.

For further information contact

**Zeal Marketing at:  
Vanguard Trading Estate,  
Storforth Lane, Chesterfield S40 2TZ  
Telephone: 0246 208555**

### QL UTILITIES

FOUR Programs on Microdrive for Sinclair QL to prevent Directory overflowing the screen, provide single-key LOADING or DELETion of files, repeat FORMATING of cartridges and back-up COPYING of whole or part of cartridge.

£10 from **WDSsoftware**, Hilltop, St Mary, Jersey, C.I.  
Tel: (0534) 81392

# QL

## £49<sup>00</sup> INC

### PARALLEL PRINTER INTERFACE

- ★ 12 months guarantee
- ★ Fully self-contained with connectors and 1.5 metre cable
- ★ Plugs into Sinclair QL's RS232C port and
- ★ Drives any CENTRONICS compatible printer eg Epson, Seikosha, Juki, OKI, NEC, Shinwa, Star, MCP-40, Roland, etc, etc

To order send name and address with cheque to:  
**MIRACLE SYSTEMS Ltd**  
6 Armitage Way  
Kings Hedges  
CAMBRIDGE CB4 2UE  
Tel: 0223 312886

*Sinclair and QL are  
trademarks of  
Sinclair Research*

# GETTING A QL

Then join IQLUG, an independent, non-profit making users group for Sinclair QL owners.

*The group offers:*

- Monthly newsletter**
- Free Software Library**
- Free advice service**
- Workshops**
- Support for local groups**

Membership of the group is by subscription to Quanta, the group's newsletter. A six month trial subscription costs £3.25.

*For further details contact:*

**BRIAN PAIN**  
**24 Oxford Street, Stony Stratford**  
**Milton Keynes MK11 1JU**  
**Telephone: (0908) 564271**

## TRANSFORM LTD.

**QL CENTRONICS INTERFACE £49.95p**

*When ordering please specify printer you are using*

### Printers

#### Dot Matrix

Epson RX80FT .....	£322.00
Epson FX80 .....	£437.00
Star Gemini 10X .....	£274.00
Brother HR5 .....	£152.00

#### Daisy Wheel

Smith Corona .....	£217.35
Brother HR15 .....	£407.00

*All above printers are supplied complete with RS 232 cards*

### Monitors

Philips TP2000 black and green .....	£86.25
Sanyo medium resolution colour .....	£313.00
Monitor leads .....	£5.75

*When ordering monitor leads specify whether monochrome or colour*

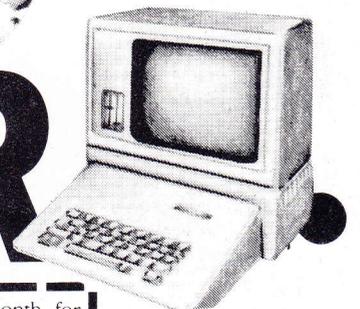
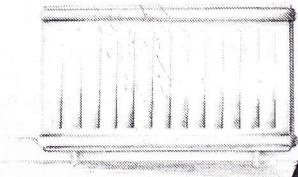
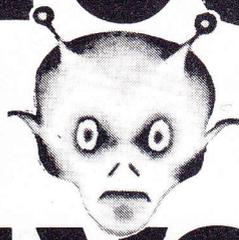


**TRANSFORM LTD.**  
**Dept QL, 41 Keats House, Porchester Mead,**  
**Beckenham, Kent. Tel: 01-658 6350**





# IN YOU WON'T FIND OR BUT YOU WILL DISCOVER HOW TO PREVENT CONTROL YOUR OR TALK TO THE WITH YOUR



**ELECTRONICS & COMPUTING**  
MONTHLY

Please send me **ELECTRONICS & COMPUTING MONTHLY** each month for the next 12 months. I enclose cheque/PO to the value of **£10.70** (UK)  
For Overseas rates, please contact Subscriptions Department

NAME .....

ADDRESS .....

SIGNED .....

DATE .....

Cheques should be made payable to:  
**ELECTRONICS & COMPUTING MONTHLY**  
Visa/Access, 24 hour answering  
service, 0858 34004. For new  
subscriptions only.

Please send to:  
Electronics & Computing Monthly  
Subscriptions Department  
Competition House, Farndon Road  
Market Harborough, Leics.



# Putting the QL Through its Paces

*On paper the QL looks like the bargain of the century. But how good is it in real life? Duncan Doenitz has been finding out.*

**U**nusually for a micro manufacturer (except Commodore), Sinclair has been markedly reluctant to provide test machines for the press. To carry out this review, we had to use a machine purchased by a member of the public. So to ward off angry letters from Sinclair, we'll make it clear now that this was one of the very first machines to be delivered – its undoubted warts certainly ought to be fixed before full-scale production starts, although as yet we have no indication from Sinclair as to when this will happen. Meanwhile, this review at least tells you what to expect if you get a QL from this initial batch...

Like previous Sinclair computers, the QL consists of a one-piece console containing the processor, all the other electronics and the keyboard. The console must be plugged in to an external monitor or TV before you can use it.

Unlike its predecessors, the Spectrum and the ZX81, the QL also has mass storage built into the console in the shape of two Sinclair Microdrive units. This makes the package much larger than before – the whole thing is about the size of the detachable keyboards you see on the IBM PC, the Sirius and the like.

## The keyboard

The case is moulded in two pieces in black plastic. It feels reasonably robust but I wouldn't want to throw it around too much. The keyboard uses those ultra-modern round

keytops on square bases that Scandinavian ergonomists love so much – and they look smart, too.

From the typist's point of view the keys have two main disadvantages: they are arranged absolutely flat, unlike most keyboards, on which the keys get towards the front, and they have a rather sticky and dead action.

The space bar makes a loud rattle when you hit it and seems to take a long time to come back up, while the carriage return is stiff and can stick if you press it too far off centre. By some unbelievable stroke of design lunacy, the space bar doesn't work when the shift key is held down! In other words, it isn't a keyboard which will appeal to fast typists or those who intend to pound it at a stretch.

A look inside the case reveals that it is in fact a calculator-style membrane keyboard with the keytops merely sitting on 'blips' in a rubber sheet, whose elasticity provides the return force – there are no mechanical springs. Still, it's much better than the Spectrum (to say nothing of the ZX81!).

The keyboard layout is very non-Sinclair, with Control, Escape and Alt (alternate) keys being provided for the first time. There are 65 keys altogether, in a more-or-less standard QWERTY layout. (Spectrum owners note: there are no second or third functions on any of them: phew!)

Down the left hand side of the keyboard is a column of five function programmable keys. The only key apparently missing is a Delete/Backspace key; this function is in fact provided by pressing Control and the left arrow cursor key. The nice touch is that Control and the right arrow key deletes to the right, as you'd expect, and in the Psion packages, Control up and down arrow delete the whole line and to the end of the line respectively. For some reason these don't

work in SuperBasic, though.

The arrow keys are unconventionally sited at the ends of the space bar, with the Control and Alt keys, but the arrangement works quite well.

## Electronics

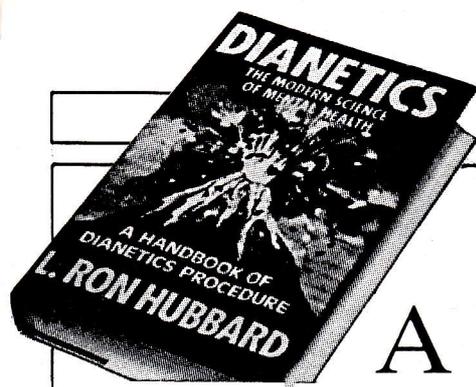
Inside the case is a single printed circuit board containing 16 64k RAM chips (making up the total of 128k of memory), the 68008 processor in its large 48-pin package, a second processor (an 8049) to control the keyboard and four custom gate arrays. Two of these arrays control the Microdrives while the others manage the memory, the colour display, the serial port and the network (which isn't yet installed).

Two EPROMs (eraseable programmable read-only memories) hold the QL's operating system, QDOS, and SuperBasic. Or rather they *nearly* hold it, because Sinclair's programmers couldn't make QDOS and SuperBasic fit into the two 16k EPROMs in time for the launch so early models – of which the review machine was one – require a third 16k EPROM in a plastic cartridge to be plugged into the ROM cartridge slot at the back of the case. Eventually the system software will go into a 32k and a 16k ROM inside the case and early buyers will get their machines upgraded for free.

The 68008 is a version of the Motorola 68000 processor which many engineers regard as the most powerful of the current generation of microprocessors. It's used in the Apple Lisa and Macintosh as well as the Sage and other big business machines.

The 68000 has a 24-bit address bus so it can directly address 32 megabytes of memory and a 16-bit data bus so it can move 16 bits at a time to and from the memory.

The 68008 is a simplified version



# A PATH TO CLARITY OF MIND

Consider the potential of a machine with a thousand times the capacity of any computer currently available. Consider the potential of an organic computer that is self programming, self generating and limitless in its scope. What you are now considering is the human brain, the single most powerful computing machine on the earth. How then do we utilize the enormous capabilities of this on a machine? This was the problem that faced L. Ron Hubbard and "Dianetics: The Modern Science of Mental Health" is the detailed research to provide the answer.

It starts like this: "A science of the mind is a goal which has engrossed thousands of generations of Man. Armies, dynasties and whole civilisations have perished for the lack of it . . . and down in the arsenal is an atom bomb, its hopeful nose full-armed in ignorance of it".

An all-embracing statement of great propensity.

It then goes into 400 pages explaining how the mind works with its variety of compartments, memory banks and linkage of the senses and sets out a system towards bringing the mind to a condition of 'clear', a Dianetics term meaning a mind devoid of aberrations and which can perceive and compute rapidly.

Over several weeks I interviewed a considerable number of people asking what the book had done for them.

Mike Phillips is now 39 and he started his working life learning stage management from the bottom up. From provincial theatre he went to West End shows. The move came at a time when stage management was in a period of transition – lighting and sound arrangements had become very sophisticated.

"I needed to know a lot more about sound engineering", he said. "I had a mind necessarily filled with a dozen things but I knew I had to grapple with new

forms of electronics".

An acquaintance introduced him to the book on Dianetics which, he claims, had a salutary effect. "I followed the reasoning in the book and even took a short course. It is clarity of mind that emerges", he said, "It gave me the extra faculty to study highly technical books".

Ultimately he was appointed to a post of chief sound engineer with a major group in London and eventually brought highly technical innovations into the entertainment world.

Another type of engineer who took up the philosophy of Dianetics is George Turney. This man, qualified in structural engineering, has a great deal to do with oil rigs in different parts of the world.

He had a long period in California where the pace of work also carried the rewards of a sunshine life.

"I had a high social life as well", he says, "but there came a time when life was an endless round of work and weekendening at one place or another and I realised there were more things to do than this convivial merry-go-round.

"Dianetics: The Modern Science of Mental Health, was one of the subjects I read up and I began to plan a future with greater clarity. In time I came back to England and started handling my own affairs and business".

The third man in this category was Dale Bulbrook who runs computer operations for an organisation in Sussex. He has had a lot of experience at home and abroad. He worked some years with ICL machines and spent time in Australia working as a programmer and systems analyst.

Back in England, he came across the Dianetics book by chance. "I picked up the book at random", he said, "and found myself agreeing with points on logic and computerisation. The

philosophy on how the mind works proved fascinating and beneficial to programming".

## The Analytical Mind

Now, turning to the book, Hubbard states this in a chapter on The Analytical Mind and the Standard Memory Banks: "The human mind can be considered to have three major divisions. First, there is the analytical mind, second, there is the reactive mind, and third, there is the somatic mind.

"Consider the analytical mind as a computing machine. This is analogy, because the analytical mind, while it behaves like a computing machine, is yet more fantastically capable than any computing machine ever constructed and infinitely more elaborate . . ."

He goes on, "What would you want in a computing machine? The action of the analytical mind – or organiser – is everything anyone could want for the best computer available. It can and does all the tricks of a computer. Over and above that it directs all the building of computers. And it is as thoroughly right as any computer ever was.

It does not take a great deal of intelligence to see what Hubbard is driving at. The mind with all its memory banks starts as a pure piece of machinery. And, of course, he points out, much can happen in the sequences of life that introduces clutter.

He postulates the theory that the memory begins from day one and goes on storing every experience from then on. The object of Dianetics therapy, he propounds, is to bring about a **release** or a **clear**.

The reference to therapy is the system in the book which can be self-tested in taking hinderances (clutter) from the memory or subconscious. Also Hubbard has adopted a series of expressions singular to the subject, hence:

A **release** is an individual whose major upsets and anxiety have been dealt with by Dianetics therapy. A **clear** is a person who, as a result of therapy, has neither active or potential aberration. To **clear**, he says, is to handle all the physical pain and painful emotion of the life of the individual with the result that the person is in full control of his

own life.

The second group of people of interviewed were in the highly creative field.

Nicky Hopkins and Graham Todd are names that may mean little to the average man in the street. But both, in their own way, are brilliant musicians. Hopkins was the pianist with the Beatles in recording sessions and later with the Rolling Stones. Now in his thirties he lives in a house in Hollywood and composes film music.

Todd is also a pianist and worked with Cliff Richards for six years and with other stars before that. He has composed theme tunes for TV series and his name is on more than 12 million records. Both these men portrayed the kind of life that comes with international travel, mass audiences and high-pressured talent.

"There comes a time when the mind loses awareness of time and place and it is then you want to find a basic value to make you realise where you are and where you age going in the world", said Hopkins. "In due course I came across Dianetics in America and figured that Hubbard was on the right lines. Since those days I followed the subject through and found the concentration and mental energy to compose music which has gone down well in America".

Todd also extolled the result of studying the book. In a staccato burst he said. "Gives you awareness, improves communications, clarifies my aims. You ought to try it . . ."

The experience of these people may be summed up in the words of Hubbard: "Dianetics will bring the optimum analytical ability for the individual and, with that, all recall. The experience of his entire life is available to the **clear** and he has all the inherent ability and imagination free to use it. His physical vitality and health are markedly improved . . . his ethical and moral standards are high, his ability to seek and experience pleasure is great".

As a book the issue has brought on one thing clear to me. I have met some very interesting and exceptional people. As an independent observer I can say they are worth a book in themselves – G.S.

The publication is available from the Dianetics Information Centre, Saint Hill, East Grinstead, Sussex. price £3.95 including postage and package.

which has only an 8-bit data bus and a reduced address space of 1 Mbyte. This makes it easier and cheaper for companies which are used to 8-bit processors to design a system around it using the techniques and peripheral chips with which they're familiar. IBM did a similar thing by using the 8088 instead of the 8086 in its PC.

The 68008 will be half as fast as the 'real' 68000 in all those operations which involve moving data to, from and within memory because it has to do twice as much work, moving two bytes rather than one word. Since not all the activity of a real program involves such moves, though, it will do rather better than half as fast overall. In the QL, the 68008 is run at 7.5 MHz.

Unlike its predecessors, the QL has a reset button, at the right side

## The picture on my TV was grim.

the power socket, a three-pin connector, not the jack used on the ZXs.

Next in line is the RGB colour video output for a monitor. This is an 8-pin DIN connector and in fact the first two pins provide monochrome monitor output so it can drive either a black and white or colour monitor. No RGB cable was supplied with the review machine and connecting it to a Microvitec Cub monitor (which has

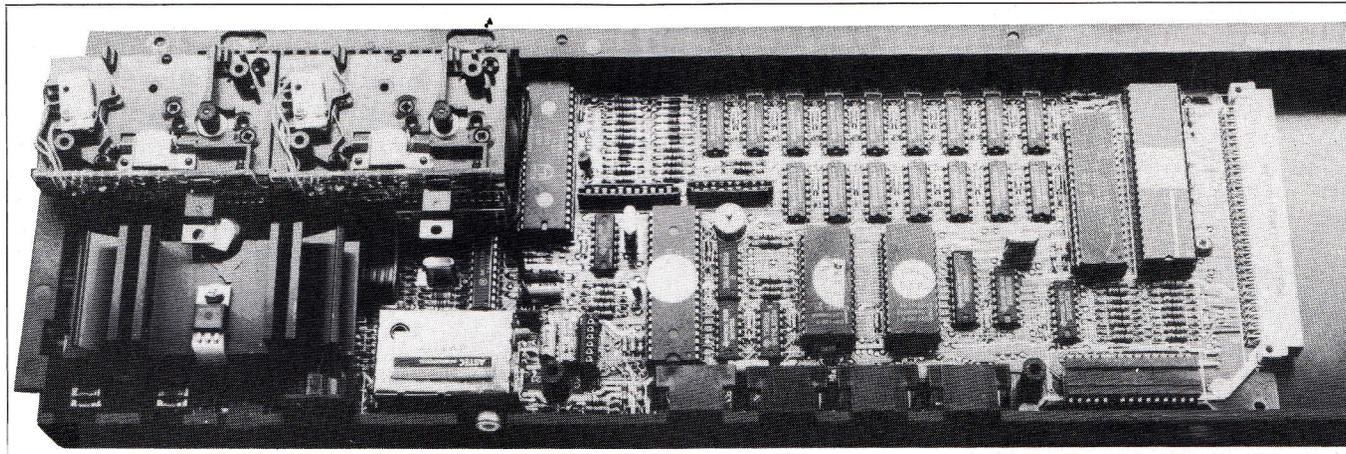
tilation slots behind the Microdrives.

## Display

When the QL is first switched on, a panel at the foot of the screen asks whether you're using a TV or a monitor. Function keys F1 or F2 are pressed to choose.

On a TV set the display is limited to 40 characters per line, which is near the limit a TV set can reasonably handle anyway. On a monitor, either 40, 64 or 80 characters per line can be displayed and the choice can be made under software control.

The picture on my Hitachi colour TV was grim: the colours shimmered and rippled and the green on black letters used in the SuperBasic editor were virtually unreadable, though the red on white were



All the I/O connectors are soldered directly to the main board. Note massive heatsinks for the voltage regulators.

of the case behind the Microdrives. But it still doesn't have an on/off switch – you just pull the power plug out as usual on Sinclair machines.

Reset performs a complete cold start of the system, so any program or data in memory will be lost. If there is a tape cartridge in Microdrive 1 then this will be inspected and if a program is found, this will automatically be loaded and run; otherwise, the system goes into SuperBasic.

There's a clock/calendar chip inside the machine which can be read from SuperBasic via the variable DATE\$, providing a string in the format '1984 May 23 12:34:04'. The early provisional manual states that this clock has its own battery backup; I doubt that the review machine did, because every time the machine was powered on it would return a different date.

## Input/output

At the back of the QL case is a row of I/O sockets. The first two (left to right) are jack sockets for connecting into the Sinclair Network – on the review machine, these were unconnected inside. Then comes

a 7-pin DIN connector) was a matter of trial and error – mostly the latter.

Next to the RGB socket comes a phono socket for connecting up to a TV set and following this are four Telecom-style sockets. The first two are RS232 serial ports, one wired as a terminal and the other as a modem. The second pair are joystick ports. The last port is an edge connector for ROM software cartridges but this is permanently occupied on the early machines by the 'tumour' containing the overflow from the operating system.

Round at the left hand side of the case is a cover plate leading into an empty space inside the case with a 24-pin edge connector at its end. This will accommodate the 512 kbyte memory expansion which will be available at some unspecified date in the future (unless another company beats Sinclair to it). Just in front of the Reset button at the right hand side is a seven-line edge connector for adding more Microdrives – up to six – to the system.

A final I/O channel, in the form of a small piezzo-crystal speaker, lives inside the case and plays its tinnny little noise through the ven-

not so bad. The display was slightly better on a Sony TV, but still not good. If you intend to do any serious word processing, then budget for a monitor, as Quill looks dreadful on a TV even in 40-column mode.

On a Microvitec monitor, by contrast, the picture is superb – steady, with well-saturated colours and crisp lettering. The only drawback here is that in 80-column mode in Basic the characters run off the edge and corners of the screen. Editing Basic programs is a pain when you can't see the line numbers! There may be some way in software to move the picture over but without manuals I couldn't find it.

Colour is quite complex on the QL. In 80-column mode it's only possible to have four colours on screen at the same time, while in 40-column mode you can have eight plus a flashing attribute too.

The palette from which you choose is different in the two modes. In low resolution (256×256 dots, equivalent to 40 characters) the palette includes the eight solid colours red, green, blue, magenta, cyan, yellow, black and white. In addition, there are numerous 'tints'

of these colours produced by overlaying various patterns of stripes and dots (called 'stipples') in a second colour on top of the first. The patterns are too coarse for these to be true tints but they are distinguishable shadings and look rather attractive. You can get them from Basic by setting the PAPER colour to a number between 0 and 255 (there are actually only 224 distinct shades and half of these are mirror image duplicates). The program in Figure 1 produces the colour chart and shows how it's composed quite neatly.

```
10 FOR j = 0 TO 255 STEP 16
20 FOR i = 0 TO 15
30 PAPER i+j:PRINT"";
40 NEXT i:PRINT
50 NEXT j
```

Figure 1

In high-res (512×256, or 80 column) mode, the loss of magenta, blue and yellow results in a reduced palette of red, green, black and white and all their tints. Some of the red and green tints can look almost yellow but there are no blues at all. The colours are set by the time-honoured Sinclair commands, PAPER and INK.

## Using the Microdrives

The Microdrives fitted in the QL are mechanically similar (but not identical) to those used on the Spectrum and use the same blank tape cartridges as the storage medium. However, the storage format is different so that no Spectrum software can be loaded (and it wouldn't work even if it could) and you can't read Spectrum data cartridges.

Each cartridge holds a maximum of 255 sectors of 512 bytes each but in practice the useful capacity will be below this. Some space is used for the directory and other system stuff.

The operating system contains error detection routines which can detect bad sectors on the tape and lock them out so that they can't be used, which could result in data loss. This means that the capacity of a cartridge will tend to shrink with age and wear.

The QDOS directory listing always shows as its first item the number of free sectors against the number of usable sectors, eg 17/219. On the cartridges supplied with the review machine there seemed to be between 215 and 224 usable sectors, some 107 to 112 kbytes.

The Microdrive cassettes contain 200 inches of video quality tape travelling at 30 inches/sec (very fast, if you think about it). Even without a computer, you can calcu-

## The capacity of a cartridge will tend to shrink with age and wear.

late that this means that one circuit of the tape takes around seven seconds, which is the longest time it can take to find a given sector. This is much slower than a floppy disk drive but faster than an audio cassette recorder (which can't in general find any old sector but must read the whole tape). Figure 2 shows some examples of the times taken to do things.

Format a blank cartridge	30
Get directory listing	0-7
Save 15-line Basic program	15
Load 15-line Basic program	15
Load Quill word processor	75

Figure 2: Times in seconds to perform Microdrive operations.

But what about the reliability of the medium? During this test, two cartridges reported 'bad media' errors and refused to load, which out of 11 supplied is a rather high percentage. It's imperative that master copies of software be used only once or twice to make backup copies for use, and that all data be backed up onto separate cartridges. I'd be happier with two backup copies if it were commercial data and worth money. The comparatively high cost of the medium for its capacity (£4.95 each) renders this an expensive business.

In short, the Microdrives are not suitable for serious business use, though they should be adequate for home word processing and hobby database activities. There is a lot of talk at Sinclair of a Winchester disk interface for the QL; perhaps Sir Clive intends to shock us all next year with a ridiculously low-priced 3½ inch Winchester unit.

## Operating system

The QL's operating system has been dubbed QDOS, the pun no doubt being fully intended. (It's perhaps an unhappy choice as the name has been used elsewhere to stand for Quick and Dirty Operating System.)

QDOS is not altogether finished (hence the extra EPROM pack) and several of its more important features, including multi-tasking, are not implemented on early machines.

Unlike the CP/M, MS-DOS and Unix operating systems, QDOS does not stand alone and has no user interface of its own. It's a collection of housekeeping routines in ROM which the user can only

access at present through SuperBasic. Later, perhaps, you will be able to access its routines through assembler language and through high-level languages – a C compiler is promised.

QDOS does all the usual operating system things, like managing file input and output to the Microdrives, getting characters typed at the keyboard and putting them on the screen. QDOS routines will also handle the RS232 ports and the Network, when it's ready. The screen handling routines go well beyond what most operating systems provide, allowing windows and some built-in graphics operations too.

All I/O operations in QDOS are channel-oriented. To move stuff in or out of the computer, you send it through one of 16 channels to a logical file. A logical filename consists of the name of one of the hardware devices, like the screen, a Microdrive, the Network, etc, with a filename tacked on using the underline symbol, which is used extensively in SuperBasic. For instance, MDV1.myfile

would be a file on Microdrive 1. To write to this file you would open a channel to it using OPEN, and then PRINT to that channel. For common operations like SAVEing a Basic program or PRINTing to the screen, it isn't necessary to specifically open a channel as SuperBasic keeps a sensible set of default channels.

The console (CON\_) device can have a window size and location and a type-ahead buffer size specified in the device name. This is one way to open a window, but SuperBasic has a WINDOW command which is less verbose to use.

SuperBasic's commands for QDOS operations are DIR to get a Microdrive directory, COPY <name> TO <name> to copy files, SAVE to save a SuperBasic program, SBYTES to save a chunk of QL memory, LOAD to load a SuperBasic program and DELETE to delete a file. OPEN is used to open a channel to a device or file.

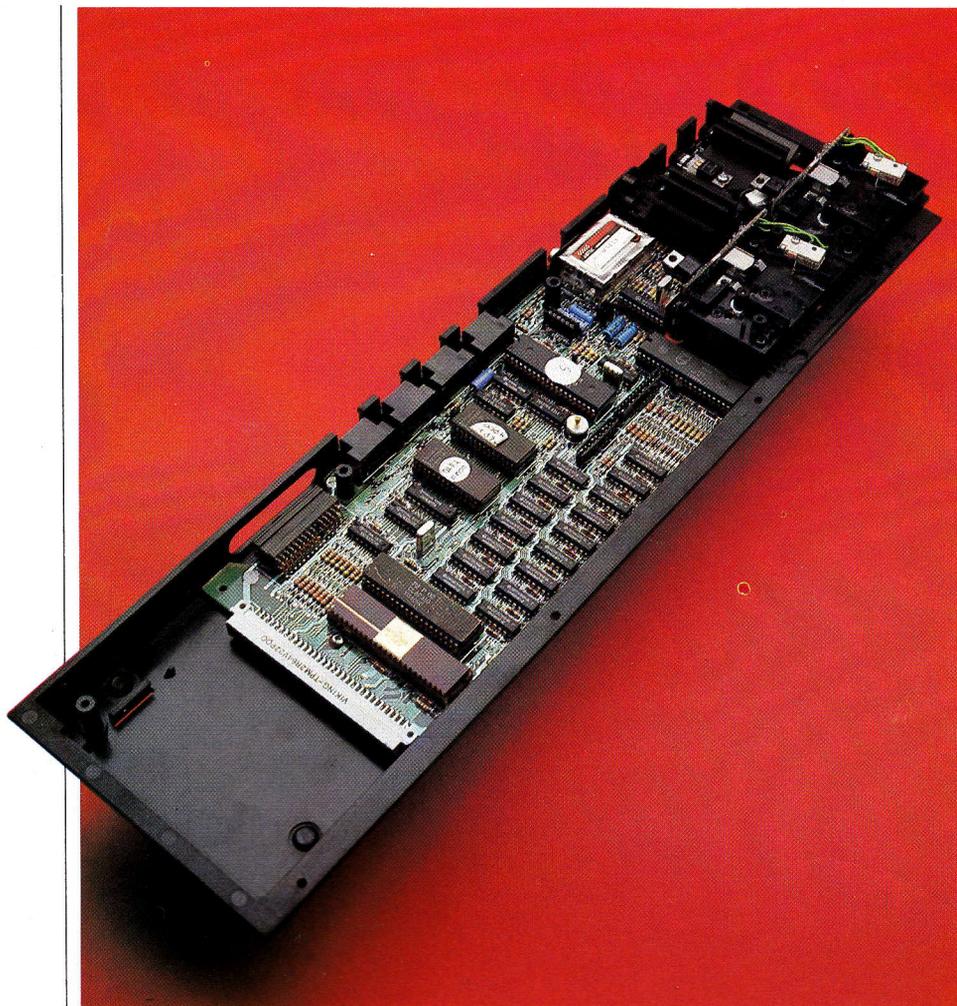
Multi-tasking, when it finally gets here, will be done through the EXEC command, which takes a list of programs and runs them all in parallel, eg:

```
EXEC MDV1_firstprog ! MDV1_
secondprog
```

where the ! sign means 'set up a pipeline for the two programs to communicate through'. I can't tell you how well it works because it didn't. A PAUSE command can be inserted into a Basic program to make it wait in multiples of 20 ms.

## SuperBasic

The Basic supplied with the QL is a



Inside the QL. The 68008 is the long chip in the foreground; undressed Microdrives lurk in the background.

big departure from that Sinclair Basic which was once going to become the world standard. (Phew! That was close . . . !) It's a big step in the right direction, too. Spectrum owners may miss the single keystroke command entry but the gains are well worth it – think of it as an opportunity to learn to type . . .

The biggest improvement is in the range of control structures offered. Though GOTO, GOSUB, ON . . . GOTO and ON . . . GOSUB are all provided for compatibility (ie, to transcribe programs from other Basics), there is no need to use them at all.

Procedures with parameters can be defined using DEFine PROCedure and END DEFine (you only need to type DEF PROC and SuperBasic fills in the rest), which knocks GOSUB on the head. So:

```
10 DEFine PROCedure hello (colour)
20 PAPER colour
30 PRINT "Hello world"
40 END DEFine
```

can be called by typing 'hello 5' to print on different background colours. The variable 'colour' could have been made LOCAL to the definition by adding '15 LOCAL colour'. The example of 'colour' also shows that variables can have long meaningful names (up to 255 characters!) which can include numbers

and the underline character, eg player3\_score.

Functions can also be defined with DEF FN, either in the usual Basic way in one line or using any number of lines of code ended by END DEF and using RETURN to return the result. Functions can also take any number of parameters.

If . . . THEN . . . ELSE . . . ENDIF can have any number of lines of code between THEN and ELSE and ENDIF, allowing properly-structured entry and exit from program sections without using GOTOS. ELSE and ENDIF are both optional, so single-line IF . . . THENs are allowed when appropriate. As well as IF there is a type of CASE construct, called SELEct ON, which has a default clause ON . . . REMAINDER to catch the exceptions:

```
10 SELEct ON flavour
20 ON flavour = "strawberry"
30 PRINT "Yecch!"
40 ON flavour = "pistachio"
50 PRINT "Yes, please!"
60 ON flavour = REMAINDER
70 PRINT "Oh, all right . . ."
80 END SELEct
```

SELEct can also be used in a short form which is ideal for range tests:

```
10 SELEct input_value - 1 TO 1200 : PRINT "Data within range"
```

The faithful old FOR . . . NEXT loop is generalised in two ways. The

## SuperBasic is a big step in the right direction.

index variable for the loop can have odd values tacked onto its range and an END FOR can be added, as in:

```
10 FOR size = 1.23, 3.45, 45.5 TO 12
20 PRINT size
30 NEXT size
40 PRINT "Finished"
50 END FOR size
```

It's permissible to leave a FOR loop early using EXIT loopname, eg EXIT size, and in this case any code between NEXT and END FOR (Sinclair calls it an 'epilogue') doesn't get executed.

Unbounded loops can be programmed with REPEAT . . . END REPEAT, which has the same syntax as FOR. You can exit with EXIT or force an immediate repeat with NEXT. Both REPEAT and FOR can be used in a single line short form analogous to that for SELEct, eg:

```
10 FOR count = 1 TO 100 : PRINT count
```

Taken altogether these are very welcome improvements to Basic and make it possible to write well-structured and readable programs as well as saving a lot of code if used properly. The only thing missing to make this a truly grown-up programming language would be some way of building up complex data structures instead of the clapped-out old DATA and READ, but that comes close to looking a gift-horse in the mouth.

All the usual arithmetic and logic operators are provided and, as a bonus, the bitwise logic operators &&, !! and ~ (bitwise AND, OR and NOT).

SuperBasic supports floating-point arithmetic with the huge dynamic range of  $10^{-615}$  to  $10^{615}$  but with only seven digits of precision, which is not enough for serious business use. Integers are supported (suffix variable name with %) but for some reason they are not allowed in a loop index in the current version, which slows things down quite severely – see under 'Performance' below. Strings can be up to 32,768 characters long, which should be quite enough!

Arrays can have a number of dimensions and they can be assigned to using 'array literals', eg:

```
10 DIM array (3,3)
20 array = {{245}{156}}
```

which is handy. They can also be

**Amazing how played out some things**



become.

# Will you think the same of your micro in 6 months' time?

Remember the days when every record player was gifted with a needle instead of a stylus. 45's were "in," 78's were "out."

Before the days of hi-fi and laser discs. When a graphic equaliser was a breakthrough on the football pitch instead of in music technology.

Look at a record player now and it's a museum piece.

It's like that with micros too. The machine you thought would give endless hours of fun and interest often becomes a five minute wonder. Played out within months. Or so you thought.

But imagine communicating with other micro users on a nationwide mainframe system. Updating yourself

daily with the very latest computer news and reviews. Paging a special Bulletin Board. Feasting from Prestel's vast menu. Even downloading a choice of software *absolutely free*.

In fact, imagine 30,000 pages at your fingertips and you've imagined what it's like to be on line with the Micronet 800 system.

For just £13 a quarter (and, for most of you, a local telephone call whenever you want to connect up) you could subscribe to the Micronet system.

The only extra you need to connect up is a modem unit. Which is a small enough outlay for what it buys.

Micronet's features are almost limitless and constantly updating so why not see it in action at John Lewis stores, selected W. H. Smith shops and Spectrum UK dealers.

Or fill in the coupon for our brochure.

You'll find you won't want to play on anything else.



Please send me the full facts about Micronet 800.

Name \_\_\_\_\_

Make/Model of Micro \_\_\_\_\_

Address \_\_\_\_\_

\_\_\_\_\_

Telephone \_\_\_\_\_

MICRONET 800, Durrant House, 8 Herbal Hill, London EC1R 5EJ.  
Telephone 01-278 3143.

MICRONET 800, Durrant House, 8 Herbal Hill, London EC1R 5EJ.  
Telephone 01-278 3143.

™Prestel and the Prestel symbol are trademarks of British Telecommunications.

## Making the most of your micro.

'sliced' the way strings are in Sinclair Basic:

months (7 to 12)

to give access to more than one element at a time.

A very unusual feature is that SuperBasic will, wherever possible, 'coerce' different data types into the correct type for an expression. This means that you can add a string to an integer, for instance, and get the right answer: 128+"345" still gives the answer 473. You'd get an error for 128+"PLONK", though, because PLONK can't be coerced into a number. If you try to put floating point numbers into an integer variable, they will be correctly rounded off.

One consequence of coercion is that '+' can't be used to concatenate two strings as it is in most Basics; '&' is used instead.

One peculiarity of SuperBasic is that RUN doesn't initialise variables to zero as it does in most Basics - uninitialised variables return an asterisk instead of their value. This can be very handy as long as you know it's happening and initialise all variables explicitly in the program.

## Graphics capabilities

Graphics are handled by a system of virtual co-ordinates which always refer to the current window, not to the whole of the screen. The graphics origin is always the bottom left hand corner of the window, which fits in better with normal practice than the top left hand corner as used on many computers.

POINT sets a single point; LINE draws lines either relative to current position or between specified points: CIRCLE will also draw ellipses of any eccentricity and at a specified angle. There is a keyword FILL which takes two parameters, a channel number and a number, but I couldn't get it to do anything at all in the absence of any documentation. Rectangular areas of colour can be defined with BLOCK so I presume FILL does something different. There's supposed to be a SCALE command to alter the scale of the virtual co-ordinate system but I couldn't make that work either.

WINDOWS can be defined and given BORDERS of any thickness and colour. To output to a different window on the same screen, it must be assigned a different channel number; you then PRINT to the appropriate channel. A window's contents may be moved sideways with PAN and up and down with SCROLL, but these commands don't buffer the contents so that anything which goes out of the

Anything which goes out of the window is lost forever.

window is lost forever.

The word CSIZE allows you to alter the size of the characters displayed on the QL screen. The height can be either 10 or 20 pixels and the width 6, 8, 12 or 16 pixels, set independently so that there are eight possible styles. The largest of these only gets 32 huge characters on a line.

Sound is handled by the single command BEEP, which takes a list of up to eight optional characters:

BEEP duration, pitch1, pitch2, grad1, grad2, wrap, fuzz, random

The first two are normal duration and pitch. Pitch2 specifies a second note and BEEP can be made to oscillate between the two notes, like a police siren. Grad1 and grad2 determine the speed with which this wobbling takes place and fuzz makes the sound progressively dirtier as it's increased, ending up with white (or at least grey) noise. I couldn't work out exactly what wrap does - presumably it wraps.

The sound generator will run in parallel to the processor so once you've set it BEEPing it carries on regardless of what else you're doing; trying to remember the parameters to make it stop is a lot of fun. The noises it makes are highly suitable for Space Invaders-type games but not for much else - certainly not for music.

As to the programmer's utilities, there's AUTO line numbering, RENUMBERing of programs and TRACE to help debugging. When the QL enters SuperBasic, you're presented with a screen split three ways. At the bottom is a black, full-width window into which you type commands. The upper part of the screen is split vertically into a white and a red half. Listings of programs appear in the left, white half and program output appears in the right, red area. These windows are assigned the default channels 0, 1 and 2.

This is all fine, in as much as

The review machine was quite clearly not a finished product.

your listings don't keep scrolling off the screen, but it doesn't disguise the fact that the QL still uses the same old line editor that originated with the ZX80. To edit a program you have to type EDIT xxx to get the line back down into the input window, where you can use the cursor keys left, right, delete left, delete right to alter it. The draft manual mentions a full screen editor which can scroll both up and down through a listing; presumably this is either not finished yet or has been dropped through lack of ROM space.

## Performance

With all the usual caveats about the significance or otherwise of Benchmarks (they only tell part of the story), I nevertheless tested SuperBasic using the Eratosthenes Sieve program employed by the US magazine *Byte*. (It finds all the prime numbers up to 16,381.) The program was adapted to make best use of the superior control structures in SuperBasic.

The results were far from impressive, taking 4480 seconds for 10 iterations of the test. This, according to *Byte's* very extensive table of results, puts it squarely down among the slowest three languages they have ever tested - slower in fact than Tandy and Texas Instruments Basics and only marginally faster than Z80 Cobols.

A different set of benchmarks published in a rival magazine shows SuperBasic in a slightly better light, roughly comparable to Microsoft Basic on a 4 MHz Z80 computer, but slower than BBC Basic (although the transcendental maths functions come out as quite fast). Given that SuperBasic is running on a 7.5 MHz 68008, though, this is hardly a cause for celebration.

Why this should be is a matter upon which one can only speculate; my own guess is that the luxury of 'coercion' is in fact very expensively bought in terms of interpreter time. Sinclair has responded to queries about the speed of SuperBasic with the defence that it does poorly in benchmark tests because the 'coercion' mechanism does not at the moment allow integer variables to be used as a FOR loop index - the index is always kept as floating point. To give this thesis a whirl I recoded the Sieve test using only REPEAT loops with integer variables. The result was a slowing down to 5300 seconds for 10 iterations...

## Applications software

The Psion applications programs

bundled free with the QL, Quill, Abacus, Archive and Easel are described elsewhere in this issue. I found them to be excellent in most respects. Easel, the graphics drawing program, is both spectacular to look at and exemplary in its ease of use.

## Documentation

There isn't much to say about the documentation as I wasn't supplied with any. Eventually, an early provisional manual, dating back to the launch in February, was obtained but the version of Super-Basic described differed in numerous ways from that implemented on the test machine, as did much of the other information.

## Conclusions

The Sinclair QL is a curious compromise indeed. It's quite definitely not suitable as a real business machine *in its present configuration*. The Microdrives are neither fast, capacious nor reliable enough, and the keyboard is scarcely adequate. The Microdrives are sufficiently fast, though, to be very satisfying to someone who is used only to cassettes – in other words, to the upgrading hobbyist.

On the other hand, it's expensive for a hobby machine, especially given that the picture quality is not up to much on a domestic TV and the graphics and sound facilities from Basic are hardly stunning compared to cheaper competitors.

SuperBasic is an improvement on all the home computer Basics currently available (Comal being the honourable exception) and one hopes that it will teach better programming practices to the next generation of hackers. It will need a lot of speeding up on the full production versions, though, and the floating point precision will need to be increased for business or scientific use. It also deserves an editor that does it more justice – why not a 'folding' editor that works at procedure level?

A point which can't be shirked is that the review machine was quite clearly not a finished product, despite the fact that it had been sold to a member of the public.

When the machine is finished and has been around for a while, then I'm sure that some exceptional software (written in machine code) will emerge for it. After all, who would have thought that 'Ant Attack' was possible on the Spectrum until somebody did it? And the QL hardware packs a lot more potential punch than either the Spectrum or the BBC.

Until then, though, it's really a machine for that ill-defined breed, the serious home user. ■



# The QL at Work

The next section of *QL User* is devoted to in-depth reviews of three of the four packages which come free with the machine: Quill, the word processor, Archive, the database system, and Abacus, the spreadsheet.

With these three packages, Sinclair has covered about 90 percent of most business micro users' general needs. That, at least, is the theory – how well does it stand up in practice, though?

We must state now that – as you'll see in the next few pages – the software as we tested it was by no means suitable for serious use. Mostly it is bug-infested and in some cases downright unreliable, the very last thing anyone needs in business.

We tested the software as supplied on a machine purchased by a member of the general public. Thus, our reviews reflect the situation faced by early QL buyers. Sinclair has steadfastly refused to supply a review machine, stating rather pompously that 'our customers come first'. Clearly, judging from the state of the software, Sinclair would prefer that the press didn't get its hands on the machine until the bugs were sorted out.

We feel otherwise: *QL User* owes no allegiance to Sinclair, only to the users of Sinclair's latest product. Rather than produce the sycophantic ravings in which some machine-specific magazines indulge, we feel we should tell the story as it is, regardless of whether this happens to fit the Sinclair party line.

Clearly, though, the bugs now infesting the machine and its software will at some stage be eliminated. We don't know when this will be but we hope it will be soon. The QL is, on paper, a remarkable machine, let down in reality only by a premature launch. As we went to press, we heard that new releases of the software were imminent and we hope that this will incorporate remedies to many of the gripes which our reviews contain.

So what's the point of reviewing the software if it's going to be

changed?

Firstly, if you are contemplating buying a QL, you'll be interested to know just how worthwhile the software is. Does it do what you want your computer to do? How useful will it be to you? We can answer these right now by examining the facilities offered by these packages, and – bugs permitting – we can tell you how easy they are to use and how useful they are, especially when viewed with the perspective of the general business micro software market.

So if you're in a pre-QL-ordering situation, our advice is to pay only passing attention to the bugs reported here. Chances are that by the time you get the machine, at least the howlers will have been fixed and you should have a pretty powerful little system at your disposal.

If you already have a QL, you'll probably be depressingly familiar with much of what we report here, but at least you'll have the cold comfort of knowing that it's the software that's faulty and not the way you were trying to use it!

We anticipate that some sort of software upgrade will be offered to existing owners by Sinclair. After all, selling products which don't fulfil their stated purpose might contravene the Sale of Goods Act if no upgrades are offered. We intend to keep a very close eye on this situation indeed: we feel that those people who invested £400 of faith in Sinclair by ordering when the machine was first launched deserve to end up with a properly-functioning machine and we'll be pushing on their behalf for this to be done.

Finally, if you're wondering what happened about Easel, the graphics package which also comes with the QL – well, wait until the next issue!

Peter Rodwell.



# QUILL

*Sid Smith, News Editor of  
Micronet 800, tackles the QL  
word processing package.*

I ordered a QL because I wanted a cheap word processing package. The machine's real keyboard, Microdrive data storage and bundled software seemed to offer three of the five essentials of word processing. Throw in the family TV and the only additional expenditure is for a printer.

But then again, aren't those three supplied essentials as much reasons for avoiding the QL as for buying it? After all, when has Sinclair ever made a decent keyboard? When has a Microdrive ever competed directly with floppy disks? And could Psion, master of games programs though it might be, really put together serviceable applications software?

First the good news: the QL keyboard is fine. Its basic mechanism is little different from that of the Spectrum: a molded rubber mat transmits your dab to what is essentially a flexible circuit board. This is pushed downwards until electrical tracks on its underside make contact with a similar board beneath it. A circuit is completed and the machine registers the input. What the QL adds to that set-up is merely an upper layer of hard plastic keys held in a rigid frame.

Surprisingly, it works very well. Even the fact that the QL's keyboard is as flat as the Spectrum's seems to make little difference to the user – although it may have been responsible for the way I tended to miss the space bar from time to time. And it even clicks.

Put all that together, add (very important, this) the automatic word wrap provided by Quill itself, and – to my surprise – I found that

the QL turned me into a faster and more accurate typist than I'd ever been on a conventional typewriter.

## Hardware quirks

However, traditionalists may be disconcerted that the shift key has been displaced (by the QL's control key) one row up from its usual bottom left hand corner position, but they'll soon get used to it. The 'enter' key on my machine, presumably because it's so big, required an authoritative poke amidstships before it paid any attention. And, far from providing a numeric cluster, the QL scatters its arithmetic functions somewhat, and puts a couple of them in shifted positions. But none of that is particularly important.

And those Microdrives? Well, these devices have been available as peripherals for the Spectrum for almost a year. There were undeniable problems with early Microdrive ROMs but these seem to have been overcome and the device has settled down as a serviceable little unit for Spectrum owners who want some of a floppy disk's performance at some of a floppy disk's price.

I was surprised at the amount of Microdrive accessing which went on with Quill. The program clearly stores in RAM only part of its command structure and only a fraction of any document under composition. Drive access is therefore required before the package can carry out many simple commands and before much in the way of text manipulation is possible.

The performance of the Microdrives themselves is disappointing.

Sinclair has been promising for some time that the present 60-odd second load time of the Psion packages would be drastically reduced. This hasn't happened yet and the poor performance of the Microdrives in this respect may also be reflected in the lengthy whirrings which frequently halt all input to the Quill program. On one occasion a Microdrive got up to one of its old Spectrum tricks, going into an unstoppable loop which eventually required a power-down and consequent loss of the data in memory.

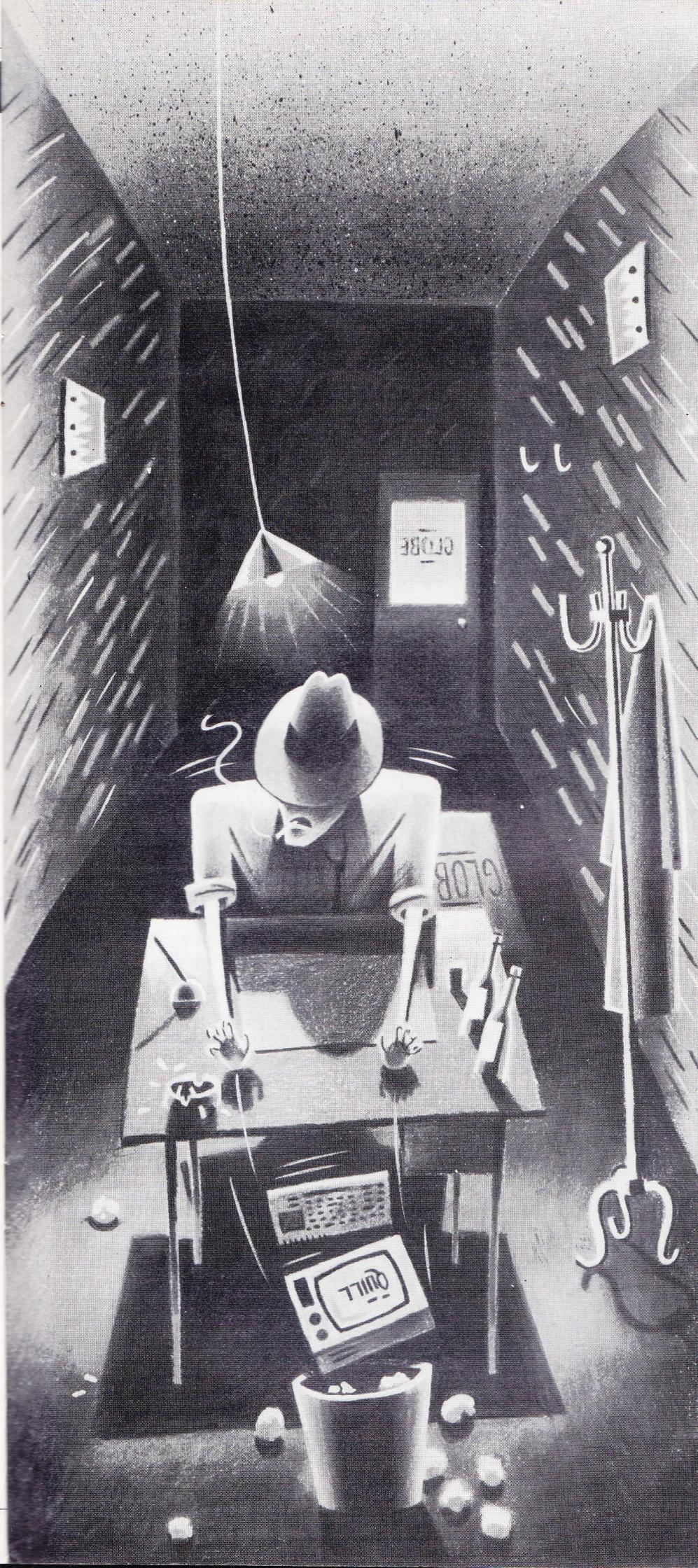
## Setting up

I had Quill version 1.0 and I was using a QL from the first batch to be delivered. Those two facts may have been responsible for what followed . . .

When you use the package for the first time, you are immediately prompted to make a back-up copy of Quill. I know a few people who had problems as early as this, but I managed OK. Your back-up copy is then placed in the left-hand Microdrive, with a formatted data cartridge in the right-hand drive (or Microdrives 1 and 2, as the manual calls them).

It is possible to load the program via some long Microdrive command but it's far easier to press the reset button and then choose either TV or monitor display – at which point the cartridge will load itself.

In TV mode the program defaults to a 64-column display, though it is possible to select 40 columns. Monitor users will find Quill automatically switching to an 80-column display. In fact I found the display was good enough to use



in 80-column mode even with my rather second-rate TV. The quality of the lettering with this setup was comparable to a dot matrix printer with a used ribbon; each character was visible but they tended to blur into each other.

I found it acceptable although other TV users might prefer to keep to the 64-column mode, save the text to the Microdrive and then load it into an 80-column monitor display for a final viewing before printout. No-one except – possibly – owners of a Sinclair flat-screen TV, will need the 40 columns display.

My chief disappointment with the display was the poor colour resolution. Pixel crawl was so bad that I had to turn my TV's colour control right down into a monochrome display, but I have seen Quill working well on other TVs.

### **Quill in use**

Like the other packages bundled with the QL, Quill divides the screen into three distinct areas.

The top few lines are occupied by the control area, which contains reminders about the uses of the function keys and the operation of the various Quill commands. The large central area is used to display your document and can be expanded to full screen size by switching out the other two areas. The final zone contains information about the current document – its name, its word, line and page counts, any special typeface you may be working in and whether you're using Insert or Overwrite mode.

Pressing function key 1 invokes the Help routine. Only one-third of the Help menus can be held in memory at one time, so F1 usually disables the keyboard and starts up the Microdrives. When the Help data has been read into RAM, the normal Quill display is replaced by frames of information about the program, the user working downwards through a series of menus into more and more specific information.

Unfortunately, the Psion programmers hadn't got around to finishing the Help function on my version of Quill. Only about two-thirds of the Quill commands had been annotated, the user being thrown back to the previous menu frame whenever an attempt was made to get details on the remainder. This was my first Quill bug; there were more to come.

The F2 key is used to toggle the two prompt areas off and on; as already described, this enables the central display area to fill the screen. It works quickly and well.

F3 gives access to the Quill commands (of which more later) and F4 enables the typeface of your document to be altered. Available typefaces are bold, superscript, subscript and underline. All four options are fully displayed directly as input. Again, this function worked well, although the lengthy Microdrive accesses were starting to grate somewhat by this time.

## Writing with Quill

Rumours were already circulating about the degree of 'queueing' which Quill users had experienced. Even moderate typists, it was said, were able to get several letters

### A disconcerting lag between text entry at the keyboard and its appearance on the screen.

ahead of the on-screen display, experiencing a disconcerting lag between text entry at the keyboard and its appearance on the screen.

But even after this preparation, I was staggered to find that, with no particular effort, I could race 25 letters ahead of the display! I should add that in most cases this phenomenon is more odd than inconvenient, though it is occasionally exasperating to notice a typing error being left several words back as the cursor huffs and puffs to catch up with you.

At times like this, the neat range of rapid cursor movements available under Quill comes in handy. As well as the expected single-space movements made by pressing only the left and right cursor controls, larger leaps can be effected with the shift and cursor keys held down together. Using shift with the left and right controls moves the cursor one word in the chosen direction; shift plus up or down keys move the cursor about in units of a paragraph.

Similar flexibility is offered by the delete function. Control and the left and right keys are used for zapping single letters and whole words disappear when shift is added. Mass destruction of lines

### Quill holds only a very limited amount of your document in RAM.

left and right of the cursor can be invoked using the control and up/down keys.

It's worth remembering, however, that Quill holds only a very limited amount of your document in RAM. Should you want to move the cursor out of that area, Microdrive access – with consequent keyboard disabling – is inevitable. But whenever you're working within the RAM-based text, operations are generally swift and efficient – provided the part of the Quill program to carry out these operations has been loaded from the Microdrive! As I said earlier, Quill seems incapable of storing its full command structure in RAM – to carry out most Quill operations, the relevant command code has to be loaded from the Microdrive.

Word-wrap and right-hand justification – aligning the right-hand edges of text, as in most printed material – are both swift and automatic. Justification is carried out by inserting extra spaces into the line so text with an unusual number of long words can look peculiar and three spaces between words is by no means rare.

The operation of Quill at this

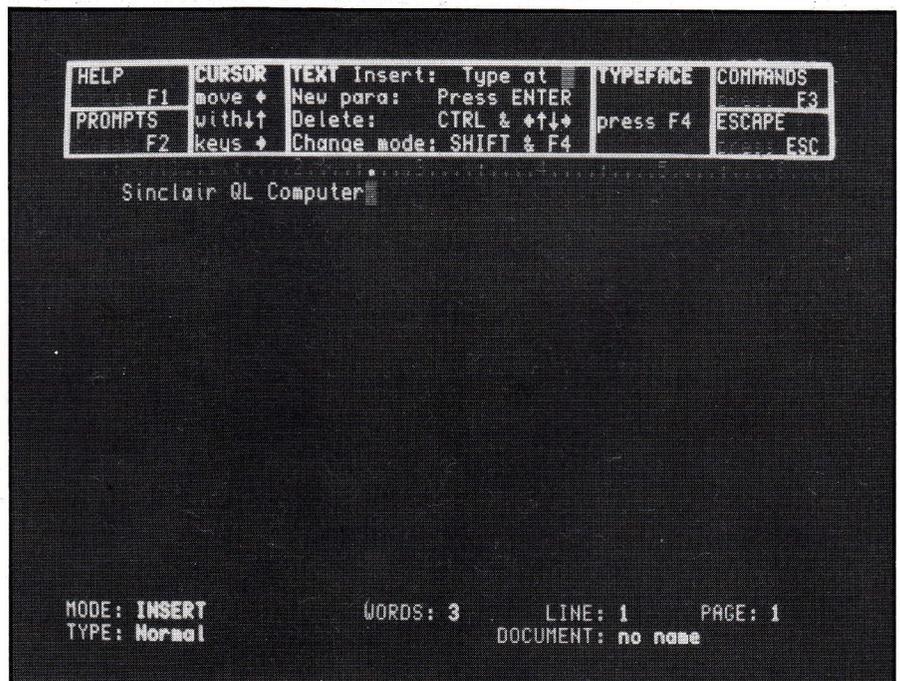
bread-and-butter text inputting level was generally pleasing. But then I started delving into the commands; they're presented here, as in the manual, in alphabetical order.

## Quill commands

*Copy* does what you'd expect: move the cursor to the beginning of the text to be duplicated and press 'enter'. As the cursor is moved down through the target text, the text is displayed in inverse video. The exceptional clarity this gives to the copy selection process turns out to be entirely necessary. Copy doesn't allow you to move the cursor backwards; should you overshoot the end of your target text, your only recourse is to hit the ESCape key and start the process again.

Provided you manage your selection successfully, the rest of the process is simple enough. Press 'enter' to register the end of the text to be copied, move the cursor to any point in the document and press 'C'; the highlighted passage will be duplicated and justified quickly and efficiently. The final stage of this process can be repeated indefinitely, enabling multiple copies to be made with little effort.

This command doubles as a block move, as you can choose to delete



Quill in its editing mode. This picture was taken with a monitor – TV sets give poorer quality.

the source text from its original position.

**Design** sets your page layout and six functions can be altered. Bottom and Upper margin fixes the number of blank lines at the top and foot of each page. These margins are not displayed on screen and neither are the Gaps Between Lines (that's line spacing to you and me).

Psion's explanation for this particular lack of what-you-see-is-what-you-get (a prized quality in word processing whereby the screen displays text exactly as it will appear on paper) is reasonable enough: it would waste a lot of space on the display.

The manual consoles us with many references to a command called View, under which an entire page can be displayed with the letters represented by tiny blocks, thus enabling the operator to cast a designer's eye over the document before it's printed out. My only regret was that this command didn't appear anywhere in the program...

Other Design options comprise the facility to change the number of your first page and the number of columns on the screen. They work well but it was while I was experimenting with Design that the program crashed for the first time: the screen froze, the keyboard went numb and there was nothing for it but to press reset and lose all my text.

**Erase** works just like Copy, with the same illuminated target area, same restriction on moving backwards and the same automatic justification afterwards.

**Files** proved to be a minefield of bugs, omissions and obscurities. Like Design, Files is composed of subordinate commands – this time

## A minefield of bugs, omissions and obscurities.

directed towards the management of Microdrive operations.

The first option is called Backup and is supposed – I think – to help you make a second, security copy of a file on the Microdrive. All I know is that I followed the instructions a dozen times and it just kept telling me, 'File does not exist'.

Delete wipes a Microdrive file – and it works! Hooray!

Format does what you'd expect to a Microdrive cartridge. The manual says: 'You can not recover the contents of a deleted file so you should think carefully before using this option.' This is called dramatic irony since the manual also says that it's the cartridge in Microdrive 2 which is the default drive for formatting, which is appropriate since that's where the data car-

## Impetuously pressing Enter leaves your Quill back-up cartridge squeaky clean and factory-fresh.

tridge lives. In fact, as the on-screen prompt reveals, the true default is Microdrive 1. Impetuously pressing Enter leaves your Quill back-up cartridge squeaky clean and factory-fresh. I know: I've done it.

Import is the fourth and last option under Files. It serves 'to insert another file from a Microdrive cartridge into your document, at the position of the cursor'. No doubt it does, though after three program crashes, losses of RAM-based text, resets and long climbs back into Quill, I decided to take Psion's word for it.

**Footer:** 'This command allows you to specify a line of text to be used as the last line on each printed page'. Good word processor functions are available for the editing and positioning of your chosen text. Again, the Footer is not displayed on screen; again reference is made in the manual to a non-existent command (this time called 'Defaults') with which to view a completed page.

**Goto** is a quick way of moving the cursor to the beginning or end of your document, or to the top of a specified page within it.

**Header** is like Footer, only higher.

**Hyphenate** started off as a good idea. The problem with automatic word-wrap, as indicated earlier, is that a log-jam of long words can leave Quill with a choice between a line that's too full or too empty.

Inevitably, the outcome is three words and 30 spaces of padding.

The Hyphenate command lets you place a buried line break into a long word. If any reformatting is necessary and the word happens to fall at the end of a line, Quill breaks the word at the specified point – and even sticks the hyphen in for you. 'The command,' says the manual, 'will have no apparent effect on the word if it is not at the end of a line.'

Well, yes, but since Hyphenate is itself well-buried (at the end of five separate keystrokes and the inevitable Microdrive accesses) I'm sure that most users will be like me: they'll wait for the awkward word wrap to happen, go to where the line break should be, move into Insert mode, type a hyphen and a space and then let Quill do the rest.

So far, I've been glossing over most of the disagreements between instructions in the manual and those on screen. With the instructions for Hyphenate, however, Quill achieves a certain purity: manual and screen prompts disagree – and they're both wrong.

**Justify** is excellent. You can select Left, Right and Centre justification, each option being enacted on the screen with impressive speed. Finally, press 'enter' to confirm your selection and return to the text mode. Quill doesn't have to work the justification all the way through the document on Microdrive since the formatting instructions for each paragraph are kept in RAM.

**Load** is used to fetch a file from Microdrive. It worked every time... but there's an unavoidable system overhead: even two words needed 15 to 20 seconds of Microdrive access.

**Margins:** as with all on-screen manipulations in Quill, Margins works well. It fixes the position of the left and right margins of your text and of the paragraph indent.

**Merge:** 'The merge command takes a copy of a named Quill document from a Microdrive cartridge and inserts it, at the position of the cursor, in the document currently in memory.' It differs from the disastrous Import command in that the latter is intended for files from Abacus or Archive and it works fine.

## Manual and screen prompts disagree – and they're both wrong.

*Page* is for forcing a page break. The manual and screen prompts disagree about how to implement this command. After experimenting, I concluded that the screen prompt was correct.

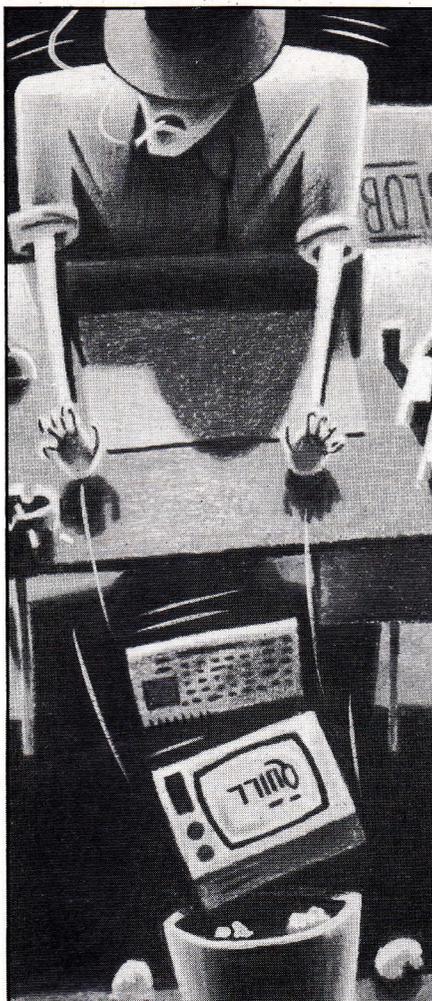
If you make a mistake, however, and want to cancel a page break, the manual and the screen prompts agree. After experimenting, I concluded that neither was correct. After further experimenting, I concluded that I'd better never want to cancel a page break.

*Print.* The Quill cartridge contains – or should contain – a file called `INSTALL.BAS`. This is a Basic program which, when run, reads from another file on the cartridge called `INSTALL.DAT`. This second file contains printer drivers for popular machines like the FX-80 and if your printer is on the list you can select the appropriate driver, which is then loaded into yet another Microdrive file. This last file is called `PRINTER.DAT` and will henceforward be accessed by your version of Quill as its printer device.

If your printer isn't on the `INSTALL.DAT` list, you can configure your own driver via a table in the same file, or you can leave Quill with the `PRINTER.DAT` file in its original state. This last enables the program to send out plain text to most RS232 printers.

*Quit* rather implies that it ought to be a fifth Psion program containing the best of QL bugs. In fact it merely contains the best Quill bug.

The purpose of *Quit* is to move you from Quill to Basic; it does this with brutal efficiency via a kind of automated reset. Along the way, though, it offers you the option of saving your current document to Microdrive before the Big Crash. So, you accept this option, the Microdrive whirrs and stops, the program dissolves and you're right back at the reset position. But when you reload Quill and try to load your document from the Mi-



crodrive – you've guessed it: it didn't get saved!

*Replace* searches your document for a specified piece of text and replaces it with another of your choosing. The replacement can be longer than the original; the original can be embedded within a longer word. The user is presented with each individual occurrence of the original string and must individually authorise its replacement. Manual and screen prompts (of course) disagree.

The *Replace* command suffers acutely from the odd way that Quill divides your document between Microdrive and RAM. Indeed, *Replace* can't even search much of the limited amount of text held in memory and exasperating Microdrive accesses are needed after every two or three lines of examination.

*Save* works OK, though it was while using this command that my data cartridge went into its one and only unstoppable spin. As with *Load*, there's an irreducible system overhead to pay on every transaction – saving 131 words to an empty cartridge took 33 seconds. Screen and manual dis-

## A predictably excellent piece of screen handling.

agree – again.

*Search* is just like *Replace* but without the replacing.

*Tabs* is a predictably excellent piece of screen handling. Four different types of tab stops are available; you can have up to 16 of them and they can be placed anywhere on the line.

The left tab stop acts like a left margin, positioning the text to its right. A right tab does the opposite and a Centre tab is halfway between the two. The handy Decimal tab organises a column of numbers of any length such that their decimal points align. The manual is wrong about this one.

The last two commands form a worthwhile end to this list of Quill commands.

*View* simply doesn't exist; and

*Zap*, despite its Buck Rogers name, is remarkably slow at its chosen task of deleting the current document. Personally, I blame the Microdrive accesses.

## The verdict

The most damning comment on this program is also the simplest: I didn't dare use Quill to write this review.

Psion says it has 'spoken in the strongest terms to Sinclair' about the latter's failure to incorporate updates to the Quill manual. And it has sent Sinclair a less bug-infested version of Quill itself. So we can hope that the second batch of QLs to be issued will contain a worthwhile version of this crucial word processing package.

Meanwhile, early users of Quill are landed with a program which bears the signs of bungled haste which characterise the entire QL project. Like the computer it accompanies, Quill will probably be terrific when it's finished. At the moment, it's a word processor you can't trust – which means it's useless. □



# ARCHIVE

*Gareth Jefferson examines  
the QL's database system.*

Computers in the home tend to end up doing one of two things. If they have joysticks but no printer, they get used mainly for playing games. If they have no joysticks but do have a printer, they spend most of their time being used to write thank-you letters or pleading letters to bank managers. Computer journalists invariably use them for writing copy for magazines such as *QL User*.

Neither of these extremes really gets the best from a computer (although few would deny that a word processor is one of the best investments a wordmonger can make). The actual data processing involved in a word processor is relatively trivial, yet everybody knows that the computer's real strength comes from its ability to handle arithmetical and logical operations in large quantities at astonishing speed.

In the middle ground between the fancy string handling of word processors and the number-crunching of sophisticated games lies the 'database'.

Databases share many of the features of word processors in that the information stored in the data files – and presented on the screen or in printouts – consists mainly of character strings. But whereas word processors are pretty 'dumb' – one word means no more and no less than any other to a word processor – a database assigns various levels of significance to some of the words within it, and can perform complex logical operations on them.

Before looking at the performance of QL Archive in more detail, it's worth describing in general terms just what a database program is.

At the very simplest level, a database is a computerised replace-

ment for a card index. The data is organised into one or more 'files', each file consisting of a collection of related data. Each file contains a number of individual 'records' and each record contains a number of individual 'fields'. A very simple card index style database on your book collection might be organised – either in a real card index, or in a computerised analogue – like this (for example):

**A file – called 'BOOKS' consisting of 700 records; Each record details one book, and is made up of separate fields – TITLE, AUTHOR, SUBJECT, PUBLISHER etc.**

Figure 1 shows the idea graphically. The important thing to remember is that all databases, from simple ones such as Cardbox (published by Caxton) to elaborate ones such as dBase II (published by Ashton Tate) are organised like this. The great differences come in what the programs are able to do with the data held in the files, records and fields, and in how easy they are to use. As a general rule of thumb, very easy-to-use data bases are not very 'powerful', and vice

versa. To see what 'powerful' means in this context, let's take a look at what a database user might want to do with the data.

## Facilities

At the simplest level, a database, QL Archive included, consists as a single file of related records, each record consisting of identical 'fields'. Naturally, the data in each field of a record will differ from the data in the fields of other records. A file might consist of records of all the books in your collection, records of all your company's clients or records of all the invoices you have issued. QL Archive can do more than that, however. Unlike many databases, it is able to work on more than one file at a time. But we'll come to these more advanced capabilities later and consider it in its simplest, 'card-index' mode.

On loading Archive into the QL, using the command LG0 MDV1—ARCHIVE<ENTER> or by pressing reset, the user is presented with a screen divided into three areas; a 'control' area occupying the top four lines of the screen, a large

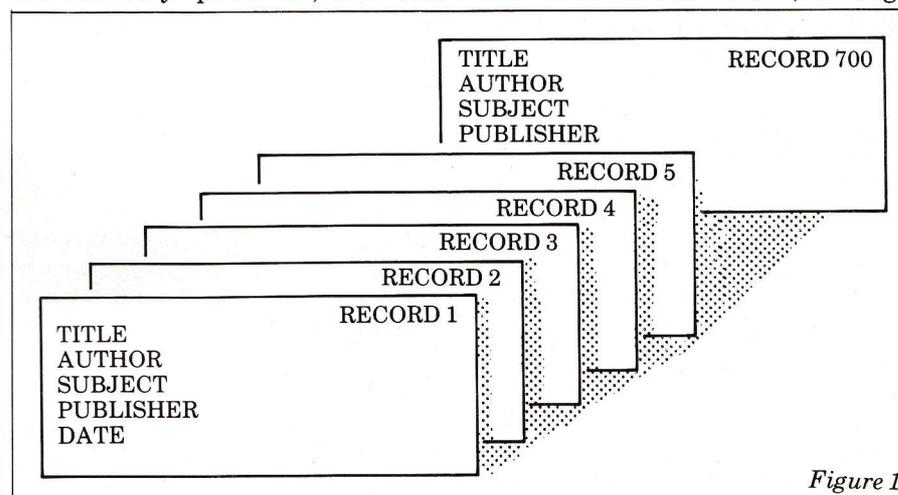


Figure 1

central 'display' area and a four line 'work' area at the bottom. Until instructed otherwise, Archive will be in the 'keyboard interpreter mode'. This means that commands recognised by Archive can be typed in at the keyboard. The commands appear in the 'work' area at the bottom of the screen. On pressing <ENTER>, the commands are executed (interpreted) directly.

Psion, the company which wrote Archive, has created a program that responds to statements that are virtually the same as SuperBasic (though, interestingly, Archive was written in C on a VAX-11 super-mini computer). The user has a choice of 63 powerful commands which can be executed directly, or he can create 'procedures' (mini-programs) to make the database do whatever he wants.

To illustrate the difference between the built-in commands and user-written procedures, suppose that a card-index style file has been created listing details of the books in your collection, and that the file has been given the filename BOOKS. To gain access to this file, you use the command OPEN "BOOKS". This causes the file to be read into memory from the Microdrive cassette, but nothing will be displayed on the screen.

## Commands and procedures

To view the first record in the file, you use the command DISPLAY. This shows only the first record. To view the next record, you type in the command NEXT. You can move about the file with a variety of commands - FIRST (to show the first record), LAST (to view the last record), BACK (to view the record before the one currently being displayed) and NEXT (to look at the record following the one being displayed).

Other often used commands include FIND, to 'find' a record containing a specified string, as in FIND "Xavier Hollander", or SEARCH to find a specified string in a specified field, as in SEARCH SUBJECT\$ "Assembly language". If all this typing seems like too much trouble, the user can write a procedure to simplify it all and allow single-key commands to be used. The EDIT command is used to invoke the program-writing editor, which allows the statements of the program to be entered from the

keyboard. A program or 'procedure' to simplify the standard commands for moving through a file could be written as follows:

```

EDIT (the command to put Archive
into program edit mode)
proc SIMPLE ('proc' indicates the
start of a procedure and is created
automatically by the EDIT command.
SIMPLE is the name we have given to
this procedure)
while key$ <> "Q" (input from the
keyboard)
  sprint (instruction to print to screen)
  let KEY$=getkey() (gets keyboard
input)
  if KEY$="F": first: endif (executes
'FIRST')
  if KEY$="L": last: endif (executes
'LAST')
  if KEY$="B": back: endif (executes
'BACK')
  if KEY$="N": next: endif (executes
'NEXT')
endwhile (end of 'WHILE' statement)
close (closes file if "Q" is pressed)
endproc (end of procedure statement;
inserted automatically by the EDIT
command)

```

This procedure can be saved by using SAVE "SIMPLE" <ENTER>. It can be invoked by typing LOAD "SIMPLE". Once this has been done, pressing the keys "F", "L", "B" and "N" will automatically cause the FIRST, LAST, BACK and NEXT commands to be executed.

This has been a very simple example of the use of the editor to write programs to be automatically executed by Archive as though the specified inputs were built-in commands. Similarly, SuperBasic-like program sequences can be written to perform actions as simple or complex as you like. A very simple example is given in the User's Manual. It shows that, while in the 'keyboard interpreter mode', SuperBasic-like sequences can be given to print characters on the screen. For example, the sequence below will print the numbers 13 down to 1 on the left side of the display area:

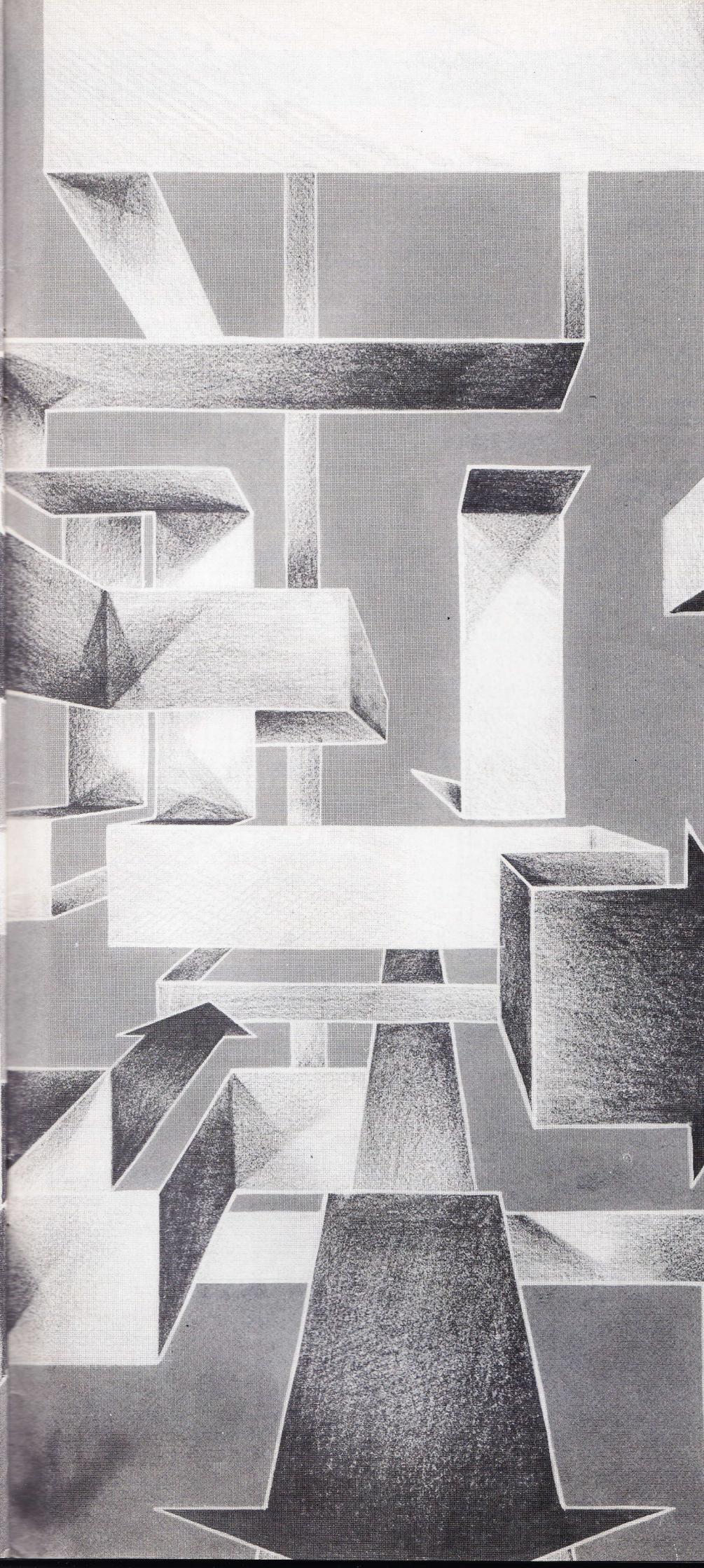
```

let x=13
while x <> 0
  print x
  let x=x-1
endwhile
<ENTER>

```

These SuperBasic-like statements can be written in as direct commands to be executed from the work area, or they can be written as part of a named procedure to be invoked whenever the procedure name is typed in.

FILE



Once a procedure has been created, it can be saved by using the SAVE ".PROCEDURENAME," command. Once saved, any procedure can be loaded (by typing LOAD ".PROCEDURENAME,") and then invoked by simply typing the name of the procedure. Thereafter, any commands defined in the procedure need only be typed in to be executed.

This may all sound very complicated until you try it. What it

---

### **Programs of any complexity may be created.**

---

means in fact is that programs (procedures) of any complexity may be created and loaded in memory to be invoked whenever they are needed. Suppose that you are an estate agent with a large number of houses on your list. The properties range from low-cost terraces to luxury penthouse suites and are located over a wide geographical area. If a buyer comes in and asks for a list of houses you might have in Croydon with three bedrooms, a garage and costing less than £45,000, you could locate it quite easily using Archive's built-in commands,

```
SEARCH
LOCATION$="CROYDON" AND
BEDROOMS<=4 AND PRICE<
45000 AND STATUS$="FOR
SALE"
```

A more elegant solution would be to use Archive's programming language to prompt you on the screen with prompts such as:

```
CLIENT'S REQUEST
HOW MANY BEDROOMS RE-
QUIRED? ____
WHAT AREA? _____
OTHER AREAS CONSIDERED?
____
HOUSE, FLAT, EITHER (Enter H,
F or E) ____ SIZE OF GARDEN
REQUIRED ((S)mall, (M)edium,
(L)arge, (N)one, (D)on't care) ____
PRICE RANGE? TOP PRICE IS?
_____
```

All of this input could be programmed both to give the prompts and to pass the parameters to the built-in commands, arithmetic and logical operations so that an appropriate search would be carried out automatically, resulting in a printed out list of all properties meeting the client's criteria.

Another procedure called OFFER could be written that

would automatically update the file and change the status of a house on the list from 'FOR SALE' to 'UNDER OFFER' if the client wished to purchase. This house would then automatically be eliminated from any searches made for further clients.

An interesting example of the power of a 'real' database program (as opposed to the non-programmable 'card-index' style programs that sometimes call themselves databases) is given in the User Manual. It uses the example of a club that charges an annual subscription and gives members six bi-monthly newsletters.

The database file consists basically of details of the individual members and whether he or she has paid his subscription. Procedures are written to automatically print mailing addresses (for the newsletter) and issue a subscription reminder whenever there is only one newsletter left to be sent out. Each time an address mailing list of labels is printed out, the number of issues left for the member to receive is decremented by one. Each time a subscription is received from a member, the number of issues to his or her credit is topped up by six. To show how simple this can be in the Archive programming language, we reproduce the procedure that accepts subscription payments and updates the records. Our comments are added in square brackets.

```

proc pay [name of the procedure]
cls [clears the screen]
let n$="x" [dummy value for n$ in
case n$ has been altered elsewhere]
while n$ <> "" [while not null]
getrec [procedure to find a specified
record]
if ok$="y" [valid record]
let issues=issues+6 [tops up newslet-
ters]
update [modifies record]
endif
endwhile
endproc

```

The procedures written in the Archive programming language are true procedures. Parameters can be passed to procedures. These parameters can be variables as well as literals and may be strings as well as numeric variables. Parameters can call other parameters and can even call themselves if necessary. This ability of parameters to call other parameters considerably eases the programmer's task. A general procedure

can be written, for example, to 'print' address labels on the screen using statements such as:

```
doline; title$+" "+fornam$(1)+" "+surname$
```

In this example, 'doline' is a named procedure and the rest are parameters passed to it. The procedure 'doline' can then be separately defined as:

```

proc doline; x$
print x$
endproc [to print on the screen]
Or:
proc doline; x$
lprint x$
endproc [to print on the printer]

```

## Problems

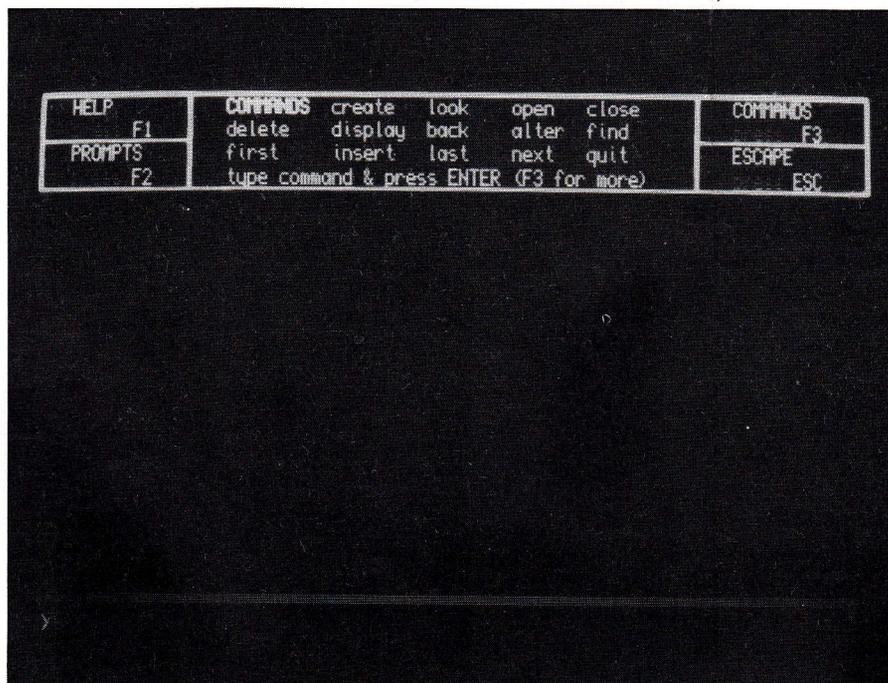
So far, our look at QL Archive has stressed its versatility and power. Criticism has been noticeably absent and now is the time to rectify that.

QL Archive is potentially a very powerful database program, equal in many respects to the well known market leader, dBase II. Pared-down databases such as Cardbox are inflexible and cannot run user-defined programs. They pale by comparison. Unfortunately, Archive is a new and (largely) untested product. The version made available for this review (version 0.3) was filled with a fascinating fauna of bugs; many of the documented commands and functions simply didn't work and there

were logical inconsistencies in those that did. For example, the FIND command to search the file for a specified string (e.g. FIND "pascal" to locate a book record) would locate "pascal" regardless of the use of upper or lower case letters, and would find the substring "pascal" even if it were in a field containing the string "Pasca-lissimo". The similar command SEARCH, as in SEARCH Title\$ "Pascal", would not work because the correct field name was TITLE\$, not Title\$. Similarly, while the command FIND "Z80" would locate a record containing the substring "Z80" in a field containing the string "Practical Microcomputer Programming: The Z80", it could not handle SEARCH TITLE\$ = "Z80" AND PUBLISHER\$ = "Sybex" because the title field consisted of the string "Z80 Applications". I feel that if a database is to have "intelligent" search facilities that can ignore upper and lower case and can find substrings in some cases, it should be able to do so in all cases.

## User image

Although, generally, the "user image" (that is, the way the programs appears to operate to the user) is simple and consistent, there are some curious inconsistencies. When a new file is created, for example, the command to initiate the creation of the new file is an "intuitive" CREATE, as in:



A blank Archive screen. Note common screen layout, with prompts at the top and command line at the bottom.

```

CREATE "FUNGUS"
SPECIES$
LOCATION$
SEASON$
CONDITION$
GRID REFERENCE
NOTES$
GROWTH$

```

Having entered all these fields, however (ending with the END-CREATE command), the variables for each field are entered by using the keys in the ordinary way. Each line is then terminated by pressing the TABULATE key, not the ENTER (RETURN) key. This is a most unnatural activity, especially for the proficient typist. Another curiosity is that when a file has been opened for editing, the whole file is written back to mass storage even if no record has been modified. Since the reading and writing operations are not particularly fast, this seems a real time waster; internal flags must surely know if modifications have been made or not!

## Documentation

Another major criticism has to be made of the manual. Although the authors have obviously tried hard to cover everything, and to illustrate the use of Archive through examples, they have not succeeded well. There are far too few practical examples. Too often, the approach is to "include a few simple examples, but the best way to learn is by using them (*sic*) yourselves" (from the chapter on Editing). Although the use of computer jargon (passing parameters, string variables etc) will be familiar enough to programmers, users new to computers are likely to find the explanations hard going.

Particularly poorly explained is the use of actual and "logical" filenames for files used in the database. Since Archive is able to operate on more than one "logical" file at a time, it is an important subject sure to leave many users confused by the skimpy treatment afforded in the manual.

Another example of the way the manual takes too much for granted is given by its treatment of global and local variables within Archive programs. Again, the concept will be familiar to most programmers. If you are not a proficient programmer, what would you make of this explanation?

"The variables used as formal parameters in a procedure are local

variables in that they are not defined outside the procedure in which they are first assigned a value. The following example may help to make the distinction clear.

```

proc demo; a,b$
print x,y$
print a,b$
endproc

let u=3 ENTER
let v$="text" ENTER

demo; u,v$ ENTER
print u v$ ENTER"

```

If the Sinclair QL becomes as popular as it deserves to become, we may hope for a plethora of "QL Archive Made Easy" and "How To Use QL Archive" books; the manual is the barest outline.

As for the numerous bugs in the program, it is difficult to make a reasonable judgment. The review program was release 0.3. The software writers, Psion, say that the release to be supplied with the QL will be version 0.5. Hopefully, the numerous commands and functions that 1) resulted in error messages, 2) did nothing, 3) dumped the user back in the operating system, or 4) caused the system to hang up completely will all be fixed. Probably, we cannot expect an improved version of the User's Manual, and that is a shame.

## Speed

As for a proper benchmarked review of Archive, that is beyond the scope of this review. Unlike, say, benchmarks for Basic interpreters, benchmarks for database programs are far from standardised and find it difficult to take into consideration the widely different capabilities of various offerings. In lieu of timed benchmarks, then, we offer the following subjective assessment which bears in mind other database programs on the market.

QL Archive is slow. Given the QL's Microdrives, this slowness was rather painful compared with comparable databases such as dBase II or Rescue. The number of bugs in the review sample was wholly unacceptable, but Psion is a prestigious company and one hopes that they will have fixed most of them before the product is released to the public.

## Conclusions

The QL Archive database program is provided with a very complete and powerful programming lan-

guage which is sufficiently similar to SuperBasic to make it extremely easy for any Basic programmer to create sophisticated procedures.

Archive is emphatically not a 'card-index' style tarted-up address book program. The user can use it that way if he wishes, but has the option of writing programs that will operate on numeric and string variables in the files to give formidable capabilities. Unlike several

## Archive is potentially a very powerful database program.

quite powerful (and expensive) databases, Archive is able to operate on more than one file at a time. It is regrettable that this aspect was so inadequately documented in the manual.

The Archive programming language, unlike, for example, dBase II's arcane and difficult language, is refreshingly easy to use. Not only are the statements virtually the same as those in SuperBasic, the programs can use global and local variables, named procedures can be defined and called, even within procedures of the same name, and all files are accessible, even those that are not the 'current' file. It is a highly interactive system that combines the best of Basic with fully procedural structured programming.

Hypothetically (ie, overlooking the bugs in the review sample of the program), Archive is one to rate alongside dBase II, but with a considerably easier programming language. Deficiencies in the user image consist mainly in the slightly inconsistent use of function keys, especially the use of the TAB key to terminate variable inputs when creating records. The 'intelligent' search facilities could certainly do with improvements.

Since Microdrives will never be as fast as floppy disks, it is important that the program be made to run as fast as possible - the review sample was not very fast - and wide acceptability will depend upon more accessible documentation. I feel sure that the greatest barrier to the wider use of this impressive software package will be its documentation. Digital Research succeeded despite its incomprehensible documentation. A smaller company like Psion might not find it so easy!

**QL****SOFTWARE**

# ABACUS

*Barry Miles tests Psion's spreadsheet*

Spreadsheets are an extremely powerful weapon in the hands of any person who needs to manipulate figures. We can include in that category not only accountants but anyone who has figurework to do whether they be an engineer, scientist, a clerk or any other person. Unfortunately the theory has grown up that spreadsheets are purely for financial forecasting but this grossly underestimates their capabilities.

In the course of their ordinary day to day work many people need to put figures onto paper. Conventionally they put these figures into rows and columns, which they then quite often need to add up both horizontally and vertically. The bugbear of this kind of work is the need which so frequently arises to change some of the figures and therefore to add them up again. This often occurs over and over

again until the person who is preparing the sheet is running out of patience and until the sheet itself is almost indecipherable.

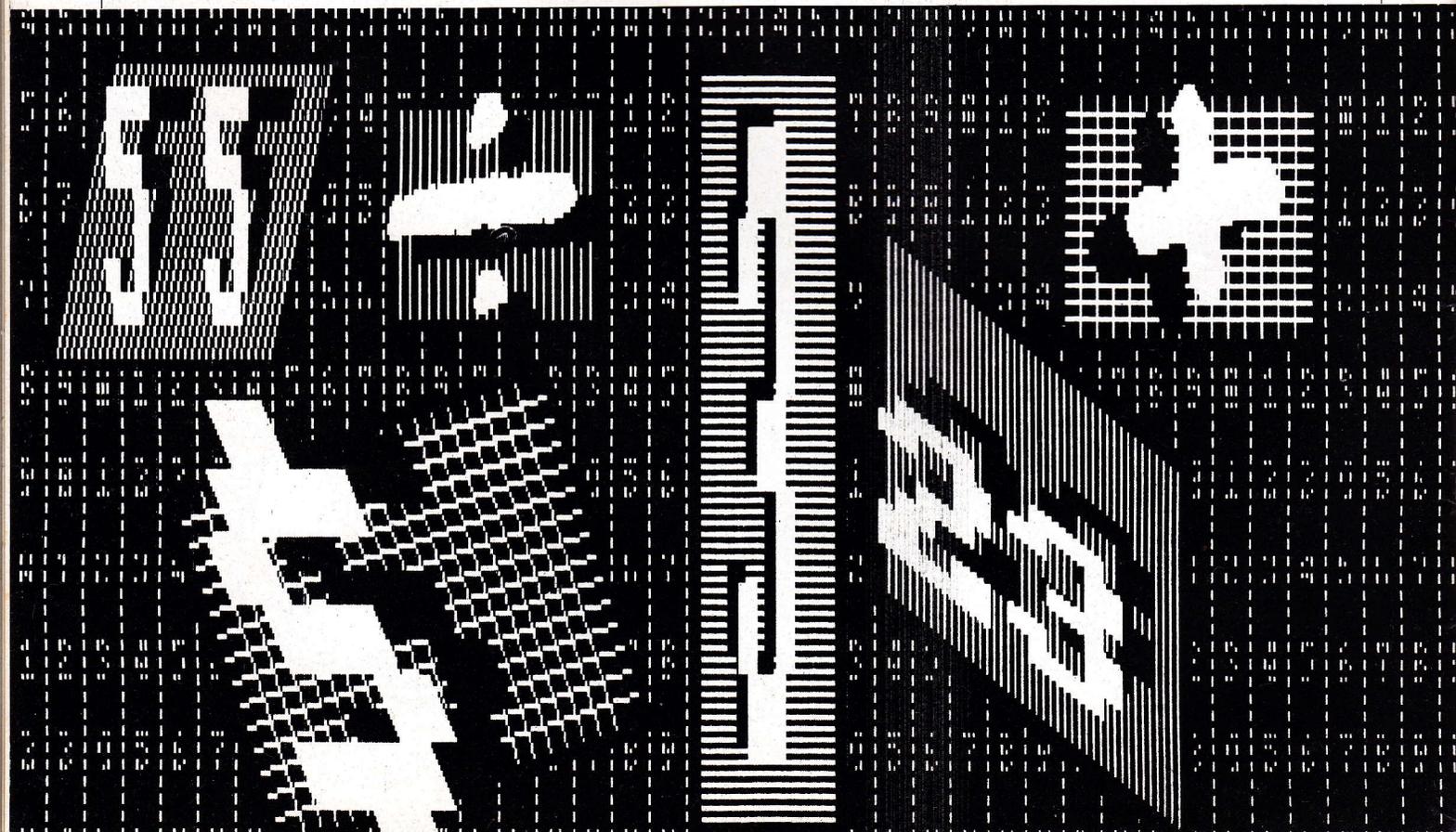
Computers, on the other hand, can rewrite a particular piece of memory with no recollection of what was there before. They don't get tired and they don't get irritated at the waste of time involved in repetitive work. It therefore follows that computerising this type of task is a very worthwhile activity.

The electronic spreadsheet has greatly eased the work of sort of people I have just described. The sheet is nothing more than a large electronic piece of paper arranged in rows and columns so that information can be inserted in boxes which are commonly called cells. Each of these cells can be identified by a 'map reference' using co-ordinates, so that you have rows

and columns and can refer to these normally by letters and numbers.

The very simplest use of the spreadsheet is to put in columns and rows of numbers and add them in the two directions reliably. To use this is rather like using a Grand Prix car to go shopping: you can do it but its not what the device is best at!

Stepping away from this simple capability one can insert extra rows or columns into the spreadsheet. Anyone who has used a large sheet of paper will know how frequently they wished they had allowed extra space for an additional column or row of figures. Normally a spreadsheet permits not only the insertion of rows or columns at will but it also permits you to move them around so that you can reassemble your data both vertically and horizontally; indeed it is also possible to sort rows or



columns into order in some spreadsheets.

More extensive use of spreadsheets starts when you begin to put formulae into cells so that the figure which is to appear in a cell depends on the content of other cells. Formulae may be simple or complex and the way in which the spreadsheet is designed varies substantially. Some spreadsheets will evaluate formulae by moving through them from left to right, whereas others will use the full hierarchy as understood by mathematicians. The QL spreadsheet Abacus is in this latter category.

Once you prepared your table of figures you will want to print it out. Most spreadsheets permit you to print out sections off the sheet itself while others, like the QL Abacus, permit you merely to print out the display you see on the screen. Naturally you will want to save the contents of your spreadsheet from time to time and this facility is always provided.

The very substantial hype which accompanied the launch of the QL has led to very great expectations of the software. Readers will therefore want to know to what extent these expectations have been fulfilled. The answer is: partially!

## **Using Abacus**

Abacus is very simple to load: you merely reset the machine with the Abacus Microdrive cartridge in the left hand slot and tell the QL

whether you're using a TV or a monitor. The machine looks for a program called Boot on that cartridge and proceeds to boot up the spreadsheet with a suitable display showing that this is in progress. The screen then clears and the standard Psion package display appears. Alternatively, you can type LRUN MDV1 BOOT and hit ENTER.

It is immediately obvious that considerable attention has been paid to the user interface. At the top of the screen you are given a list of the functions which can be accessed by hitting one of the five function keys on the QL.

F1 will get you into the help facility, available on all packages delivered free with the QL. Immediately you touch F1 a help screen is loaded from the Microdrive cartridge. The help screen obtained is 'context sensitive' - you get the help screen relevant to the activity which you are trying to carry out at the time. There is a hierarchy of help screens so that if the screen you are presented with does not give enough information you can usually get further help. In addition, when you leave the help facility by pressing the Escape key (again a very common function with the QL packages) you find that you pass back up through the hierarchy of help screens at which you have previously looked. This means you are always well aware of exactly where you are in the program. Some lesser programs

have you darting about so that you have no mental picture of what's happening.

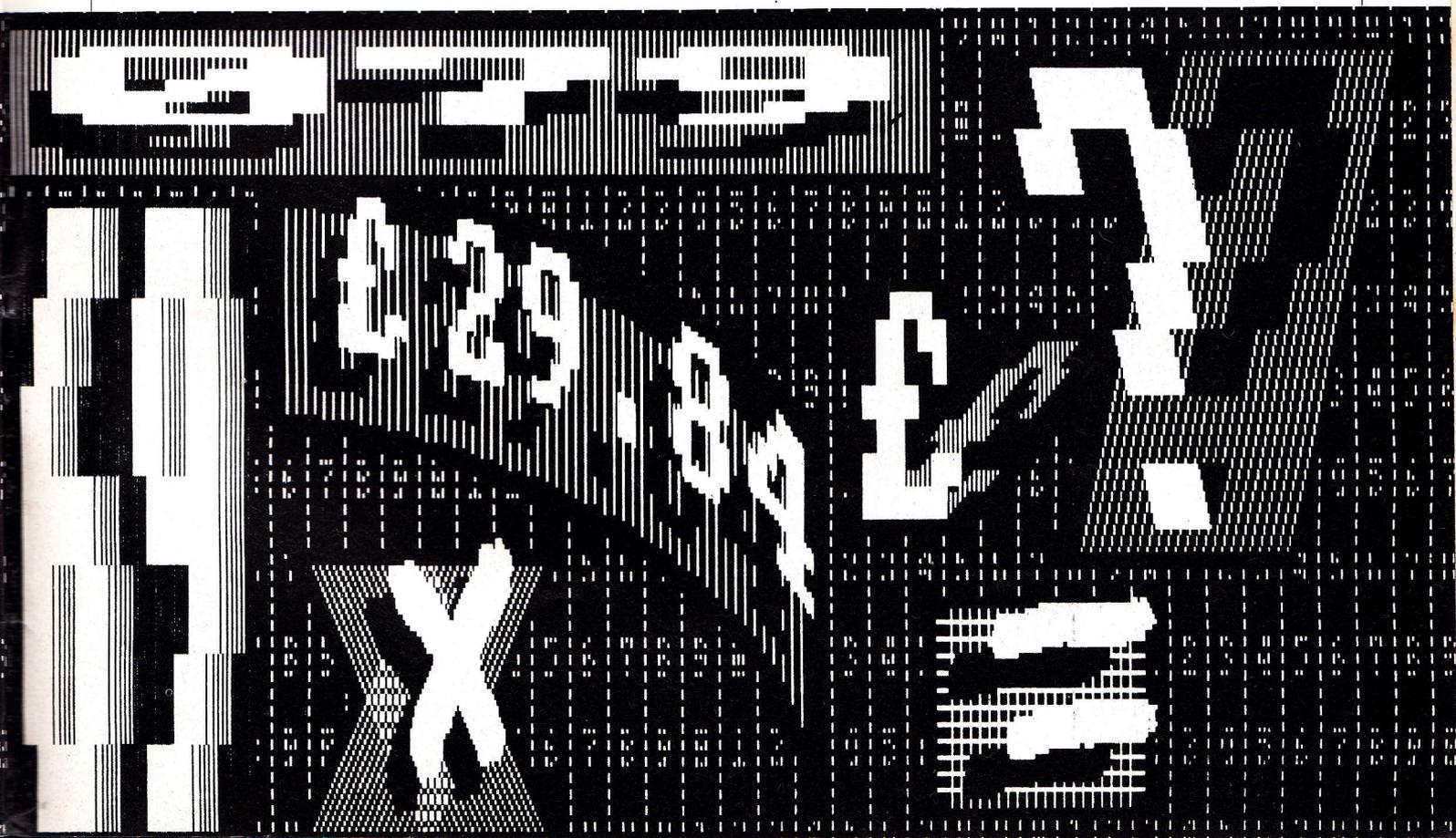
Function key 2 lets you eliminate the prompts which are normally on the top few lines of the screen thereby enlarging the amount of screen display which is available for data. As soon as you become familiar with the package you'll be likely to use this quite a bit. Hitting F2 a second time will bring back the prompts whenever you need them.

F3 gives you access to the Commands with which you can really start to exploit the power of Abacus. The moment you hit F3 a list of various commands to choose from appears on the screen. You select these by typing the initial letter. If you change your mind and wish to cancel this command before it is implemented you hit ESC.

F4 moves you from one part of a split screen to the other. The split screen is somewhat misleadingly referred to as a 'window' in the literature.

F5 is the GOTO function key; whenever you press this the prompt 'A1' appears, Abacus making the reasonable assumption that you will frequently want to go the top left hand corner of the sheet. You merely type in any other co-ordinate and the cursor moves to that particular location, which is very convenient.

The bottom two lines of the screen are defined as the status area. This gives you information



about the current state of the spreadsheet. It is rather important as the memory available is surprisingly limited in Abacus so you need to know how much memory you have left. This is measured in kbytes; you start off with only 15k and end up with 0 as you fill the sheet. In practice you will find that you can put in somewhere less than 1100 pieces of information before the sheet becomes full.

### **The display**

You do have the choice of deciding (when loading any package into an empty QL) whether you will be using a monitor or a TV. The brutal fact is that you are very much better off with a monitor than with even the highest quality television: the definition of a domestic TV set is not sufficiently good to sustain a display of 80 columns without a considerable degree of eye strain.

Dropping the display down to 64 columns (which you can easily do with the 'design' command) is rather tiring on the eyes and many people will end up using merely 40 columns which does not give a great many figures available for immediate viewing on the screen. Some users might be deterred from buying a monitor when they have managed to buy the QL and four packages so cheaply. This strikes me as rather a false economy.

Information on the status line includes the extent of the grid, in other words the furthest cell from A1 which is so far most allocated information. The spreadsheet is fairly large: it offers 64 columns labelled from A to BL and 256 rows.

In most spreadsheets the information that you are seeking to put into the cell is shown on a prompt line. In Abacus this takes place within the control area. Editing the data as you input is relatively simple: you simply hold down the shift key and hit cursor left or cursor right to delete characters to the left of the cursor or beneath the cursor. You may find it aggravating that the designers have seen fit not to include a delete key on the QL. I certainly do!

The machine understands that you are about to enter text by the fact that you remember, hopefully, to insert inverted commas before typing the text. In the absence of those inverted commas it assumes

you are typing in a formula.

### **First impressions**

Generally the first impression given by Abacus is favourable. There is pleasant use of colour in the display and the prompts are clear; the appropriate prompt on the prompt line lights up according to what you are doing so if you start to enter text, for instance, the 'Text' box lights up.

But some caution is necessary when using Abacus. This arises fundamentally from the fact that the spreadsheet could fairly be described as a 'programmer's spreadsheet'.

There are two distinct schools of thought about spreadsheet design. The first spreadsheets – such as VisiCalc – assumed that the person who was entering the formulae would also enter the data. In fact it was assumed that the formula and the data would be entered at more or less the same time, rather as an artist builds up a picture. As the spreadsheet was built further and further and more and more data was added, the programmer would think of new calculations to perform and immediately add additional formulae to perform them.

This approach is fine provided the person using the spreadsheet knows exactly what he or she is doing. More recent spreadsheet designs have lent towards the idea that the person who writes the formula will have the skills to design the spreadsheet but that later, other people who do not necessarily have those skills will insert data into this proforma. Spreadsheets of the latter variety contain many protections to make sure that formulae are not accidentally overwritten by people trying to enter the data.

Abacus undoubtedly falls into the first of these categories. There is complete freedom to overwrite any formula at any time quite without hindrance, hence the term programmer's spreadsheet.

### **Editing facilities**

The editing facilities within Abacus are very attractive – in fact they would do credit to a word processing program! Interestingly, you are normally in insert mode at all times. This is often a time-saver and to some extent offsets the absence of a delete key from the keyboard. The left and right cursor keys will move you left and right, one character at a time, and when shifted allow you to move a word at a time, to the next space or comma.

If you hold down the control and hit the left or right cursor key, the characters to the left or right of the cursor are deleted one at a time; if you press the up cursor key you will get to the beginning of the editable text and hitting the down cursor will move you to the end of the text. Holding down the control key and hitting 'up cursor' or 'down cursor' will delete all text to the left or right of the cursor immediately. Mastery of these commands comes easily with frequent use of the spreadsheet and once achieved makes editing a simple and comfortable process.

You need to be alert to the dangers of losing all your work by the injudicious use of the 'zap' and 'quit' commands. Zap deletes the spreadsheet from the Abacus program but leaves you in the program, ready to start off some fresh work. Quit on the other hand not only loses your spreadsheet but also the program as well. The reason you need to be cautious is because hitting F3 gets you into the command mode, and Z gets you into the zap command; there is a warning on the control line that you will lose all your text if you hit Enter, but that is the only warning that you get. Similarly, if you are about to quit there is a warning that you are about to lose your spreadsheet. Now the point about this is that frequent use of any particular piece of software causes very rapid key-striking to become the norm. There is a danger therefore that you will hit F3, Z and Enter all in one rapid movement, only to find that you wish you had not done so because you had forgotten to save the file onto the Micro-drive cassette!

The problem is worsened by the fact that the Quill program automatically gives you the chance to save your file before you kill off that file. So people who get used to using Quill before they get used to Abacus may pay an extensive penalty. The point is that the work that goes into the average word processing file is likely to be less than the amount which goes into preparing a complicated spreadsheet and putting in the data. Therefore you may be destroying five hours' work with a moment's thoughtlessness. Some warning – flashing, perhaps – for potentially devastating commands should be included.

### **Other commands**

In other respects, though, Abacus is well designed. It holds the numbers in its memory to 16 significant digits but you can display your

Abacus arrives at a cell in which this command has been given, it prints in the prompt line the text which has been put into this command. As soon as the user answers, the recalculation continues. The effect is to make your spreadsheet into an interactive device, which can be used by people who do not lay out the formulae. This is extremely attractive and allows you to design multi-purpose spreadsheets.

You can head columns with the words 'January' to 'December'; in sequence automatically. There is a month command and you can even arrange offsets so that you can start any month in any column and know that it will go from July, say, to December and then resume at January, continuing across the columns in a sensible fashion. This saves an awful lot of typing and editing.

Accountants and other financial analysts will welcome the fact that the 'internal rate or return' and 'net present value' commands are extremely powerful – they even allow you to discount at intervals of less than a year, which is a pretty advanced feature.

## Working with other software

The 'import' and 'export' commands are the ones which permit the transfer of data – albeit slowly – between Abacus and the other packages in the Psion series. Thus you can bring data from Archive, the database, or export to Archive or indeed to Easel, the graphics package.

This is done by preparing an 'export' file in Abacus, saving it on a Microdrive cartridge and then loading the appropriate package in order to use that export file. This is tedious business bearing in mind the slow speed of the Microdrives and therefore the sensible thing to do whenever you are planning to export is to export *everything* you may conceivably want into a number of files so that once you have loaded the other program you are able to make best use of the data you have prepared.

## Good and bad points

The advantages of Abacus are that there is a variety of special com-

mands not present in all spreadsheets that make this more useful than some. The ability to use headings as labels and then refer to labels rather than the column or the row makes it very easy to set up your formulae even without looking at the sheet at the time.

The option to have 80, 64 or 40 column displays is good, as is the way text is allowed to spill into neighbouring column(s) until they are needed – a real plus factor.

The row and column commands, used carefully, help you to prepare the most common spreadsheet applications and are very convenient. The ability to send control codes to your printer so that your printout can be formatted neatly is attractive.

It's nice to be able to transfer to and from the other Psion packages, even though this is a pretty tedious business, because the Microdrives are so slow.

As far as usability goes, the on-line help facility reduces dependency on the manual to a minimum and speeds up learning time. The comparative uniformity of display and function of this package with the other Psion packages for the QL shortens the learning time for the whole set of four packages.

The intelligent attempts made by Abacus to foresee the nature of your commands helps speed spreadsheet preparation and the availability of a command which permits the user to answer questions upon which the spreadsheet activity will depend is a welcome one.

On the minus side is the slow speed of the Microdrive cartridge system which makes loading and saving data very tedious. The very primitive split screen facility is very disappointing particularly in view of the amount of publicity given to multi-tasking and window operation of the QL itself.

A major defect is that when you move the cursor rapidly by holding down the cursor key, the screen is not updated until the movement is over. If you're moving across the screen to a particular column, it's very easy to fly right past it!

The consolidation feature is somewhat primitive. One would have expected to be able to produce a single total sheet from a large number of files automatically. However, the limited capacity of a Microdrive cartridge – 120k – prevents this.

I was disappointed to find that

you can't construct your formulae by pointing the cursor at the cell concerned, a common spreadsheet feature.

The lack of cell protection is on one hand an advantage, giving you the ability to change your spreadsheet very rapidly, but it is also a disadvantage because it gives you the ability to change unintentionally.

The lack of a delete key on the keyboard is really irritating – you have to use two keys instead of one.

Despite the QL's 128k of memory, Abacus has a remarkably small capacity: 15k is a very small amount of space and you will find that if you fill approximately 1100 cells you have used all the memory that you have available.

## Conclusion

Abacus is a run-of-the-mill spreadsheet which doesn't push back the frontiers of design of this type of product. It contains a number of good features and some unattractive ones. Its use of memory is profligate and this probably stems from the fact the QL itself and its operating system were not available to the software designers when they designed this package so it was necessary to develop the package on another machine and then cross-compile it for the QL, something notoriously expensive in memory terms.

The great disappointment which will no doubt come as a shock to many people who have read the QL brochure is that multi-tasking for a number of programs operating simultaneously is a feature of the QL but not of the packages which accompany it! A careful reading of the QL brochure shows that this multi-tasking and windowing capability is only claimed for programs operating in SuperBasic. The brochure does not make it clear that the Psion packages do not operate in SuperBasic.

I can't help making comparisons with spreadsheets operating on 8-bit computers with only 64k of memory available which have vastly greater memory capacity than Abacus and indeed operate just as quickly. Let's hope that future versions of Abacus will make more effective use of memory, will be faster and will permit the QL to at least have the graphic capabilities of the Easel package working at the same time as Abacus.

# MIMI MICROCOMPUTERS

*The Ultimate Choice....*

for  
Software  
Adaptability  
Flexibility  
Ease of Use



**BRITISH MADE**  
\*  
**BUY**  
**BRITISH MICRO**



AHEGOTRON GROUP COMPANY



**BRITISH MICRO**

Penfold Works, Imperial Way, Watford, Herts WD2 4YY, England. Tel: (0923) 48222 (Marketing 43956) Telex: 946024