December/January 85p

THE ONLY MAGAZINE FOR THE

QL USER

Unlocking Sinclair's Supermicro

QL QUARTET
Four new languages compiled and interpreted

PSION CHESS
A Grand Master for the QL?

QL vs PLUS 4
"Gloves off" battle or walkover?

FIRST QL GAMES
Exclusive Reviews

THE NEXT GENERATION:
An audience with Sir Clive

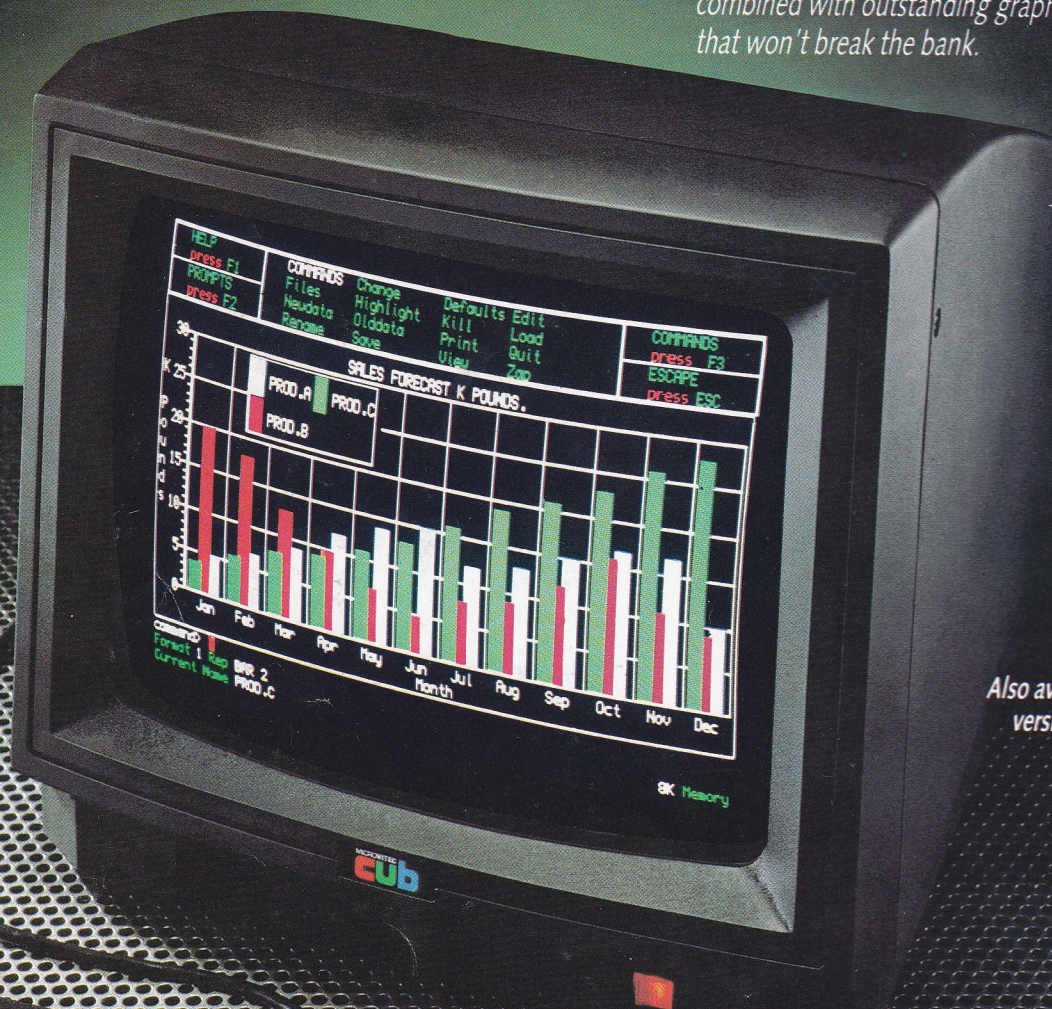
An EMAP Publication

AVAILABLE FROM
High Street Computer Retailers
and branches of W. H. Smith, Boots,
John Lewis Partnership, Laskeys, Currys.

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

December/January

Editor Paul Coster BSc
Technical Editor Paolo Baccanello
Editorial Assistant Shirley Eborn
Art Editor Paul Carpenter
Technical Consultant Adam Denning
Associate Editor Peter Rodwell

Advertising Manager Phil Baker
Advertising Production Yvonne Moyser
Sales Executive Tracey Keighley

Publisher Alfred Rolington

Contributors

Sid Smith, Leon Heller, Nicky Trevett, Jeff Fenton, David Creasey, Alec Short, C Harker, Leslie Green.

Editorial 01-278 6767

Advertising 01-278 6995/6/7/8

Competitions

The winners of our Oct/Nov competition were W M Waite of Stoke-on Trent (who wins the Microvitec monitor), and G M Darling from Ruislip Manor in Middlesex (winning Opus Supplies' monitor). Congratulations to them both.

Information

First a few apologies arising from the last issue:

The review of last month's books was by Gordon Taylor who we mysteriously called Nicky Trevett (she of this month's reviews) – sorry!

The caption on p.23 regarding the pictured monitor was wrong, but confused readers are referred to this month's reappraisal.

Another matter revisited this month relates to GST's assembler and OS – pages 26-27 set the record straight.

Finally a note from Microvitec for all CUB QL-compatible users. Owners of models 1451/DQ/3 and 1451/MQ/3, with serial numbers between 85692 and 103317, are asked to phone David Kaye on Bradford 726500 to arrange a free up-grading service which may be necessary.

CONTENTS

STOP PRESS

As from the next edition, due to response QL User will be on sale monthly.

So, look out for the February issue – in the shops 21st January.

ADD-ONS FOR YOUR QL LOOK NO FURTHER THAN TECHNOMATIC

We've built our reputation on providing a reliable service

- ✱ Wide Range
- ✱ Competitive Prices
- ✱ Deliveries from stock
- ✱ 12 months guarantee on all printers
- ✱ Accessories and consumables readily available

DOT MATRIX PRINTERS

EPSON

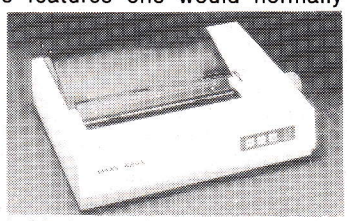
This ever popular range of dot matrix printers set the industry standard and provide reliability and quality 'second to none'. The budget RX80T & RX80FT have 100cps and all standard printing and graphic functions with tractor feed on RX80T and both tractor and friction feed on the RX80FT. The deluxe FX80 has all the above, as well as a 160cps, buffer, programmable characters etc. For wider paper use — upto 15″ — the RX100/FX100 are ideal.

RX80T £210(a)
RX80FT 230(a)
FX80 £312(a)
RX100 £345(a)
FX100 £435(a)

EPSON Serial Interface c/w 2K buffer **£60(c)** (**£52.50** if bought with the printer)
Serial Lead QL/EPSON: 2m long **£12(d)**; 1m long **£8.50(d)**
Spare Ribbon Cartridges: RX/FX 80 **£5.00(d)**
RX/FX 100 **£11.50(d)**
Dust Covers: RX80 **£4(d)** FX80 **£4.50(d)**
Paper Roll Holder **£17(b)**
Tractor Attachment for FX80 **£32(b)**

KAGA

This new range of printers with EPSON compatible control codes feature NEAR LETTER QUALITY print using a 23 × 18 matrix in addition to the features one would normally expect from good quality dot matrix printers. Text modes include Normal, Italic, Enlarged, condensed Super & Sub script and Proportional spacing. Defined characters can be placed in a rom to give personalised print.



An integral 3K buffer, both friction and adjustable tractor feeds with built in paper roll holder, etc, etc make these superb 'value for money' printers unique.

KP810 (80col) £249(a) **KP910 (156 col) £369(a)**

Kaga Serial Interface c/w 2K buffer **£85(c)** (**£77.50** if bought with the printer)
Serial Lead QL/Kaga: 2m long **£12(d)**; 1m long **£8.50(d)**
Spare Ribbon Cartridge **£6.75(d)**
Enquire about the special font roms.

DAISY WHEEL PRINTERS

JUKI 6100 £340(a)

Serial Interface Juki **£65(c)** Tractor Feed Attachment **£99(a)**
Single Sheet Feeder **£199(a)** Ribbon **£2.50(d)**
2K additional RAM for Juki buffer **£7.50(d)**

BROTHER HR15 £340(a)

Serial Interface integral with the printer.

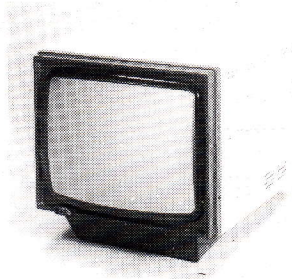
Tractor Feed Attachment **£90(a)** Cut Sheet Feeder **£199(a)**
Ribbon **£4.75(d)**

Plain Fanfold Paper with extra fine perforation (Clean Edge)
2000 Sheets 9.5″ × 11″ **£13(b)**; 2000 Sheets 14.5″ × 11″ **£18.50(b)**
Serial Lead for Daisy Wheels £8.50(d)

SOON TO BE IN STOCK
PARALLEL INTERFACE FOR THE QL £42(b)

MONITORS

The British made MICROVITEC 14″ RGB Monitor with high clarity 85 character display offers a consistent and reliable performance.



MICROVITEC 1451DQ3 £239(a)
Slanting Plinth for Microvitec **£8.50(b)**

KAGA SINCLAIR QL Super Hi Res 12″ RGB Monitor
A high quality monitor with anti glare screen **£273(a)**

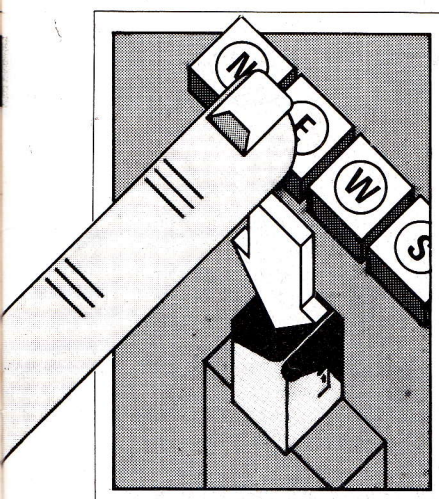
ALL MONITORS SUPPLIED WITH SUITABLE LEADS

TECHNOMATIC LTD

MAIL ORDERS TO: 17 Burnley Road, London NW10 1ED
(Tel: 01-208 1177 Telex 922800)
SHOPS AT: NW London: 15 Burnley Road, London NW10 1ED
⊖ (Dollis Hill 2 mins walk, ample car parking space)
West End: 305 Edgware Road, London W2. Tel: 01-723 0233
(Near Edgware Road) ⊖

01-208 1177

All prices exclude VAT. Please add 50p carriage unless indicated as follows:
(a) £8: (b) £2.50: (c) £1.50: (d) £1.00.
add 15% VAT to the total order value. For fast delivery telephone your order quoting VISA or Access card or official order number. (Minimum telephone order £5).



NEWS • NEWS •

Another batch of hard-hitting reports from QL User's 'ear to the ground' – Sid Smith

QLN

News Editor Wanted

Sinclair and Psion have hired a professional journalist to run their bi-monthly QLUB Newsletter.

Thanks to his efforts, members – who indignantly recall the four loose sheets which masqueraded as Issue One – can now read interviews with QL personalities and inside-info about forthcoming products (which this news area will shamelessly plagiarize).

Life, however, isn't always easy for the earnest scribbler, one of the most respected journalists in the industry, and he's obliged to explain to potential Newsletter contributors that critical remarks about Sinclair, Psion and their products won't necessarily be welcomed by his employers.

One freelance, commissioned to describe his near-daily use of Quill, was forced to abandon the project after discovering the impossibility of 1500 words on the wonders of QL word-processing which made not a mention of numb keyboards and slow screen handling.

However, at £90 per 1000 words, I'm still not sure I couldn't have found a way.

Monitoring Controversy

One of the more obscure firsts clocked up by the QL is Sinclair Research's agreement on badging deals with two hardware companies.

Sinclair's attitude to peripheral manufacturers has always been tolerant. The company has never worried that small firms, like birds behind the plough, might pick up scraps from its passage, deciding instead that what mattered about modems and printer interfaces was that they existed, not that they were made by Sinclair.

However, the process begun with software for the Spectrum has now extended to hardware for the QL; MBS has already issued its 'Sinclair' monitor (made by Kaga of Japan), and an equivalently-endorsed modem, complete with matt-black go-faster QL fluting, will shortly appear from OEL of Cumbria.

There's no mystery about this choice of OEL as modem manufacturer By Appointment To Sir Clive; the company has probably sold more of the devices than anyone else in the UK. BUT how does Bradford's own Microvitec, Queen's Award to Industry-winner, government subsidised for sales into schools, and itself manufacturer of an all-black QL monitor, feel about Clive's patronage going to a Japanese device?

'We're not bothered,' said the company's Tony Fall. 'Our monitor has a bigger screen, the same resolution, and everybody tells us it's the best around. The Kaga one is more expensive as well.' (In that case why didn't Microvitec get the Sinclair deal?)

'They offered it to us, and we'd have liked to go along with them. We just decided that the terms they offered weren't financially attractive.'

Birth Of A Modem

Details of the most important QL peripheral in the history of the world (I'm prejudiced, though) can now be revealed.

The Sinclair/OE modem, due for imminent release, is part of a three module communications package which allows applications including links to a DEC mini, and down-the-phone connection to a dedicated QL area on Prestel.

The whole tri-partite package is called QCOM. Its base module is a controller and terminal emulator (ie, a device which allows the QL to act like a terminal linked to a larger computer) called QCON; in the upper surface of QCON is a plastic plug which can be removed to allow a second module, the QMOD modem, to be stacked on top; a final QCALL module can likewise be stacked atop the modem, giving it an auto-dial and auto-answer capability.

The QCON module contains the software heart of the system. With this bare unit connected, and its partly microdrive-based software installed, the QL can be linked to large DEC computers via the DEC VT100 terminal emulator standard, or to a modem of the user's choice. QCON runs all the modem control software, including bi-directional buffers to cope with the inadequacies of the QL's serial ports.

The bare QL has been widely criticised for allowing only a single baud (bits per second) rate for both serial ports. The new unit will permit baud rates of 75, 300, 600, 1200, 2400, 4800 and 9600 to be independently set for the two ports, its most important application being implementation of the 1200/75 Prestel rates.

QCON also supports the XON/XOFF handshaking protocol and the VTS/RTS protocol – whatever they may be. It boasts an RS232 port for access to standard modems, and is powered from the QL's power pack with a lead then running to the computer itself.

The module allows the phone transmission of any unprotected mdv file, and includes a data encription capability. Great ease of use is claimed for the device, its more awesome applications being invisible until invoked by the menu-driven command structure.

No firm price has yet been fixed, but a figure between £90 and £100 seems likely.

The modem is a very straight-forward unit, thank goodness, giving either 1200/1200 or Prestel-style 1200/75 baud rates. No 300/300 rate is installed, since such provision would increase costs merely for the sake of a few one-man bulletin boards which are always engaged anyway.

The QMOD modem has a telephone extension socket for manual dialing, but if you want to get into the really fancy stuff then the final module, QCALL, allows autodial and autoanswer. This latter facility is a real hacker's delight, permitting incoming calls to be accepted, even in the user's absence, and to trigger pre-programmed activity from the QL; its more legal applications might include

This one's specially adapted to take pirate software.

receiving data, printing it out, and then – via the autodial facility – passing it on to a third QCOM unit. £50 has been mentioned as a possible price.

Quill II

The new version of QL Quill won't be available until after Christmas. Despite the hopes of Psion MD David Potter, quoted in our last issue, an upgrade of the 'slow' Quill version I missed its September deadline; a Psion executive has explained that the new version is presenting unexpected difficulties.

After signing his name in blood to all kinds of hair-raising promises, your reporter has been given sight of a pre-pre-pre-production version of Quill II, but is allowed to say almost nothing – except that the program, to be issued to new owners and QLUB members only, should be altogether splendid.

Equally desirable is the new Psion Chess game, reviewed in this issue. Psion *will* release more QL material, says Software Manager Charles Davies, but stops short of saying what it might be.

Has he any idea why there is still so little games software available for the machine? 'There just haven't been the tools around so far. At least there are a few assemblers available now, but even that's not the complete answer; most people writing for Z80 machines have been working with disk-based systems of some kind, and it takes a long time to set up new hardware like that.'

'If Imagine were still around, they'd be well-placed because they had Sages, but very few people are in that situation.' (One of the few QL games known to be in the offing is the ex-Imagine Bandersnatch 'megagame', now acquired by Sinclair and expected out in the first quarter of '85.)

Davies wasn't surprised to hear that the only decent non-Psion games we'd come across were two adventures.

'Adventures will do very well on the QL, but you won't get too many fast arcade-style Invaders routines – simply because the QL's screen handling is so difficult.'

'A high resolution image, in which a small part of the screen moves about, would be OK and I wouldn't be surprised if somebody was doing a super-dooper version of Manic Miner, for instance, or The Hobbit. But although the 68008 is slightly faster, you can't compare moving 32K of QL screen memory with the 6 to 7K of the Spectrum. Games like Defender, requiring scrolling, would be out of the question.'

So if, by some remote chance, Psion were to move games over from the Spectrum, which ones would they choose?

'Things like Tennis, and Scrabble,' said Davies carefully. 'And not things like Chequered Flag.'

Processing Power

An add-on which gives the QL access to the huge store of software written for the CP/M operating system has been announced by QL Plus Ltd.

An extension card containing a Z80 processor and 64K RAM which plugs into the machine's main external bus should be on sale before Christmas for the remarkably low price of £199.

John Fuller, MD of QL Plus, says, 'There are literally thousands of business, educational and games programs available under CP/M. That isn't true of any of the other operating systems available for the QL. And unlike the four Psion packages supplied with the machine, major CP/M programs such as dBase II and Wordstar have been proven over a period of years by millions of users.'

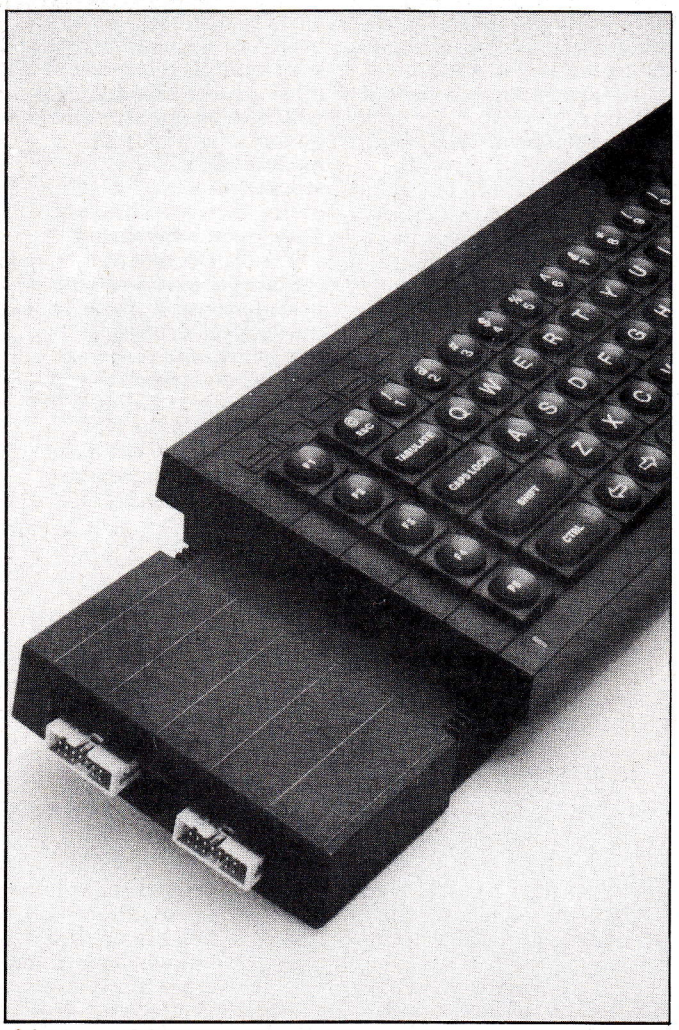
Giant retailer W H Smith is also involved in the venture, converting CP/M software onto microdrive and raising speculation about High St distribution for the card.

The link between the Z80 second processor (or, third, to be pedantic) and the QL has partly been engineered by the ubiquitous Tony Tebby who is himself confident of the appeal of the new device.

The 4MHz Z80 will run in parallel to the QL's 68008, with the latter being used for extremely fast I/O operations. The additional 64K RAM offers the prospect of a capacious print buffer, and a reduced reliance on microdrive accesses.

Two 8-bit parallel ports on the card will be software configured as a mouse controller and a parallel printer interface, though the company intends to release alternative software which will allow them to be used as a disk interface and robotics controller.

John Fuller claims that his product will be the most potent argument so far for persuading small businesses to invest in a QL . . . He could well be right.



Of mice and CP/M! Add-on and on and on . . .

"Dragging Their Feet"

Sinclair Research intends to release a microdrive cartridge containing around 58 extra SuperBasic commands. But the author of the extension is threatening to take his product to another publisher.

Plans are well advanced at Sinclair for the early release of the cartridge, the original intention being to have it in the shops for Christmas. However, the man who wrote the extra commands is the fearsome Tony Tebby, author of QDOS and scourge of Sinclair; he's so annoyed by what he describes as 'unforgivable feet-dragging' at the company that he's already had discussions with two other publishers about an alternative deal.

'Sinclair have documentation which is ready for the printers, and software which has been used by testers without significant difficulty for over a month. And yet they've just told me the material will be delayed for a further five weeks. It's absurd.'

The extra commands will include all kinds of goodies, such as direct file handling from BASIC and a full screen editor, which Tony was discussing in our last issue. Many of the commands, he says, would

have been included in the standard QL if the hardware had been delayed in development for another few weeks; they are also likely to appear in the QDOS upgrade promised for '85.

'I've had meetings with Sinclair in the last week,' he told us, 'and things are looking bad. Somebody needs to go in there with a very big boot.'

Pi In The Sky?

CST, manufacturers of the Q-Pi centronics printer interface, have announced a second QL device. Their Q-488 interface enables the computer to exchange data with the numerous scientific and laboratory instruments which use the IEEE-488 protocol.

The company already markets a high-quality IEEE interface for the BBC Micro, and feels that the Sinclair machine will likewise benefit from access to devices such as voltmeters, oscilloscopes and frequency counters.

The Q-488 responds to any high level language installed in the QL (it says here), incorporates an on-board QDOS device driver, and can handle up to 16 connected devices.

CST promise further QL add-ons, though they wouldn't tell me precisely which; 'to equip the computer for serious industrial and research work'.

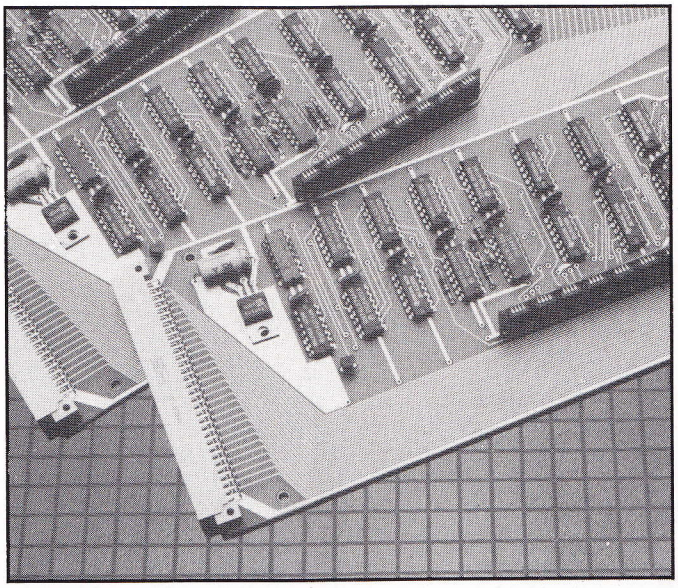
The Q-488 costs £170 (ex VAT), and is available from CST on (0223) 323302.

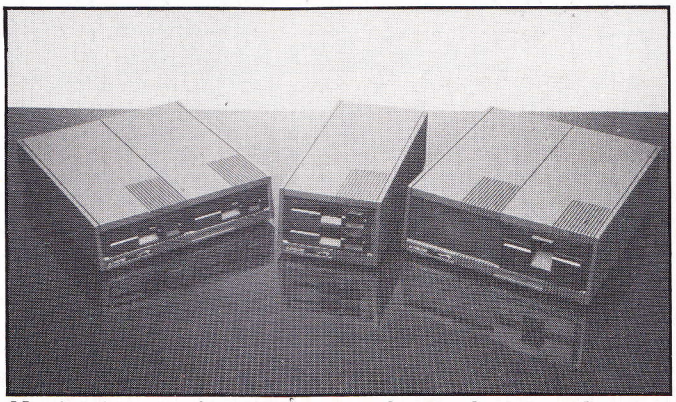
Quest For The Engram

Right on schedule, Quest have released their range of disk systems and RAM extensions.

Firmly establishing themselves as the biggest peripherals maker for the QL, Quest unveiled Shugart 5.25" floppy drives of 200K, 400K and 800K capacities, with corresponding prices of £249 to £599.

The company also plans a 7.5Mb (formatted) Winchester hard disc system for £990, and have exhibited a staggering 2 Gigabyte laser disk for the QL at a price of 'less than £30,000'. (How can we resist?)





No running out of memory now as Quest pile it on and wrap it all up in a sleek black executive console.

And Quest have leaped into the increasingly competitive RAM extension market with a range of devices offering 64K (£99), 128K (£159), 256K (£299) and 512K (£499). The two product ranges will be important for liberating the QL from its untrustworthy microdrives. The RAM extensions will relegate microdrive access to an initial booting up and final storage of files, while access to disk drives will please potential business users of the machine.

Speaking of which, Quest have released a QL accounting suite called Tally. The program will run under both CP/M 68K and QDOS, and consists of Sales Ledger, Sales Invoicing and Stock Control, and costs £99. A version called Tally II, costing £50, covers Purchase and Nominal Ledger.

Modules integrate not only with themselves but with all four Psion progs, and if the Stock Control package is any indication then Tally should be something special.

Spot The OS

By the time you read this, ICL should have launched its Personal Workstation device, built around the QL board.

Twin microdrives and the Psion Xchange suite on ROM – all of which explains the sighting of a twin microdrive computer down at Psion HQ.

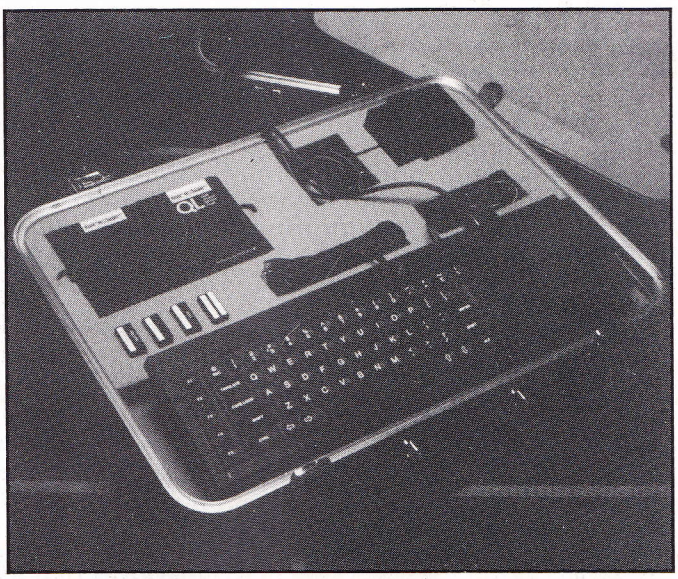
It also explains the heavy hints from Sir Clive about another company employing this technology.

The most intriguing question in advance of the launch, however, is which operating system the device will employ. Nigel Searle has told your reporter that ICL haven't used QDOS; development schedules for the two companies apparently didn't synchronise.

Sir Clive, in turn, told me that ICL has gone for GST's 68K/OS; GST says it hasn't.

According to yet another insider, ICL have produced their own OS, but have gone through hell in the process.

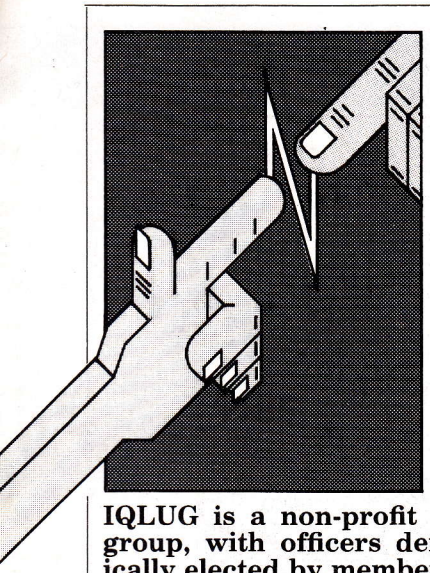
Whatever happens, the already complex world of 68008 operating systems looks set for a further twist.



Want a portable QL? Well, this new case from Management Science Ltd costs £69 (plus £4 p&p) and lets you carry the QL, power supply, manual, microdrives and cables with one hand.

The colour scheme is guaranteed to match Sinclair's own (ie, it's black), and the case is constructed with a rigid framework to shrug off the knocks.

Management Science can be found at 17 West Hill, London SW18, and if you get your order in quick they reckon to fill it by Christmas.



QL USER GROUP

Leon Heller, Chairman of the Independent QL Users' Group, presents a round-up of club news and information.

User Group News

IQLUG is a non-profit making group, with officers democratically elected by members at an Annual General Meeting. Accounts are independently audited and made available to members. The group is affiliated to the Association of Computer Clubs.

Membership is by subscription to Quanta, the group's monthly newsletter, currently containing 28 pages of members' letters, hints and tips, news on the QL scene, program listings and so on.

The group maintains a (mostly) free software library, which currently contains 15 programs. All library software is written by our members, and only non-commercial items will be held there. In addition a free advice service is provided: members can phone in with their problems, and be put in touch with someone who can help them – a register of members with expertise in various areas is kept.

Membership details are available from:

Brian Pain,
24 Oxford Street,
Stony Stratford,
Milton Keynes MK11 1JU.

Following the non-renewal of some members' subscriptions, we were not totally unsurprised that the reason turned out to be disenchantment with Sinclair and the QL. Those members in question had either cancelled their order, or returned the machine for a full refund. I have sent the responses to Sinclair's Customer Relations Manager, in the hope that they will mend their ways in future.

DIY Pascal Compiler

One of our members is adapting the Tiny Pascal compiler, featured in Byte magazine some years ago, to run on the QL. The original compiler was written in North Star BASIC, which had to be translated into SuperBasic, for the QL. However, a Pascal compiler written in SuperBasic is not really suitable, so the compiler was rewritten in Pascal and then compiled into object code, using the SuperBasic version of the compiler. The Pascal version of the compiler is then used to compile itself, and the resultant object code should be identical to that produced by the original SuperBasic compiler.

Terminal Emulator

All it takes to turn your QL into a very dumb terminal, for connection to a modem or another computer, is a simple seven line program:

```
100 CLS: BAUD 300: OPEN
    #4,ser2r
110 REPeat term
120   char$=INKEY$(4)
130   IF char$<>"" THEN
      PRINT char$;
140   char$=INKEY$
150   IF char$<>"" THEN
      PRINT #4,char$;
160 END REPeat term
```

Due to the fact that the QL serial ports require the same baud rate for data input and output, this program will not work with the 1200/75 baud modems used for the standard Prestel service, but it worked well enough for me to leave a message on a couple of computerised bulletin boards.

Psion Response Times

A few members have complained about the speed of Psion's response to their queries, so I wrote on their behalf to David Potter of Psion. I received the following analysis of their performance in responding to queries sent in by QLUB members:

65% of enquiries answered and posted on the same day as received.

15% of enquiries answered and posted in the following 24 hours.

8% of enquiries answered in the following 24 hours.

The remaining 12% are usually printer-related queries which require a lot of additional research. As their database gets bigger, they expect this figure to drop.

Archive Limitation

A couple of members using Archive have complained about a problem that arises with relatively large files (about 500 records). When they try to add more records, they get an Error 80 'out of memory' message. This is due, apparently, to there being insufficient space for the index needed to access records in the file. The problem will be overcome when the upgrades to the Psion software become available.

Putting The Brake On

I have found that the safest thing to do if a file has become corrupted, and the Microdrive won't stop, is just to pull out the cartridge. QDOS then returns an eror message, and you are back in control. You will have probably lost the file it was trying to read, but you will not have corrupted any other files, which might happen if you just press the reset button or switch off the power.

QDOS Version Numbers

I suppose everybody now knows about PRINT VER$, to find out which version of QL you have. What might not be so generally known is that the version returned (JM for the machines currently being shipped) is the version of SuperBasic. QDOS has its own version number, which is not accessible directly via SuperBasic, and can only be found by going into assembly language.

An Apology

In the last issue, I stated that Computer One's assembler for the QL was written partly in SuperBasic. In fact, it is written entirely in assembly language.

THE COMPUTER PROTECTORS.

Keep everything you want in.

Static and dust, the biggest enemies of the home computer. Fortunately there are now Targus computer bags.

Thanks to the protectastat, specially treated material, these unique bags give your computer total anti-static and dust protection. Only Targus can do this.

COMPLETE PROTECTION

And that's not all. Beneath the hard wearing nylon outer, is a high-density, impact-proof, foam padding and a thermal lining to stop the build up of condensation.

It's also fire-retardent and once sealed with the heavy-duty nylon zips, rain-proof as well.

A PLACE FOR EVERYTHING

There's also a handy compartment for your programmes, joy-sticks, cables, leads, accessories and manuals; so it's ideal for neat home storage, as well as transportation between home and work or school.

FITS ANY COMPUTER

These sturdy, attractive bags are black with a red trim and come in 3 sizes, fitting any model. Just check the list to see which bag you need.

And everything you don't, out.

FREE MICROWIPES

Order any bag and we'll also send you a dispenser of microwipe anti-static treated screen cleaners – absolutely free.

ONLY AVAILABLE DIRECT

Targus computer bags aren't on sale in the shops, so order direct from us now.

Simply tick box A, B, or C, clip the coupon and send it with the correct remittance, plus £1.75 to cover postage, to: Personal Purchase Ltd., Engineers Way, Wembley, Middlesex HA9 0EB.
Allow 21-28 days for delivery.

Check this list for the bag to fit your model of computer:

A. SINCLAIR QL
A. AMSTRAD CPC 464
B. BBC MICRO B
B. DRAGON 64
B. ATARI 800 48K
B. ATARI 800 XL 64K
B. ENTERPRISE 64
B. MITSUBISHI – MSX ML – 8000
B. SANYO – MSX WAVY 10
B. SONY – MSX HB – 75
B. TOSHIBA – MSX – 64K
B. SHARP – MZ – 700
C. COMMODORE 64
C. COMMODORE – C – 16
C. VIC 20
C. ORIC ATMOS
C. ACORN ELECTRON
C. SORD

RECORDERS

C. TR – 17
C. TR – 30
C. SAISNO DR 70
C. COMMODORE 1530

PRINTERS

B. COMMODORE MPS 801
C. ORIC

DISCS

C. ORIC
C. ZX SPECTRUM
C. ZX INTERFACE
C. COMMODORE 1541

Personal Purchase Limited

Engineers Way, Wembley, Middlesex HA9 0EB.

Please send the Targus bag to fit my home computer (tick either box A, B, or C) at £_____ plus £1.75 p&p. I enclose my cheque/PO for £_____ payable to Personal Purchase Ltd., or please debit my Access/Diners Club/Visa/American Express.

Card Number_____ Expiry Date_____

Signature_____ A. £19.95 ☐

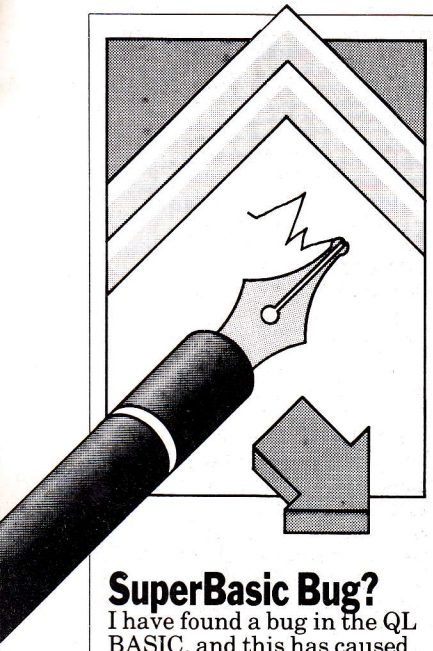
Name_____
BLOCK LETTERS PLEASE

Address_____ B. £24.95 ☐

_____

_____ Postcode_____ C. £29.95 ☐

Your remittance should be made payable to Personal Purchase Ltd., at the address specified.
Delivery on UK mainland only. Allow 21-28 days for delivery. Registered Office, Personal Purchase Ltd., Engineers Way, Wembley, Middlesex HA9 0EB. Reg. No. 1573900.
If not fully satisfied return your Targus bag within 28 days of delivery for a full refund. QL/12/84



OPEN CHANNEL

This is the spot where we turn the magazine over to you, our readers. We welcome any comments, criticisms or anecdotes about either the QL or QL User.

The address to send your letters is: ***Open Channel, QL User, Scriptor Court, 155 Farringdon Road, EC1R 3AD.***

SuperBasic Bug?

I have found a bug in the QL BASIC, and this has caused me much head scratching, as it was giving errors in a program which I had successfully used on my Spectrum. To illustrate it try the following:

```
10 DIM A$(10) : DIM
   B$(9)
20 LET
   A$="QWERTYUIOPA"
30 LET
   B$="QWERTYUIOPA"
40 IF A$=B$: PRINT
   "9=";B$(0)
```

This will give you 9=10. It seems to be that dimensioning a string of less than 10 characters gives one more character than it should.

Note: A$(0) and B$(0) do exist and contain the length of the string.

G L Riddle
Kettering

Well spotted. You'll find this confirmed in 'QL SuperBasic' by A A Berk (Granada Publishing) page 95. However, with obvious differences between Spectrum BASIC and SuperBasic it would seem generally inadvisable to expect that a program working on one machine would work on the other without extensive modification.

Uncertain Futures

Sid Smith's news item in the Oct/Nov issue implies faults in QL hardware prior to the D07 level release. I have a D06 machine and would appreciate information on faults and weaknesses.

G R Chatburn
Essex

In your latest issue of QL you mentioned that if the first letters after the D on the serial no are 07 then the version is supposed to be good. But what about mine which is '06'.

I am a student, I cannot afford to maintain my computer with further charges for updates.

M M Patel
Coventry

Provided that you have either versions AH or JM of the operating system then you have no worries as far as the firmware is concerned. At least this would appear to be the view of Sinclair Research who maintain that the differences between the two are so slight as not to merit an upgrade.

As regards hardware, all we have to go on are rumours that later releases incorporate improvements to the microdrives and keyboard.

Program Stopwatch

With reference to Mr Timothy Fuller's letter in your last issue concerning amongst other things timing.

I use the following procedure that calculates the lapsed time of a program running.

```
DEF PROC USED__
TIME (TYPE)
SELECT ON TYPE
ON TYPE=0: LET
FT=DATE
ON TYPE=1: LET
UT=DATE-FT: LET
UT$=DATE$(UT)
END SELECT: END
DEF USED__TIME
```

then at the start of my program I put

```
USED__TIME(0)
```

and at the end

```
USED__TIME(1)
```

S Dudley
Sussex

Timely Request

The QL is my first computer and I have no knowledge of computers; rather like buying a Rolls Royce then learning to drive.

I am using a Panasonic portable colour TV and whilst SuperBasic gives a satisfactory picture, the Quill, Abacus etc, are wholly unreadable. This seems to nullify any future advantages that I might expect. Please advise.

Peter Bunley
Colchester

If you read the introduction to our monitors review you will appreciate some of the problems associated with TVs. It is worth adding that displays on Japanese TVs are generally worse than those produced on the home grown variety.

A Word to the Wise

Thank you for the very favourable review of Psion's EASEL program which appeared in your October issue. It is a generally accurate and well written piece, but the criticisms levelled in the paragraphs headed 'Fraught Filing' and 'Invisible Exports' are unfounded. The Save and Export commands in Easel in fact deal with all those sets of figures (eg, Costs, Sales, Prices) which are currently on display. This allows some choice in precisely what you save to a file.

An error in the manual would appear to be to blame for misleading your reviewer into thinking that all figures in memory would be saved, and evidently, by some accident, he never tried to Save or Export with more than one set of figures on display.

Dr Martin G Brown
EASEL's author

We have been in touch with our reviewer and he confirms this to be the case. He also adds that this puts EASEL on an equal footing with both ARCHIVE and ABACUS in terms of quality and presentation. Shame about the manual though!

Anyone for a Game

I feel that the QL is an excellent machine despite its premature launch. I would like to see some games for it and I have read that Psion are currently working on some. Any information?

J Wallace
Glasgow

Aside from Talent and Psion, both of which are dealt with in this issue, no other software houses would appear to have given the QL a look-in as far as games are concerned.

There would appear to be three reasons for this. One; games software houses with a few exceptions simply do not have the expertise necessary to write 68000 machine code games.

Two; there would appear to be a somewhat misguided notion that the QL is exclusively a business machine and that any games written for it will not sell.

And three (and by far the most important reason); the cost and limited availability of microdrives plus the fact that they cannot be duplicated in the same way as cassettes, acts as a considerable deterrent to the enterprising independent.

Beginner's Luck

I am a salaried architect . . . I received my QL in mid-August. It is my first computer and I have been exploring ways to use it to advantage in my work. I find the QL to be very powerful but very perplexing at times. Sometimes it seems to have a mind of its own and will not accept instructions which it ran quite happily the night before. Whether this is the machine or the operator has yet to be assessed.

John McKevitt
Newry

With its numerous teething

problems and the dubious reliability of its microdrives nothing is certain on the QL. However, drives aside, if there are errors in QDOS, SuperBasic or Psion's packages these will occur the night before as well as the morning after. So your problems are more likely to arise from inexperience. Which is hardly surprising bearing in mind the QL's idiosyncratic and exacting command syntax.

Edit Aid

My QL seems to be different! At least I haven't seen the following procedure explained anywhere.

Where there is a program in memory and if AUTO has to be used or is entered, then the DOWN CURSOR will step through the program and at the end continues to produce blank spaces.

Similarly the UP CURSOR will step backwards (missing out blank lines created) but if you go past the start you cannot get back without entering AUTO. Edit also works but doesn't produce blank lines at the end so you cannot step back again so easily. Also lines can be edited without loosing the AUTO facility.

I find this useful for going back a few lines but it is a bit slow when editing in the middle of a program.

Brian C Bird
Aberdeen

Lucky Blighter

Maybe I am one of the lucky ones who have received a production version of the QL with all of the errors corrected. As yet I am not sure as I have not yet thoroughly put the machine through its paces but certainly some of your criticisms seem wide of the mark.

How much more useful your magazine would be if it took a positive approach to getting the best out of the QL instead of always comparing it to a VAX or 11/73 machine and pointing out things that the QL cannot do.

Ronald C Wally
Kent

In many ways your criticism has been anticipated by the comparison, in this issue, of the QL with an equivalent 8-bit machine. It is worth pointing out, however, that documenting the machine's faults is hardly negative as it often gets things put right.

Psion ROM

I am a newcomer to computing, and I find the manual supplied with the QL difficult to follow. Can you advise me of any publications which might prove easier. Also I have read somewhere, that the four programs provided are going to be updated and loaded via the ROM socket. Will people, such as myself, who have already purchased the machine receive copies of these.

D H Scott
West Yorks

There are a number of books generally available which deal with most aspects of the QL and Psion's programs. However if you want a good all rounder we suggest 'Desk top computing with the Sinclair QL' by Barry Miles and published by Hutchinsons Computer Press.

As regards your query concerning ROMs and upgrades, Sinclair Research have informed us that Psion's programs will continue to appear in Microdrive form.

Manual Endeavour

A few days after I sent off my Registration Form I received my QL and have had some time to try it out.

I find that when I follow a 'Stored Program' on p.10 of the instruction manual the answer appears on a different background colour, obliterating the first line of the program.

Also, on the 'pens' program on p.12 it is only after the RUN command has been given that the question 'How many pens?' appears – and again this is over the top line of the program, the inserted number appears in the position of the 1 in 110 and the total cost covers the 12 in 120.

There is no note in the manual to say this is how it should happen.

I'm also having difficulty in finding the way to store information in the Microdrive!!

Incidentally I'm retired, have bought the QL mainly for the benefit of the next two generations of my family.

Arthur Nunn
Ripon

In A Nutshell

Many thanks for the communiqué, and a magazine which promises to become the forum for all QL users in Europe (and if Big C's plans bear fruit Stateside too?)

The problem with such a radically different machine is that no one has the technology to support it. Z80 whiz kids are having to go back to the drawing board and methinks it will be some time before they start turning out machine-stretching software for the baby 68008.

Hardware support is essential too, and the minute the profits are enough to give Sinclair Research the launch it deserves in the world of public finance, the price of a microdrive cartridge must come down to reasonable levels (byte for byte it will have to become cheaper than disk.) A scurrilous rumour I picked up in Cambridge puts the cost of manufacture of the cartridge itself (presumably neglecting the over-the-top packaging which fools nobody, Nigel) at under twenty pence!

The explosive days of the computer Big Bang were fuelled by one simple factor: accessibility. Anyone who wanted to write software and maybe start off a small business only needed a Spectrum, a twelve quid tape recorder and a bit of imagination. The inherent inventiveness of the Brits might prove to be as bankable as it was in Victorian times once more? So let's see a policy of support for the QL that allows the maximum number of people to refine their techniques. Keep it simple and in sequence, let's have some series of in-depth teach-ins. QL User could do worse than to track down electronics firms known to be using MC68000's (like Quantel, the TV effects people) and induce their QL-owning staff to part with some of their secrets. Maybe the mag can carry a Societies section, be instrumental in helping readers set up their local user-groups, and aid their efforts to communicate between groups.

I must confess, I am a Sinclair believer, I have no wish to see Foreign Powers do to our computer industry what they did to our motor bike, auto, engineering, shipbuilding and consumer electronics markets. Death to MSX! Gentlemen of the press, careful what you say, the rest of the world are watching. To see the frustrations of the magazine sector working themselves out on paper in the age of the 'kludge', you would think no computer launch had ever before been premature. If the Jolly Blue Giant can get late on computer delivery, there's an excuse for everyone . . .

Which brings me to add-on systems: a process akin to Darwinism leaves us with 'preferred producers' after a short while, (computer standardisation is so enormously in the interests of the consumer that market forces inevitably favour the leader, as Spectrum competitors have already found to their cost). A modest editorial effort from QL User to hasten this process by including comparison reviews of all add-ons that hit the market is vital. As a consumer I want to be able to identify the preferred producers of printer interfaces, floppy disk systems, modems, software utilities and user support magazines as soon as possible. In the case of the latter there is already no doubt in my mind. (Well done QL User!)

Rob Carter
London

Errors!

As a beginner at computing I had anticipated that I would get help from the 'User Guide'; unfortunately it bears all the hallmarks of having been rushed out, even though my machine was issued in August. It appears to be full of printer's errors as well as clearly incompletely edited sections. For the expert probably only a momentary annoyance, but for a beginner it is almost totally inhibiting, since he can never be sure if failure to solve a problem is due to his own stupidity, or to an editorial oversight. I have in consequence gone through the guide up to Chapter 7 of the 'Beginner's Guide' twice. Some of my early problems were solved on repetition due to obtaining an answer further on – for example, the fact that one cannot use the word 'width' when invited to calculate the dimensions of a carpet is nowhere explained (it is reserved as a 'Keyword').

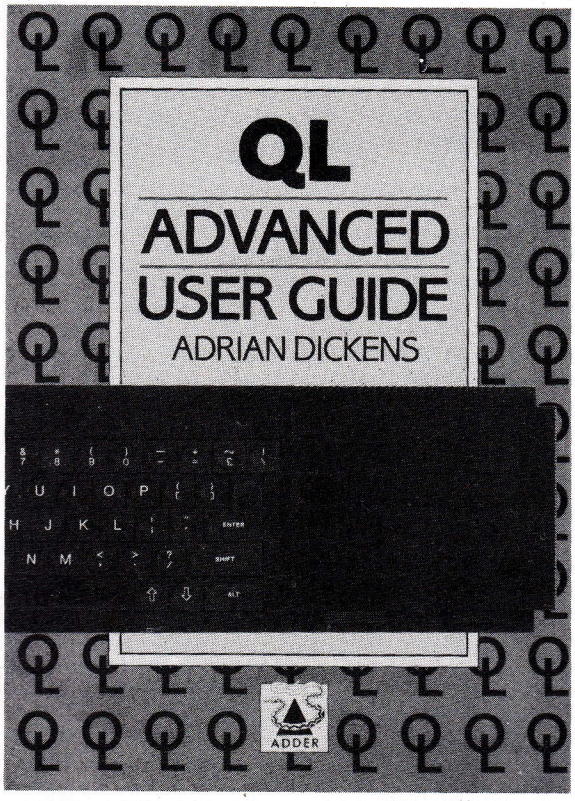
I have struggled away despite this, but to-day, after my fourth attempt to solve the 'Buzzword' program (with many different solutions for individual lines) my patience has gone.

W G Taylor
Ayr

ASSEMBLER

The ADDER 68K Assembler is a professional, fully integrated editor/MACRO assembler development package for the QL. Providing a quick turn-around on the *editing, assembling, debugging* cycle, it assembles Motorola format source files to produce multi-tasking programs, additional Super BASIC facilities plus much more. Features include:

- ★ FULL 68000 MACRO Assembler
- ★ Integrated full screen editor
- ★ Conditional assembly
- ★ External library file inclusion
- ★ Complete error checking diagnostics
- ★ Comprehensive manual
- ★ MACRO ASSEMBLER only £29.95*



THE BOOK

The QL Advanced User Guide (£12.95*) has been written by Adrian Dickens in collaboration with Tony Tebby (QDOS System designer). It is the complete guide to QDOS and the Sinclair QL, covering multi-tasking, transient programs, resident procedures, heaps and stacks, traps and utilities, 68008 assembler programming plus much more. All of these features are illustrated by practical examples, and the powerful QDOS experimentor program allows many facilities to be tried out from BASIC. All of the programs from the book are available on a microdrive cartridge which can be purchased with the book (£9.95*).

ORDER FORM

☐ Assembler development package **£34.95** incl. VAT and p&p
☐ QL Advanced User Guide Book **£14.45** incl. p&p (no VAT)
☐ QL A.U.G. programs from book **£11.95** incl. VAT and p&p
☐ **SPECIAL CHRISTMAS OFFER**
All three items from above (normally £61.35) **£55.95** ALL inclusive

I enclose cheque/PO for £.....
Name .....
Address.....
.....
City..... Postcode .....
Send to: **ADDER PUBLISHING LIMITED, P.O. Box 148, Cambridge CB1 2EQ**
Dealer enquiries welcome

QL and QDOS are trademarks of Sinclair Research Limited
** Price excluding VAT and p&p*

ADDER

QL

PARALLEL PRINTER INTERFACE

£39.95 INC

- ★ 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, Canon, etc, etc
- ★ Fully QDOS and SuperBasic compatible

To order send name and address with cheque to:
MIRACLE SYSTEMS Ltd
Avondale Workshops
Woodland Way
Kingswood
Bristol BS15 1QL

Sinclair and QL are trademarks of Sinclair Research

0272 603871 Ext 210

Access
Available at larger branches of **WHSMITH**
Access orders welcomed

TIM HARTNELL'S QL GAMES COMPENDIUM

From Tim Hartnell, the author PCW magazine called 'Mr Sinclair' comes this major games book for the QL. . . .

Not only does it contain listings for more than 20 complete games, but the detailed instructions and ideas will help you develop your own games, while improving your programming as you do so. There is even a complete chapter to teach you how to create moving graphics games on your QL!

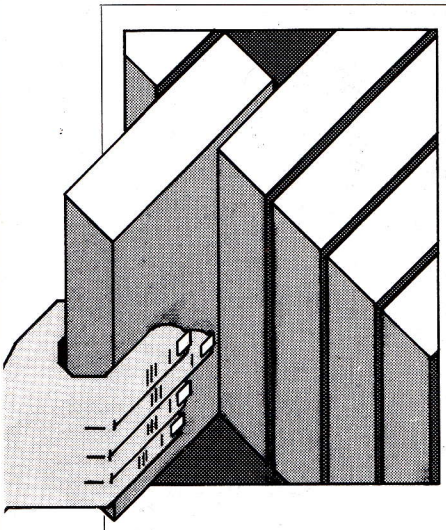
The programs include:

ELIZA — in which your QL acts as a psychiatrist and talks to you
QTHELLO — challenge your QL to a round of Reversi/Othello
MAGICIAN'S MAZE — high adventure exploring ruins after gold
DETECTIVE Q — solve the murder; different each time it runs
STOCKBROKER — a challenging simulation puts thousands in your control as you work towards getting a million
QL RACER — real-time moving graphics, in a game few will win
TRYST WITH MEPHISTOPHELES — nearly 20K of challenging adventure

another great book from Interface

INTERFACE PUBLICATIONS, DEPT. QL, 9/11 Kensington High Street, London, W8 5NP

Please send me the following. I enclose £.....
( )Tim Hartnell's QL Games Compendium — £5.95
Name .....
Address.....
.....



BOOKMARKS

Despite its brief history, the QL has acquired considerable literary support – Nicky Trevett selects from amongst the latest.

Books, books and more books continue to appear, jostling for the QL user's attention, as book publishers cluster round what they perceive to be the QL honeypot. Buzzing round at the moment may be found the likes of Hutchinson, Century, Granada and Sunshine.

Series Bookings

Hutchinson extends its Sinclair QL Series with two volumes on SuperBasic – the QL's unique (and often superior) version of BASIC. Whilst one book is designed to be an introduction; the other a more advanced study, they do not necessarily form a two-part whole.

Introduction to SuperBasic on the Sinclair QL, is a

simple view of the subject aimed at anyone interested in writing, or learning to write, SuperBasic programs on the QL. It treats the topic in a logical series of steps, starting with 'plugging in' and 'switching on' (for the true beginner) and fundamental commands, like PRINT. Readers then progress to bolder things – actually writing simple programs – by chapter three. The book follows this with the usual aspects of programming – inputting data to the program, selection and repetition, standard mathematical functions, strings, variables, arrays, procedures and functions, rounded off with a quick look at subroutines.

Sample programs are drawn from every subject, including games, random numbers and straightforward conversion of Celsius to Fahrenheit (demonstrating the DEFine FuNction). There are exercises at the end of every chapter to enable testing of your new-found skills, with the answers (mostly in the form of program listings) at the back. There is also a long section on SuperBasic keywords, plus a glossary – particularly relevent here, but probably familiar to readers of Hutchinson's *Desk-top Computing with the Sinclair QL* and *Introducing the QL* books.

Advanced Programming with the Sinclair QL adopts quite a different approach to the above; though complementing much of the other book's material.

Author Martin Gandoff presents a personal view of SuperBasic programming based on his own experience and specifically aimed at commercial applications. Thus the sample programs

here deal with payroll, personnel files, invoicing and so on, which is likely to dismay all but those wishing to write programs for their business.

It would be a pity if the title were to put off inexperienced programmers, because while a certain level of programming knowledge is assumed, the down-to-earth and practical hints provided throughout the book would be of real value to any commercial user. Two chapters, called 'SuperBasic 1' and 'SuperBasic 2' respectively, run through the workings of that language, more or less summarising the content of *Introduction to SuperBasic*. The rest of the book looks at the theory and practice of programming within a commercial context, and other thoroughly practical matters like error detection, program development and debugging. This is followed by a series of program listings demonstrating a variety of applications.

Both books embody a vast amount of information and are subsequently a real snip at £6.95 each.

Family Frolics

Century Communications has likewise dreamed up an idea for a series of QL books – if you buy one, you might as well buy them all. And as an incentive, the team of authors has thought up a cunning gimmick – all the books are based around the often hilarious activities of the eccentric Blake family and their friends.

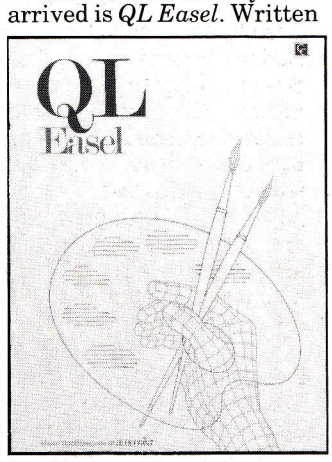
Take *QL Quill*, by Francesca Simon and Clare Spottiswoode. No-one could accuse this of being just another introduction to the Quill word processing package. Presented in a clear, easy to read format, the reader is expected to work through the book, doing all the 'exercises', until

every aspect of the program has been learned and practised.

It *would* be tedious, except that the exercises take the form of letters and documents composed by members of the Blake family. Readers are urged to correct, for instance, Nigel Wooten's pretentious letter to Harold Blake, asking for permission to marry Harold's daughter; experiment with typefaces and underlining while writing a business letter to Joyce Blake's trading partner in Bangkok; manipulate margins in an attempt to reproduce the tricky directions to Nigel and Miranda's wedding; and create headers and footers as you compose Harold's marketing strategy for his company's new toy, 'Lawrence Lizard' (phew!)

Frivolous maybe, but a relatively painless way to learn. Before you know it you've covered all the editing commands, justification, stationery and printing, tabs, search and replace, standard paragraphs and forms, and even looked (briefly) at the integration of Quill with its three companion packages – Abacus, Archive and Easel. This brevity no doubt suggests similar titles are on the way to cover each of these individually.

One that has already arrived is *QL Easel*. Written



by Alison Spottiswoode, this continues the theme of *QL Quill*, although to less effect. The book costs the same as its word processing cousin (£7.95) and even manages an extra 30 pages or so, but given that Psion's Easel manual is reasonably adequate it doesn't give the same impression of value for money.

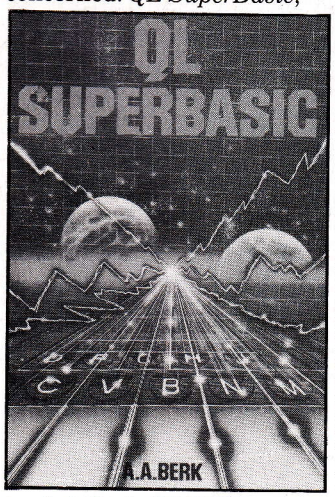
BOOKMARKS

Like *QL Quill*, the book is entertainingly written and professionally presented, but almost goes into too much detail. There's the obligatory chapter introducing the QL, two chapters on plotting numbers and sets of numbers, setting up graph skeletons, choosing the best graph type, designing graphs, special sections on bar charts, line graphs and pie charts, saving and copying graphs . . . get the picture? There's even a quick look at using Easel with Abacus, Archive and Quill.

You'll certainly find everything you ever wanted to know about Easel, and probably more; but does one really need it?

The Sober Approach

Budding QL programmers are likely to be spoilt for choice as far as SuperBasic programming manuals are concerned. *QL SuperBasic*,

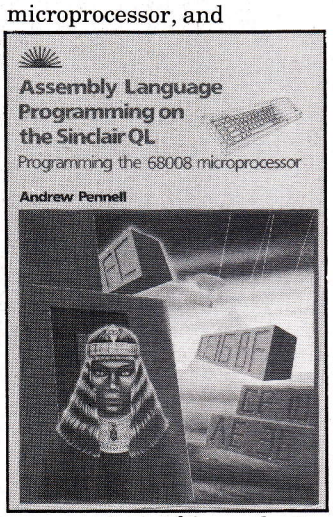


by A A Berk (£6.95 from Granada) approaches the subject in much the same way as Hutchinson's *Introduction to SuperBasic* but rather more soberly. The blurb on the back cover claims that "Many programs are provided as examples, setting out the principles of structured programming and the applications which are possible with the QL", but I can't say they leapt off the page. Most of the sample listings are short, abstract, and concentrate on illustrating a specific programming technique rather than demonstrating any practical application. No context is given for day-to-day progams, which is a particular drawback for the beginner.

The book is, however, competent and comprehensive, covering input and output, loops and decisions, strings and arrays, calculation, graphics and sound, Microdrives, files and devices, procedures, subroutines and functions.

Here Comes The Sun

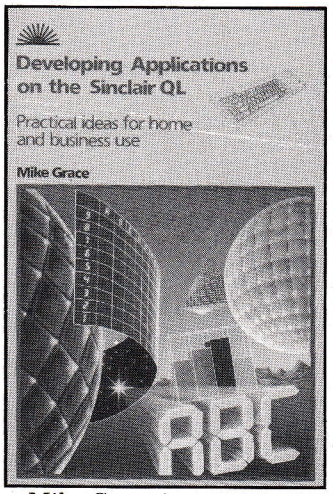
Intrepid programmers might like to tackle Sunshine Books' *Assembly Language Programming on the Sinclair QL* by Andrew Pennell, price £7.95. This deals with programming the QL's Motorola 68008 microprocessor, and



manages to combine a chatty style with technical content and the sort of jargon that's probably unavoidable – 'trace mode', 'interrupt priorities' and the like. It will be most accessible to the QL user already familiar with the principles of machine code programming; though newcomers are catered for in a chapter that covers some of the fundamentals, like ROM and RAM, hex, binary and addressing modes. Other chapters look at ways of using the QL's hardware, including a sideways glance at the Intel 8049 slave processor and the coding of 68008 instructions – a huge A-Z of 68008 instructions taking up a third of the book.

Usefully, a complete disassembler is provided to convert sections of memory from numbers back into recognisable instructions.

From programming we move on to applications – *Developing Applications on the Sinclair QL*, by Mike Grace, professes to offer practical ideas for home and business use, but it's really not much more than another book on Quill, Archive, Abacus and Easel.



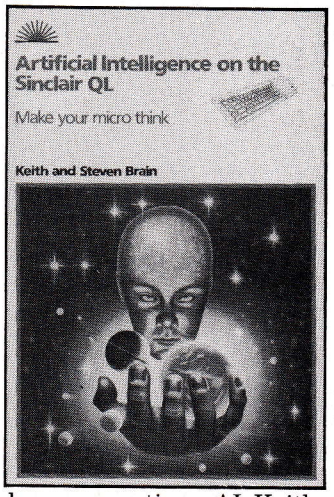
Mike Grace is a genuine Sinclair enthusiast ('After a very short time I had fallen in love with both the QL and the software . . .') and his enthusiasm lends this book a certain liveliness; although the Sinclair/Psion hero-worship gets a bit wearing.

Target readership seems to be just about anyone who possesses a QL. Each of the four Psion packages is dealt with in turn, approaching them with the cheerful assumption that readers are quite ignorant of the program's great potential. For example, the chapters on Quill discuss the variety of uses to which the word processor can be put, examines the way Quill works and offers advice on how to get more from the program.

The book is short on applications ideas and long on how to use the software. It would be of value, perhaps, for the QL novice confronted with four mysterious packages and willing to pay £6.95 for some friendly advice on what to do with them.

Official Art?

And so to one of the most fashionable applications in



home computing – AI. Keith and Steven Brain's *Artificial Intelligence on the Sinclair QL* arouses expectations which are not entirely fulfilled. 'Make your micro think' blusters the front cover. Inside, the authors embark on a merry jaunt through the world of sci-fi fantasy which might be fun, but has little to do with AI.

This frivolous foray is followed by a short section summarising the history of AI and discussing the attributes of modern robots – neither prepares you for the anti-climax to come.

What is possible, however, is implementing a few AI techniques within the SuperBasic programming framework. There's an interesting chapter on the knotty problem of natural language, wherein your QL tries to make sense of sentences and come up with plausible replies, and another on building your own expert system. There's a quick look at fuzzy matching, and an entertaining digression into ways of teaching a computer to find its way round a maze.

Very little of all this is likely to prove of real, practical use in the everyday running of a business or home, but for £6.95 there is a great deal of entertainment here for those with enquiring minds and time to spare. Anyone genuinely interested in the complexities of artificial intelligence will find the subject treated at far greater depth elsewhere, but the newcomer will find an adequate introduction to an important and potentially far-reaching subject.

Choosing a printer is a lot easier than choosing a computer.

THERE are dozens of quality printers from which to choose. With quality price tags of around £250.

The Brother M-1009, however, breaks all the rules.

Stays defiantly below the £200 barrier.

Though it has far more than its fair share of features, it maintains the extraordinarily low price of £199.95.

Travels at a steady fifty.

In the speed stakes, the M-1009 is certainly no slouch, being fully capable of up to 50 characters per second.

Providing bi-directional and logic seeking printing for normal characters and uni-directional printing for super and sub script and graphics.

Prints on any paper.

Being an impact printer, the M-1009 will print on virtually any paper, including letter headings, invoices and standard office stationery.

It will even print two copies together with your original.

A superb character recommendation.

In its price range, the M-1009 has a great deal more character than many printers.

96 no less, plus international type and graphic characters.

Reliability comes as standard.

Built to the same exacting standards as Brother's elite office printers, the Brother M-1009 already has faultless credentials for reliability.

Its 9 pin dot matrix head, for example, has an astonishing 20 million character service life.

One printer that doesn't block out the light.

Many home computers tend to be a little on the large side.

In contrast, the compact M-1009, at only 7 cm high, keeps a discreet profile.

Well designed, reliable – and conscientious.

The Brother M-1009.

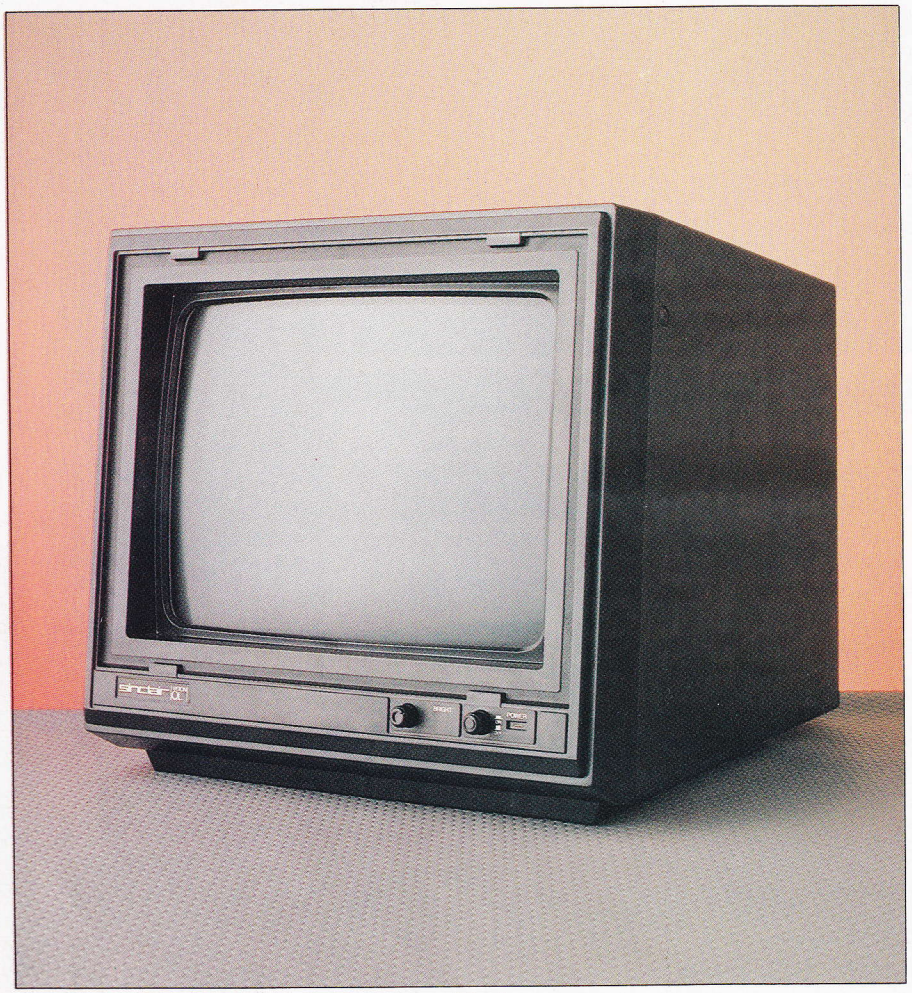
brother®

The future at your fingertips.

DEPT P, BROTHER OFFICE EQUIPMENT DIVISION, JONES + BROTHER, SHEPLEY STREET, GUIDE BRIDGE, AUDENSHAW, MANCHESTER M34 5JD.
TEL: 061-330 6531 (10 LINES) 061-330 0111 (6 LINES) 061-330 3036 (4 LINES) TELEX: 669092
BROTHER INDUSTRIES LIMITED, NAGOYA, JAPAN.

AVAILABLE FROM
BOOTS, WILDINGS, SPECTRUM, JOHN MENZIES, MICRO MANAGEMENT, MAJOR DEPARTMENT STORES AND ALL GOOD COMPUTER SHOPS.

QL



Centel's neat little monitor in customary black.

The official 'Vision QL' monitor from Data Efficiency.

Few would dispute that the QL has lowered the threshold of professional computing. With its 32-bit processing power, microdrives and 128K RAM it occupies a unique niche in the market. However, its £400 price tag is deceptive. Whilst this buys a powerful machine, it does not include the instruments that will enable you to make the most of it.

This is especially true of the QL's video display, which handles 85 column text. However, standard TV can barely support the Quill's default 64 column format, let alone the 80 column one. If you doubt this try typing in a five page manuscript in a single session. By the fifth page you will almost certainly be bleary-eyed and suffering from a mild headache. So, buying a monitor for your machine is not a luxury but a necessity.

The QL produces two kinds of video output: composite video and RGB. With the former a single signal is transmitted from the computer to the monitor and then split up into three separate colour signals controlling the red, green and blue colour guns. With the latter the three colours are *transmitted* separately.

Unfortunately, not only is the QL's 85 character data window unique, but also the manner in which signals are transmitted would appear not to conform with certain industry standards. The choice of monitor is therefore limited to those specially designed for the machine.

Feeling The Width

First, most manufacturers specify a minimum bandwidth for signals the monitor is capable of receiving. This figure is a useful measure of the machine's display capability. As approximately 1MHz is required to accurately display 60 pixels on a line and the QL is capable in high res mode of produc-

Screen Duo

Paolo Baccanello assesses two of the latest QL-specific monitors – Data Efficiency's official 'Vision QL' and Centel's new model.

ing a 512 pixel line, it does not take a brilliant mathematician to work out that a minimum of 9MHz would be required to produce a reasonable picture (TVs transmit at 4.5MHz).

Secondly, unwarranted emphasis is often placed on the dot resolution that a monitor is capable of. The important thing is it does not fall short of the QL's maximum, namely 512 dots horizontally and 256 dots vertically. Any surplus can safely be ignored until such time as Sinclair feels it necessary to swop the existing video chip for something more powerful.

Finally, it is easy to get carried away by the first monitor you set eyes upon simply because it is such an improvement upon a TV. However there is no excuse for not testing it. The following checklist identifies some potential shortcomings. All could easily be detected by running a straightforward SuperBasic program displaying graphics and characters with different foreground and background colours in low and high resolution modes.

DIY Checklist

1. Are the characters easily legible at normal reading distance?
2. Do colours adjacent in the spectrum stand out against each other? For example, can yellow characters printed on green background be made out?
3. Are characters distorted at the edges of the screen?
4. Is the screen rock steady or does it shimmer or fade?

Vision QL

Made in Japan, this unit seems to have benefited from that country's flair for making things smaller without any loss in quality. The display is as crisp and sharp as that found on the more typical 14" medium resolution monitors; though at maximum intensity the smaller characters do tend to blur in a dazzle of light. Phosphors are of short persistence leaving no decaying image when switched off.

The unit is anti-glare, meaning that it is less tiring on the eyes. Furthermore, with the screen at a slight tilt and raised above the level of the QL's keyboard, displays are easy to view. Finally, the facia has four built-in clips which will presumably afford the user the option of fitting a separate protective cover over the screen.

The on/off switch and brightness control are located at the front of the unit making them readily accessible. Mains and RGB leads are permanent fixtures at the back. Both come complete with the appropriate connector so that installation is simply a matter of 'plugging in'.

Housed in a semi-textured black plastic casing the monitor blends in well with the QL. Bearing in mind that the QL itself is a compact machine, the two make a handsome couple and if the legendary reliability of Japanese TV's is anything to go by then Sinclair Research's patronage has been well deserved. However, this does little to resolve the mystery as to why the smaller screen commands the higher price? Nevertheless if space is at a premium at your workstation the Vision QL would certainly be in contention to fill it.

Centel Monitor

Centel would appear to have gone to some lengths to produce a sturdy product destined for use in the office. The monitor's anti-glare internally etched CRT is designed to minimise reflection from the strip/fluorescent lighting. The high quality foam moulded cabinet is of the kind adopted by the more prestigious computer companies, namely IBM and ICL. Black and semi-textured, it provides a suitable match for the QL. Also, the absence of any sharp edges should please the safety conscious.

The medium resolution monitor is easily capable of supporting any display that the QL can generate. Dots are bigger in size than on 12", giving a 15% greater picture size. This means that characters are larger and legibility is improved. Whilst not as brilliant as smaller or unprotected screens, clarity was sharp and characters easily discerned on a variety of coloured backgrounds.

Resting firmly on four plastic pads, screen raised and tilted at the optimum angle for easy viewing there is little to fault in the machine's design. One small criticism: with the power switch and brightness controls located at the rear of the machine, turning on or off and adjusting the display involves crooking your arm around the back.

Well finished and robust, Centel's monitor should appeal to the more professional user looking for reliability and practicality. No sacrifices have been made in the interests of cosmetics.

Vision QL 12" RGB Monitor

Supplier: MBS Data Efficiency, Maxted Road, Hemel Hempstead, Herts
Price: £299 inc VAT
Size: Height 250mm
Width 315mm
Depth 320mm
Bandwidth: 18MHz-20MHz
Dot width: 0.38mm
Dot resolution: 640(H)×440(V)

Centel 14" RGB Monitor

Price: £299 inc VAT
Supplier: Centel, Unit 5, High Technology Park, Shearbridge Road, Bradford
Size: Height 325mm
Width 370mm
Depth 345mm
Bandwidth: 18MHz-20MHz
Dot width: 0.42mm
Dot resolution: 650(H)×440(V)

Sir Clive Live!

Peter Rodwell talks to Clive Sinclair about the industry generally and the QL specifically.

'It's not true that I have lost interest in computers – it's the computer side which overwhelmingly takes up my time and excites me.'

Sir Clive Sinclair was sitting in his surprisingly modest – but tastefully decorated – office at the top of the rather pokey Sinclair Research building in London. He was referring to a comment we made in *QL User* a couple of issues ago when we alleged that he appeared to be more interested these days in his electric car and flat-screen TV than in computers.

We hadn't meant it as a criticism, but Sir Clive wanted to make it clear that, even though he might have stepped back from the day-to-day manufacturing and selling operation, he was still very much involved in research and development. That R & D work is taking place in Cambridge, where Sinclair has two laboratories, one of which is the famous Metalab, a pure research establishment where the only brief seems to be a 'blue sky' one. Sir Clive and his researchers are deeply into a fifth-generation computer project, it turns out.

A fifth-generation computer is an artificial intelligence machine, a super-expert system. If you saw the film *2001: A Space Odyssey* then you'll remember HAL, the machine with a mind of its own. When it (he?) was functioning properly, HAL was pretty much the sort of thing we mean when we talk fifth-generation; but I hasten to add that unless we're crazy we won't build computers which take matters into their own hands in the way HAL did.

Developing AI

Currently, the USA and Japan seem to be putting the most into 5G; Britain has the Alvey project, which comes nowhere near to matching the scale of the US and Japanese efforts and which Sir Clive is already on record as considering inadequate. So – true to form – he's decided to go his own way.

'We've got involved in this fifth-generation project much more deeply than people suppose, because we can see the Japanese pouring in enormous investments and the US has got two very big projects,' Sir Clive explained. 'In Britain there's us, so we won't find ourselves out of the club – there's no-one else in Britain with a fifth-generation project as far as we can see.'

Of course, Sir Clive isn't going to spell out precisely where his research work is heading or what the outcome will be. With much 5G work pushing hard against the 'techno-boundary' anyway, there may as yet be no clearly-defined goal. He makes it clear, though, that some sort of low-cost expert system machine is on the cards: 'The first artificial intelligence machines will be enormously expensive but eventually you will see domestic machines.'

I could have talked futures all day with Sir Clive – he has long been a source of interesting and sometimes controversial ideas. But what about the present?

Spin-Offs

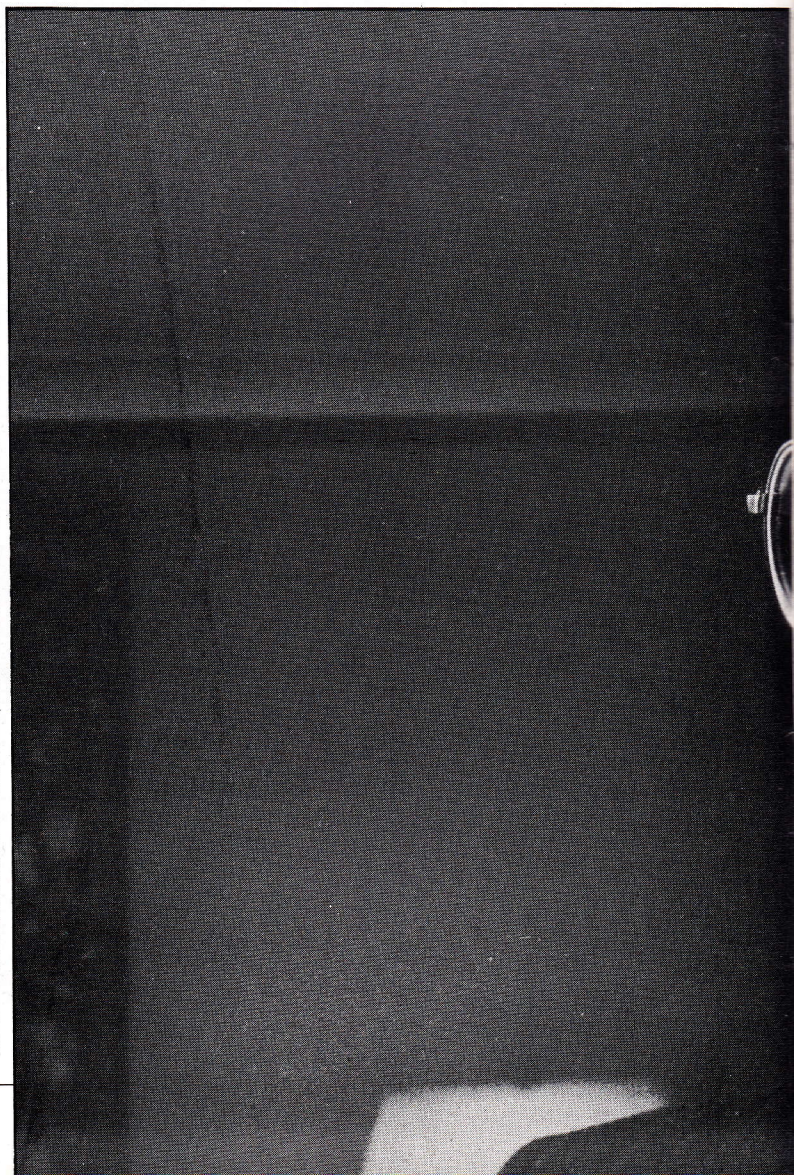
Well, one advantage of running a 5G research project is that you can spin off some of its findings into real products as you go along.

'To make a fifth-generation computer we're going to need enormous amounts of memory – we're going to need gigabytes,' explains Sir Clive. 'I don't know how many, but we can't do it with ordinary RAM technologies.'

The solution is called wafer integration and requires a little explanation. Currently, chips are manufactured in batches on discs of silicon about four inches in diameter called wafers. Each wafer contains dozens of identical chips and at the end of the production line they are cut up into individual chips. What Sinclair is proposing is to produce super-sized chips, where each wafer is just one enormous chip containing perhaps millions of memory cells in one big component.

This may seem a rather obvious step, but the problems are huge. Designing a chip of this complexity will be horrendous; many normal-sized processor chips are already too complex for any one person to understand and require extensive computing power to design.

Then there's an economic problem. A wafer is expensive to make and the process is very delicate: a speck of dust or an impurity in one of the chemicals can ruin all or part of a wafer. If the wafer is a conventional



one, containing many identical chips, you just throw away the faulty ones. The secret of successful chip making is to get the yield rate – the percentage of working chips on each wafer – as high as possible. Horror stories abound of plants running for months with zero yield rates as they try to put a new chip into production.

If a wafer is one huge component, then the yield rate is even more important: you can't afford to throw away very many of them and have cash left over for the staff Christmas party. In fact, you can't afford to let one speck of dust or whatever ruin an entire wafer so you have to build redundancy into the design: you duplicate parts of the circuitry and if faulty areas are found during testing, you reconfigure the 'chip' to use the extra parts instead of the damaged ones. But this does of course add to the chip's complexity . . .

Sir Clive seems confident that he has – or will – solve these problems. 'We are very confident that we do know how to do it and we expect to have a product some time next year,' he said.

And what will that product be? A solid-state Winchester 'disk', that's what. It will come in half-megabyte modules which you'll be able to stack together to build up a massive data storage.

What it comprises is actually a wafer-sized memory chip with battery back-up and what sounds like a complex system for detecting any errors and recovering from them. It will behave just like a hard disk with one important exception – it will work more quickly, maybe 100 times faster than a Winchester disk. If you've been cursing at your Microdrives' slowness, an ordinary Winchester would seem like magic; these solid-state 'disks' will seem positively psychic.

'The speed is so high that you will be able to store whole screens of information and then play them back quickly, to produce animation,' says Sir Clive, adding that at the moment the plan is to produce them only for the QL to start with. And the cost? 'About £200 per module.'

Software Solution

One interesting feature of the Sinclair offices is the almost total lack of computers on view. Sir Clive's secretary has an Olivetti machine on her desk, but Sinclair himself appears to have no machine within reach. Why not?

'Much of the work I do personally is not particularly computer-oriented,' he said. 'And the program I need doesn't really exist yet.'

What he wants, it turns out, is a package which will allow him to type in mathematical equations just as though he was writing them on paper and then allow him to play around with them, altering them, observing results in graph form, etc. It seems there are a couple of such packages available in the States, but neither do exactly what he wants.

Now there's a challenge if ever there was one – write Sir Clive's ideal program! Who knows, he might give you an AI machine for your troubles. Pity I failed maths 'O'-level three times . . .

There's another reason why Sinclair doesn't use computers much: 'I move around a lot and at the moment I can just dump everything in my case. I'm personally pushing very hard for us to get into the portable computer field.' Aha! A portable QL? 'No, there isn't a CMOS version of the 68008 at the moment so we can't do that. However, we will be making a portable Spectrum as soon as possible.'

This led naturally to a crucial problem with portable computers: display technology. Conventional CRT displays are too bulky, delicate and power-hungry, while LCD displays are still unsatisfactory: even the 80-column, 25-line LCD on the ACT Portable isn't too brilliant and that's one of the best around.

Sinclair naturally intends to capitalise on his flat-screen technology. This is rather similar to a conventional TV but is bent at right-angles, producing a very small, flattish, low power display. The screens used in Sinclair's flat screen TVs are too small for computer use, so a projection version is on its way, which will use lenses to back-project an enlarged version of its image onto a more conveniently-sized screen.

'We are now developing a 12-inch full colour display which will have a resolution of 1000 dots by 1000,' he said, 'although the early versions may be 700 by 700. The whole thing will weigh ounces.'

QL Questions

We then turned to the QL. Its history was most unfortunate to start with: production delays, 'kludged' machines, extra ROMs hanging off the back. Altogether rather unsatisfactory, no?

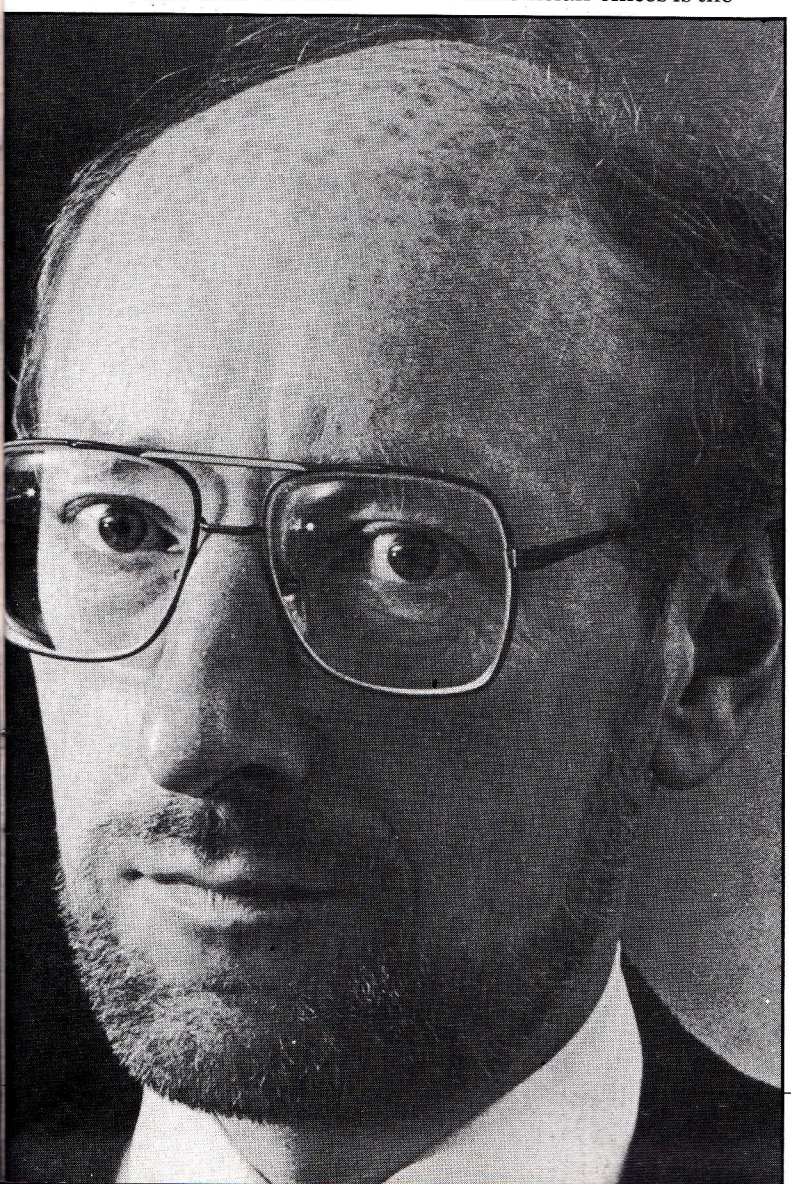
'We had the most appalling problems with it,' he admitted. 'It just turned out to be a bigger job than we thought it was. We launched it too early.' But it does seem that most of the QL's problems have been solved, and as Sir Clive pointed out, some of the criticisms levelled at early machines centred around the way the software was using the Microdrives. In particular the Psion word processing package was almost unusable as it needed to access the drives for each command. This, he pointed out, was no longer the case as Psion had re-written the product to eliminate this. I'll bet the early version was written on a machine with a hard disk!

A problem that the QL faced, which Sinclair's earlier machines hadn't come up against, was that very few people were able to write software for it. The Spectrum, for instance, was based around the familiar Z80 processor, but programmers familiar with the 68000-family processors are few. Hence the long delay in third-party software becoming available.

Now, though, it looks as though this situation is being resolved, with the Sage accounting system becoming available (through Sinclair) in January. There are, says Sir Clive, some 10 companies working on QL software products.

We ended the interview by running quickly through the software which will be coming soon from Sinclair: By the end of this year we should be able to get our hands on a QL Toolkit, a 68000 debugger and a BCPL compiler. A Pascal compiler will be available during January or February and the long-awaited (by me) C compiler will appear in February or March. It will, incidentally, be a version of Lattice C, published through Sinclair. An IBM PC to QL cross-compiler and a 68000 assembler are also on their way.

On the hardware side, the promised half megabyte RAM expansion (not to be confused with the silicon disk) is 'fairly imminent' and a hard disk interface has also been developed.



Aspects of QDOS

Hidden from the scrutiny of all but the lowest level programmers, QDOS lurks unseen yet all-powerful. Adam Denning sinks to the depths and reveals its true nature.

The QL's operating system has been variously praised, criticised and ignored, despite the fact that only software and peripheral manufacturers have so far been able to get to grips with it. As the first real home-orientated multi-tasking operating system it warrants a closer appraisal to see just how well it really does perform.

First things first. QDOS was written by Tony Tebby, formerly with Sinclair Research before he resigned just after the QL's launch. The system was ostensibly written as a test-bed for the SuperBasic interpreter, which would imply that it was never meant to see the light of day publicly. Somehow, it did – not necessarily a bad thing. In fact parts of it are damned good and only a foolhardy comparison with bigger mainframe operating systems would show up any weaknesses. It *does* support multi-tasking, it *is* in the main well written and it *is* satisfactory for a machine of the QL's calibre.

Tebby says that QDOS is not the sort of system which programs run 'under', it is merely a collection of useful routines and utilities which the programmer may find he wants to use. Such is the scope of these utilities that it's almost always certain that a programmer will want to use the facilities of QDOS.

Multi-tasking

This aspect is looked at in greater detail elsewhere in this issue, but it means that the system can run more than one program by allocating a slice of the processor's time to each one. These time slices are small enough to make the system appear to be running all the programs ('jobs') at the same time, but of course only one is running at any one instant. Jobs execute in the 68000's user mode for most of their duration, while the program which controls all the jobs, the scheduler, runs in supervisor mode. A job only goes into supervisor mode for one of a couple of reasons. It may want to ensure that some operation is performed immediately, without danger of the system stopping it halfway through an operation because its time slice is up. Or it might want to do something which won't work unless the system is in supervisor mode, such as a change of interrupt priorities. Each job runs independently by keeping a record of all the 68000 registers in a defined place, so that switching between jobs simply involves saving all the registers to this area for the current job at the time, and then retrieving the registers of another job from its allocated area. There is a physical limit on the number of jobs which can be in the machine at any one time, but as memory is going to be a more immediate constraint it is unlikely that this condition will ever be met.

Extendability

As programs do not run *under* QDOS, it has been written to take advantage of whatever it finds. Adding more memory to the system is an instant and obvious example – as soon as QDOS finds that it's there, it makes use of it and tells everything it controls that the memory is present. Another feature in this ilk is the ability to add I/O devices using what is known as a device driver. QDOS has some of these already built in, such as those for microdrives, the serial ports and the screen. There is also a system which makes provision for adding a device driver, making such addition (for, say, floppy disks) a lot easier. You merely have to control the disk and build the hardware – the QDOS interface is almost completely written for you.

Redirectable I/O

When a programmer wants to talk to the outside world, it has to be done through input and output devices. Although each device – screen, keyboard, Microdrive and so on – is entirely different, there should be no need for the programmer to take account of this. If you're talking to the outside world why should it matter if that outside world is a printer or a microdrive file? By using redirectable input and output, QDOS gives the programmer an environment wherein the outside world *IS* the outside world. Once a name has been given to open a channel to, that channel is the same as any other channel. So, sending a byte down it, is one process and reading bytes from it is another.

The device driver, which the general programmer never sees, decides how to treat the device to which the channel is connected. This arrangement inevitably means that each channel is inherently capable of both input and output, so it is up to the device driver to react accordingly if this is not the case. Unfortunately, QDOS lets itself down in this area – a channel open call is not device independent, particularly in the case of the PIPE device.

Considering the price of the QL, QDOS is wonderful. There are problems here and there, but who can *really* complain?

CASE SOLVED

£69.00
+P&P.

- ***antler*** manufactured specially for the QL System.
- *Angled Keyboard for practical in-case use.
- *Space for 24 Cartridges, QL-accessories & Manual.
- *Size conforms to Airline hand luggage specifications.
- *Fully Guaranteed for 2 years.

Offer is for the case only.
'QL' is a registered trademark of Sinclair Research Ltd.

To: Management Science Ltd. QL Case offer, 17 West Hill, London, SW18 1RB.
Please supply ... Case(s) for my QL(s) at £69 & £4 p&p. (inc. VAT)
I enclose a cheque/postal order for £.....
Made payable to: Management Science Ltd.

Name: .....
Street: .....
Town: .....
County: .....
Postcode: .....

Please allow 14 days for delivery.

QL vs +4

The battle of the giants – Sinclair and Commodore – is about to begin. Will it be a triumph for innovation or consolidation – Paolo Baccanello weighs up the opponents.

Until recently the QL has defied comparison for the simple reason that equivalent systems have cost two to three times its price. This has meant that analyses of the machine and its bundled software have suffered from chronic lack of perspective. The appearance of the Commodore Plus 4 remedies this situation.

The Plus 4 can be seen as the 8-bit riposte to the QL. With a suite of programs contained in ROM, offering facilities for word processing, records filing, spreadsheet operations and the creation of simple graphs.

Technically, the machines have little in common. The Plus 4 uses established technology throughout. The 7510 processor is a revised version of the 8-bit 6502, capable only of addressing a maximum of 64K of memory at any one time. The keyboard mechanism is solid and easy to use and the disk drive, if a little slow, is robust and reliable.

The QL, on the other hand, completely breaks with convention. The membrane keyboard whilst adequate has a quicksand quality which requires some familiarisation. Also the machine, without a cassette interface, relies exclusively upon microdrives. Whilst these have a combined storage capacity equal to the Commodore disk drive they are slow, and are not noted for their reliability. Also the cartridges that they use are byte for byte more expensive than disks.

On the plus side, however, the QL comes with 128K RAM built-in, of which 90K is available even in high-resolution mode. This compares very favourably with the Plus 4's 60K maximum user RAM. Furthermore, in terms of expansion possibilities, the QL with its two standard RS-232-C ports may more easily be linked up to a number independently manufactured devices.

Also, whilst the QL sound facilities are considerably more limited than those on the Plus 4 its video display is far superior. Far higher resolutions are possible and, even though the QL supports only eight colours these may be varied in contrast and composition to create a very broad spectrum of differing shades and hues.

Finally, and most importantly, the QL's 68008 32-bit processor is capable of operating at very much faster speeds than the 7512 as well as processing larger chunks of data. In real terms this means that the QL can support programs of a sophistication and quality far superior to those possible on the Plus 4. This is reflected both in differences in the BASICs that either machine supports as well as the bundled software.

Specification	Plus 4	QL
Microprocessor	7510 (8bit)	68008 (32bit)
Clock Speed	0.89MHz-1.76MHz	7.5MHz
Keyboard Mechanism	Typewriter (67key)	Membrane (65key)
Storage Medium	Cassette, Disk (170K)	Microdrives (2×100K)
ROM	32K	48K
RAM (expandable)	64K	128K (640K)
Expansion	Serial, cassette, Disk, ROM slot, 2×Joystick, Video, UHF	2×RS232-C,Expansion Bus Slot, ROM slot, 2×Network, 2×Joystick ,UHF, RGB
Screen Resolution	320×200 (10K)	512×256 (32K)
Text Display	25×40	25×85
Sound	4 channels	1 channel
Colour	16 (8 luminances)	8
Languages	**Plus 4**	**QL**
BASIC	V3.5-Extended	SuperBasic-Structured
Other	Assembler, disassembler, Machine code Monitor	None
Editor	Full screen	Line

Bundled Software

Both Plus 4 and QL support slow BASICs. These incorporate commands which give the user easy access to the machines' respective special features such as disk or microdrive handling, graphics, sound and error trapping. However, only SuperBasic goes beyond these essentially conventional facilities permitting the user to construct well organised and self-documenting programs using definable procedures similar to those found in the much praised BBC BASIC.

On the side of the Plus 4, whilst its BASIC may not be as powerful and as versatile as the QL's, a full screen editor makes programs easier to maintain and a built-in monitor, assembler and disassembler provides a useful, albeit somewhat primitive (no trace facility), vehicle for the user to extend his programming experience into the realms of machine code.

Whilst technical and language considerations have an important place in any assessment of a machine, the principal determinant must be quality and the quantity of software available for it. This is especially true of machines geared for the professional and small business user who will be looking to put his machine to work with a minimum of effort but maximum return.

As the QL and Plus 4 are recent introductions in the marketplace both include a suite of practical programs available for immediate use. Whilst these may not be representative of all the software yet to come they do illustrate inherent shortcomings in either machine and such provide a useful indicator.

The Plus 4's word processor, benefiting from faster access times afforded by a disk drive and the fact that the program is permanently resident in ROM, is faster at carrying out commands. Furthermore, as the word processor may be easily linked to the database printing out mailing lists or circulars is quite straightforward.

QL Quill, on the other hand has a more advanced and larger repertoire of functions which afford considerably greater control over printed output. It has the benefit that all operations have their effect on screen so that what you see is what you get. It is capable of displaying the entirety of a document on screen without the need to scroll across its width.

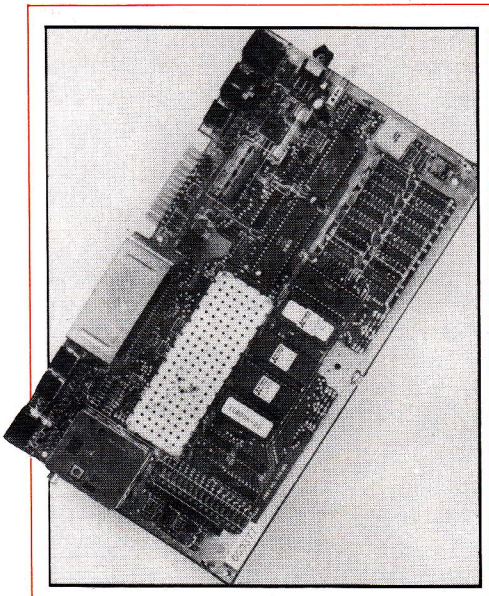
In many respects the two word processors are suited to very different applications. The Plus 4's with its clumsy command entry and awkward screen display but fast access times and easy integration with file manager is ideal for mail shots where short standard forms are repeatedly printed out. Quill on the other hand is with its slow access times and poor integration but excellent and easy to use formatting facilities lends itself to the production of lengthy reports, books and even articles.

On the Spreadsheet the Plus 4 faster access times are not so pronounced. Indeed the packages single advantage would appear to be its capacity to show the spreadsheet (seven rows deep) on a split screen along with the word processor. Beyond this the package is very limited.

On the other hand Abacus has a great deal to offer. It is capable of handling more information. ROW and COLUMN commands permit a vast amount of calculation with a minimum of effort. Heading may be used as labels and thereafter incorporated within formulae. Complicated formulae may be entered in a single cell without having to be broken down across a number of cells. Month names may be easily inserted. Text may be easily justified and the spreadsheet may be printed out directly and control codes may be passed across. Finally, Abacus provides a number of extremely specialised features such as lookup tables, conditional statements that mix text and numeric arguments together and even a facility to create user-defined prompts.

Business Graphics

The Plus 4 graphics package is little more than a cosmetic extra attached to the spreadsheet. Quite simply it permits data from a single row to be used to create a low resolution bar graph. Provided that the information is stored on the first row on the sheet the graph



Inside the Plus 4 – nothing new here.

may then be transferred across to the word processor and from there printed out. Automatic scaling is used to create a graph 20 characters high between maximum and minimum values. However no labels of numeric value are attached and the horizontal axis is permanently fixed at 18 characters wide.

Psion's EASEL, unlike the Plus 4's graphics package stands on its own right. Data may be entered, saved and loaded independently. Again, scaling is automatic but in this case along both axes and limited only to the amount of pixels that may be displayed on screen. Furthermore more than a single set of figures may be displayed at any one time in a variety of different formats (Bar and Pie charts, line graphs). Background and foreground colour may be altered, labels, text and keys may be attached and the result may be printed out directly.

Filing System

The Plus 4's database resembles a card index in its operation. When number of fields per record and the size of each field has been defined the program formats a disk in such a fashion that equal amounts of storage are reserved for each record. Once formatting is complete the program returns the maximum number of records available for use. Thereafter the user may enter and amend records, search and sort fields and transfer information to the word processor to be printed.

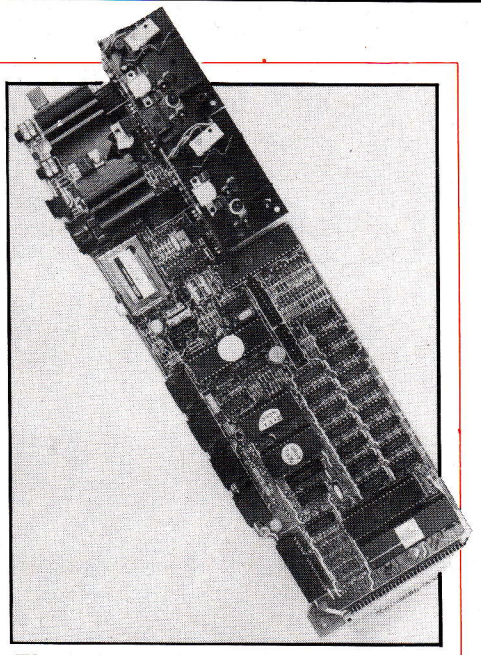
Psion's Archive performs all these functions at its lowest level. Beyond this it provides a well structured programming language that enables the user to define complex filing operations and thereafter call them by name. Further the package permits more than one file to be opened and allows for additional fields to be added to current databases. Also, variable field lengths within records greatly simplify file creation and make efficient use of the limited memory available on microdrive (100K). To speed up execution, all operations are carried out in RAM and the database is only saved at the end of a session.

Integrated Software

Both suites of programs purport to be integrated, however in either case this is true only to a limited degree. On the Plus 4 information may be passed to the word processor from the other packages but the flow is one way. With the programs permanently resident in ROM the transfer of information is virtually simultaneous.

With Psion's program the transfer of information is slower as data must be transferred to an export file on microdrive and loaded back in after the next application has been run. As the user must ensure that information is stored in the correct format the process is also quite complicated. However, the flow of information between all packages with the exception of Easel-Quill is two-way making this feature very much more powerful than its equivalent on the Plus 4.

The Plus 4's packages provide little or no on screen aids to facilitate data or command entry. All information must be gleaned from the manual. Furthermore there are very few procedures that are common to all the packages. Overall,



The QL's PCB (dongle version).

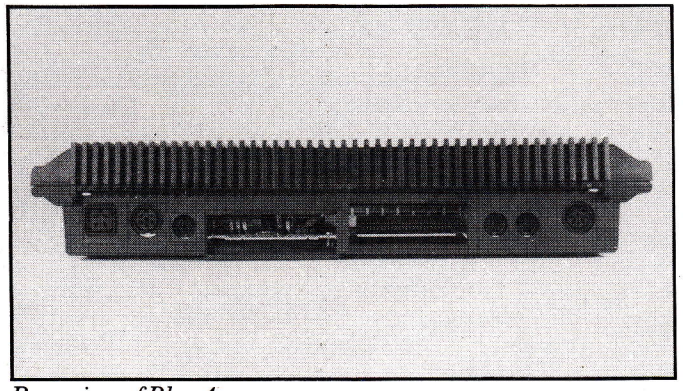
despite being unsophisticated the suite is unfriendly.

Throughout all their packages, Psion have tried to standardise formats enabling the user to very quickly accustomise himself to the displays and providing him with a knowledge of where to look for particular aspects of a package. Careful use of colour throughout not only makes the packages more attractive but provides guidance as to which area of the screen does what. All packages have extremely well structured commands and with the exception of Archive supply prompts for every step in the command process and defaults where necessary.

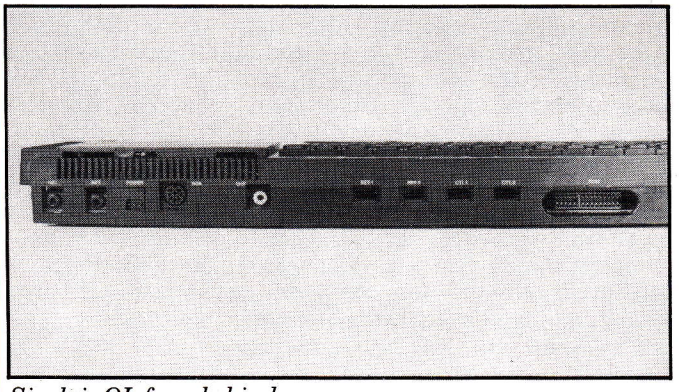
The Last Word

The QL is a considerably more powerful machine than the Commodore Plus 4. Furthermore, the bundled software that comes with it is of a far superior standard in terms of capability and performance.

However, in terms of reliability and speed of execution the Plus 4's unsophisticated set of packages comes up trumps. This has little to do with the programs themselves and a great deal to do with the inadequacies of microdrives as a storage medium compared to disk drives.



Rear view of Plus 4.



Sinclair QL from behind.

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 **WDSoftware**, Hilltop, St Mary, Jersey, C.I. Tel: (0534) 81392

★ SNOWSOFT ★

SPRITE DESIGNER: Create fantastic multi-colour, multi-size cartoon type characters for inclusion in your own programs with the supplied M/C routines. Great fun! – requires no previous knowledge. **£15.00**

Also discover the full power of your QL, phone for details of a **'68008 Tutor Program'** for beginners wanting to learn machine code programming.

★SNOWSOFT★ Tel: 0325 312417

6 Bousfield Crescent, Newton Aycliffe, Co. Durham DL5 4HZ

QL PAYMASTER

A COMPLETE PAYROLL/TAX SYSTEM FOR THE QL

Covers all tax rates and codes, pay rates and NH codes up to 'C' contracted out.

All tax bands, percentage tax rates, NH codes, etc, alterable by the user. No need to return program for updating.

Printout replaces normal tax records for up to 50 employees.

SSP handled by the system with printout of amounts paid, totals carried forward.

Monthly tax-liability printout, year carried forward totals; generally, and for each employee.

Simple to operate with comprehensive on-screen prompts.

Program on MICRODRIVE cartridge complete with instruction booklet. Price, including VAT, £59. Instruction booklet only – £3, which is refundable on program purchase, from:

WILLDEN SERVICES LTD
2B Beaconfield House, Beacon Road
Crowborough, East Sussex TN6 1AX

QLPRINT

Your QL documents/listings printed £2 per file any length inc. p&p. Send your tape and cheque or PO to

QLPRINT, Chubbs Cottage, Buddle Lane, Hatherliegh, Devon EX20 3HX or phone Hatherliegh 535

DEQUILL- SCREEN ★ ★ EDITING ★

- ★ **EXPORT QUILL PROGRAM & DATA FILES TO SUPERBASIC**
- ★ **OPTIONAL SYNTAX VALIDATION & COMMENT STRIPPING**
- ★ **SEND ONLY £9.95 TO: SOFTSELL, 3 MALLORY COURT,**
- ★ **CONGLETON, CHESHIRE CW12 4NW (02602 77173)**

PROVE TO YOUR FRIENDS THAT YOUR COMPUTER IS REALLY USEFUL.

Plan their garden with our data base on plants. Send £9.95 for data base on Sinclair QL microdrive cartridge from:

Llangeitho Nurseries, Llangeitho, Tregaron, Dyfed SY25 6QG or phone 097 423 223.

QL GRAPHIC SCREEN DUMP

100% m/c loaded as resident procedure for Epson-type printer. Also monitor program includes Hex-memory dump to screen or printer, alter memory, Hex/Dec Converter, etc. All on one cartridge supplied with instructions.

Send cheque/PO for £9.95 or SAE for details to:

LOTUS SOFT
43 Maes Road, Llangennech, Llanelli, Dyfed SA14 8UH
Tel: 0554 820473 or 0554 759624

QL FOR BUSINESS

A programme of linked courses.
Introductory and Advanced Applications

OWL COMPUTER TRAINING
044 282 7302
FREEPOST, TRING, HERTS HP23 4BR

Strathern UTILITIES for the QL file Utility Program

Bidirectional scrolling directory with file length, copy complete cartridge, single key multiple copy and delete, full wildcard named copy and delete, print directory, print file block sector map, single key multiple format, verify file.

BOOT PROGRAM

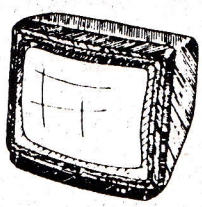
Bidirectional scrolling directory with file length. Single key load or run.

£8.50 + 25p p&p for both

M. STRATHERN
174 Richmond Road
London E8

Have a Quite Lovely Christmas

Microvitec Colour Monitor with cable for QL
£265 FULLY INCLUSIVE



Sinclair QL Computer

£389-99 FULLY INC.

Pack of 4 microdrive cassettes in wallet £19
QL Toolkit £14-95

QL Centronics printer interface £37

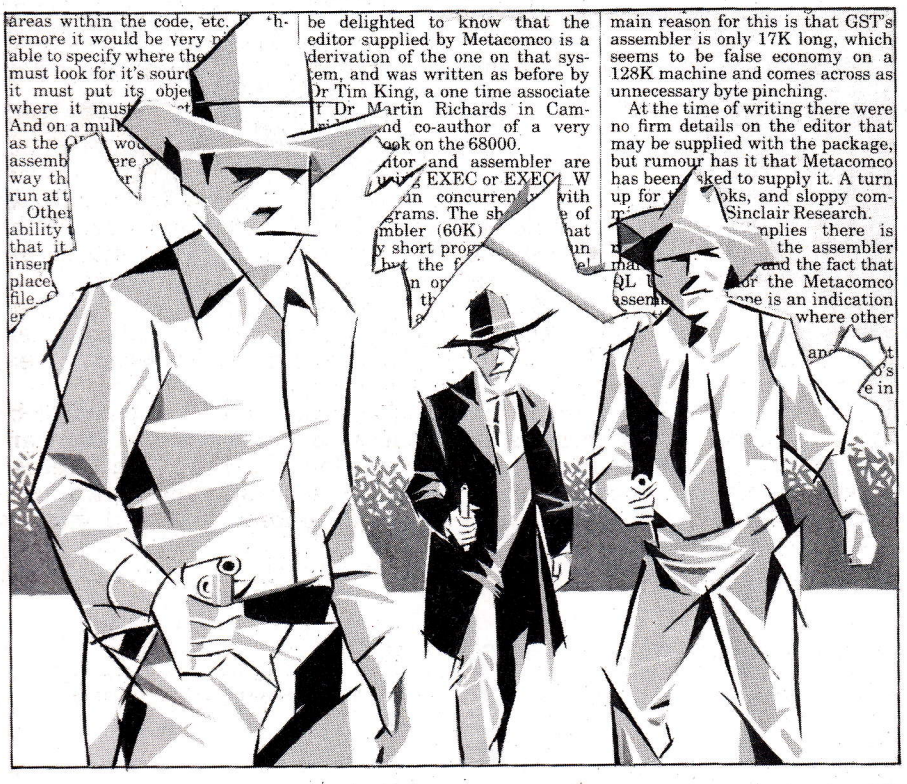
PRICES INCLUDE VAT & DELIVERY

COMPWARE (0270) 582301

57 REPTON DRIVE
HASLINGTON CREWE CW1 1SA

Please send __ QL computer(s),
__ QL cub Monitor(s) and
__ QL Centronics Interface(s).
I enclose a cheque £ _____
NAME : _____
ADDR. : _____

Gunfight at the



areas within the code, etc. ... h-
ermore it would be very n...
able to specify where the...
must look for it's sour...
it must put its obje...
where it mus...
And on a mul...
as the Q... wo...
assemb... ere ...
way th... r ...
run at t...
Othe...
ability t...
that it ...
inser...
place...
file_...
er...

be delighted to know that the editor supplied by Metacomco is a derivation of the one on that system, and was written as before by Dr Tim King, a one time associate of Dr Martin Richards in Cam... and co-author of a very ...ook on the 68000.
...itor and assembler are ...ing EXEC or EXEC_W ...in concurren... with ...grams. The sh... e of ...mbler (60K) ...hat ...y short prog... an ...but the f... el ...n op... th... a...

main reason for this is that GST's assembler is only 17K long, which seems to be false economy on a 128K machine and comes across as unnecessary byte pinching.
At the time of writing there were no firm details on the editor that may be supplied with the package, but rumour has it that Metacomco has been ...sked to supply it. A turn up for t... ...oks, and sloppy com... ...Sinclair Research.
m... ...nplies there is ... the assembler ma... ...and the fact that QL ... ...r the Metacomco assem... ...one is an indication ... where other ...an... ...t ...o's ...e in ...

Illustration by Stuart Briers.

GST Corral

'The word was out – Doc Fenton and the GST marshals were gunning for that Denning boy. The row about last month's review meant the time for talking was over. There was only one way to settle it now . . .'

Last month, the reviews of GST's assembler and 68K/OS stirred up quite a reaction at their Cambridge offices. Subsequent correspondence revealed several aspects of these products which went unmentioned and it was clear that a reappraisal was in order.

So, to clarify matters, Adam Denning re-assesses the assembler and GST remark upon some features of K/OS . . .

The 68000 assembler comes on a microdrive cartridge along with a reasonably large manual and can be started either by using the EXEC / EXEC_W commands or by loading and running the boot program. Also included on the cartridge are a number of utilities to help the programmer. One of these is the Metacomco screen editor, which may not have been GST's first choice. Along with the editor are two very large files containing all the QDOS trap keys, system variables, equates, offsets and masks. Any of these can be used during assembly with the INCLUDE assembler directive.

Other files supplied are the aforementioned boot program, a cloning program and a small collection of library routines. The latter can also be included in an assembly if required, or the library can be extended to include user-written subroutines.

When the assembler is run it first opens a small window and asks for a console definition to use. Pressing ENTER alone causes the system to use the default of CON_. The heading is then printed and a prompt appears asking for the command line. This method of invoking assembly still seems a little unfriendly, but then it's really a matter of what one expects.

The command line is used to tell the assembler what file to assemble, what listing to produce, where to send any listing and so on. Numerous assembler options can also be included here.

Once assembly has been started it ends very quickly – this assembler is extremely fast! As files to be assembled must be postfixed with **_asm** and the resultant listing and binary files normally have extensions of **_list** and **_bin** respectively, the command line could be as simple as **mdv1_asmtest**, as the defaults would be used in each case resulting in a code file called **mdv1_asmtest_bin** and a listing file called **mdv1_asmtest_list**.

Naturally enough, the user is able to alter this in most respects.

The listing produced is fairly standard, giving a heading and the resultant hex code; though it also includes the time of assembly and a full symbol table listing at the end of the assembly. Contrary to our earlier reports, the assembler also produces error messages – very comprehensive ones in fact. Indeed, considering the price of £39.95 and the fact that it can theoretically assemble larger files than Metacomco's, it's difficult to recommend one over the other.

Featuring K/OS

Many prospective purchasers of QLs are more interested in raw hardware power than SuperBasic or Psion software. These users will be writing software for themselves or to sell to others as a turnkey system. Here, 68K/OS is a serious proposition, because it offers a lot of features QDOS doesn't have.

The first thing noticed when running 68K/OS is that the hardware power is much more impressive. Microdrives run faster, programs load quicker and the response is much faster. The main feature, however, is the windowing facilities. Each task under K/OS has its own window that the user can expand or shrink up and down the screen as is seen fit. Each horizontal window can be split vertically into further sub-windows – it's not quite up to *Lisa* standard, but it's still very flexible.

The editor uses between 2 and 4 separate windows; one for the main text, a single line window for the ruler line and a menu window on the right-hand side which can be removed when the user is familiar with the commands. Other menu windows appear after a help key prompt.

The multi-tasking is a delight to use. You can run two copies of the editor or transfer information along 'pipes' from one file to another. For example, selecting blocks from a standard file and merging them into a letter is easy. 68K/OS also offers full multi-tasking. Once there were 15 separate tasks running, which is more than enough for any practical application. The only real limit is screen size.

A lot of the programs use a standard menu facility which is actually built into the operating system. This means using the menus on the standard products is easy because the control commands are all the same. Users can write programs to call the menu handler which reduces the code to set up a menu to a few lines.

On the negative side there's not a lot of software available for 68K/OS yet. GST have an assembler on microdrive (a ROM-based version will be available at the end of the year) and a word processor. They expect to get C, Pascal and FORTRAN up early in 1985.

Another slight irritation is that QDOS and K/OS tape formats are not compatible. GST said that this was essential to improve microdrive performance but it is a pity all the same. A utility *will* be available to read and write QDOS files under K/OS, but it would be much easier if the two formats were compatible.

Supplied on a small board, 68K/OS plugs into the expansion bus, with a switch that flicks from QDOS to K/OS. The board has two slots for ROM software (such as the assembler) and comes with a microdrive cartridge with 12 utilities. Users can easily blow their own programs into ROM and put them onto the card, giving instant access to programs.

Generally the purchaser who wants to run standard software won't find K/OS of much interest, but for software developers with the QL in mind it is well worth a look.

GST Computer Systems Limited

91 High Street, Longstanton, Cambridge CB4 5BS Telephone: Crafts Hill (0954) 81991 Telex: 817057 (GSTUK G)

The Editor
QL User Magazine
Scriptor Court
155 Farringdon Road
London EC1R 3AD

24 October 1984

Dear Sir

I read with interest your article on 68K/OS in last month's QL User. There are some basic misconceptions in the article which I must ask you to rectify.

1. 68K/OS is not seen as an alternative to QDOS. It is intended for the serious software writer who doesn't want SuperBasic or Psion software and wants to develop his own software.

2. The article fails to mention the advantages of the features of 68K/OS

   - full genuine multitasking
   - full generalised windowing
   - improved speed and microdrive performance
   - built in menu handling
   - compatibility with Motorola Pascal and Assembler

No-one else who has seen the product has expressed any negative view at all and we hope your readers will look at the product themselves before they decide.

Yours faithfully

Dr J S Fenton
Technical Director

Directors: J.S. Fenton MA, PhD; L.F. Fenton, R.M. Keyworth LLB Reg in England No: 1426456 Reg Office as above VAT Reg No: 249 9838 91

Unix Potential

With a 68000 processor, the QL cries out for some exciting 32-bit firmware. Paolo Baccanello considers the case for Unix.

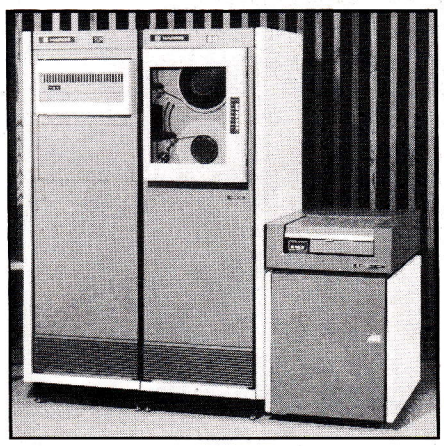
The QL's built-in operating system, QDOS is unique. QDOS is unlikely to appear on any other machine. The case for some kind of standard operating system to complement or replace QDOS is therefore particularly convincing, both from the point of view of software houses intent upon getting as much mileage from their programs as they possibly can and from users wishing to expand their systems.

Unix, an operating system developed by Bell Laboratories, is well placed to enter the fray. Already Unix and its look-alikes (Venix, Xenix) are well established on more powerful 68000 machines. A move down-market would not only extend the user base, but also reinforce the system's dominant position.

Sharing The Load

Unix offers many advantages over competing systems. To begin with it affords true multi-user and multi-tasking facilities. On the one hand it permits a number of computers to be linked together to share data and hardware resources. On the other, it allows individual users to run more than one task at any one time. Furthermore, as it is written mostly in machine-independent code, the system is extraordinarily portable. Unlike CP/M or MS-DOS, applications written within Unix may run, almost without modification, on machines with different microprocessors. Coupled with the excellent communications facilities that Unix affords, users would have easy access to a potentially enormous and rapidly expanding pool of software.

Unix also provides the best possible environment for program development. The system includes a C compiler and an arsenal of built-in reusable utilities and system commands. An advanced filing system permits data and program files to be split up into related groups or 'directories' on disk. Such directories are hierarchically organised in a kind of inverted tree structure, providing a simple and coherent layout within which to develop even the most complicated of applications. There are also a range of built-in software protection facilities, permitting access security codes to be set for any file or directory.



Design Drawbacks

Designed as a powerful tool for computer professionals, Unix's principal shortcoming lies in not being user-friendly. Unix's abbreviated commands are not self documenting and many of the utilities provided require a thorough understanding of a number of complex concepts. However, the system permits programmers to construct a simplified menu-driven command shell. Inside this, users may carry out all the operations required with relative ease.

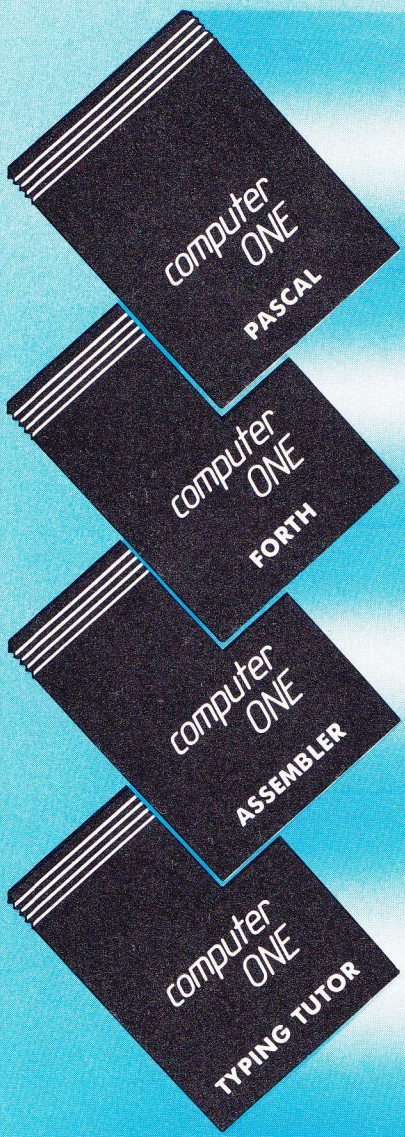
At the heart of Unix lies the 'kernel'. This controls machine resources, manages the processor, directs I/O and organises the filing system. The entire system is made up of some 20,000 lines of code. Around a thousand are written in low level assembler and the rest in C. To transfer the system to a new machine, the low level code must be rewritten along with a further 4000 lines in C relating to processor and memory management.

The major obstacle to implementing Unix on the QL is its size. Consisting of approximately 200 programs, the system would require some 500K simply to house it. Programs would also require more than the 128K RAM currently available in which to run. As a consequence, a 256K RAM extension and a 10Mb Winchester hard disk would seem to be required. Whilst both are now available, the overall cost of the system is close to £2,000. The QL being aimed at the home and small business-user, such a cost is likely to prove prohibitive.

Whilst most QL users are unlikely to come into contact with Unix directly, however, there can be little doubt its influence will be felt. Already smaller operating systems are evolving into toned-down Unix look-alikes. For example, MS-DOS – the dominant 16-bit operating system – mimics the kind of software development environment afforded by Unix.

QL POWER

AT YOUR FINGERTIPS FROM COMPUTER ONE



PASCAL £39.95 Inc. VAT & p&p.

Computer One PASCAL is a powerful implementation of this classic programming language. Produced specifically for the QL, this package provides a professional and highly educational programming system. Complete with comprehensive 80 page manual, this is the ultimate language package that no QL enthusiast can do without.

INCLUDES: ★ Full Screen Editor ★ QL graphics and sound extensions
★ Example programmes ★ Complete 87 page User Guide
★ Full implementation close to ISO-standard ★ Typically 15x faster than BASIC

FORTH £29.95 Inc. VAT & p&p.

FORTH is easy to learn, highly efficient, and allows you to explore the full capabilities of your QL. Computer One FORTH is a full implementation of the latest FORTH-83 standard with graphics and sound extensions, and may be used to produce machine code applications for your QL. Ideal for writing real-time or games programs.

INCLUDES: ★ FORTH-83 system ★ Forth Screen Editor ★ 48-page manual
★ QL-graphics and sound extensions ★ Example FORTH programs

ASSEMBLER £29.95 Inc. VAT & p&p.

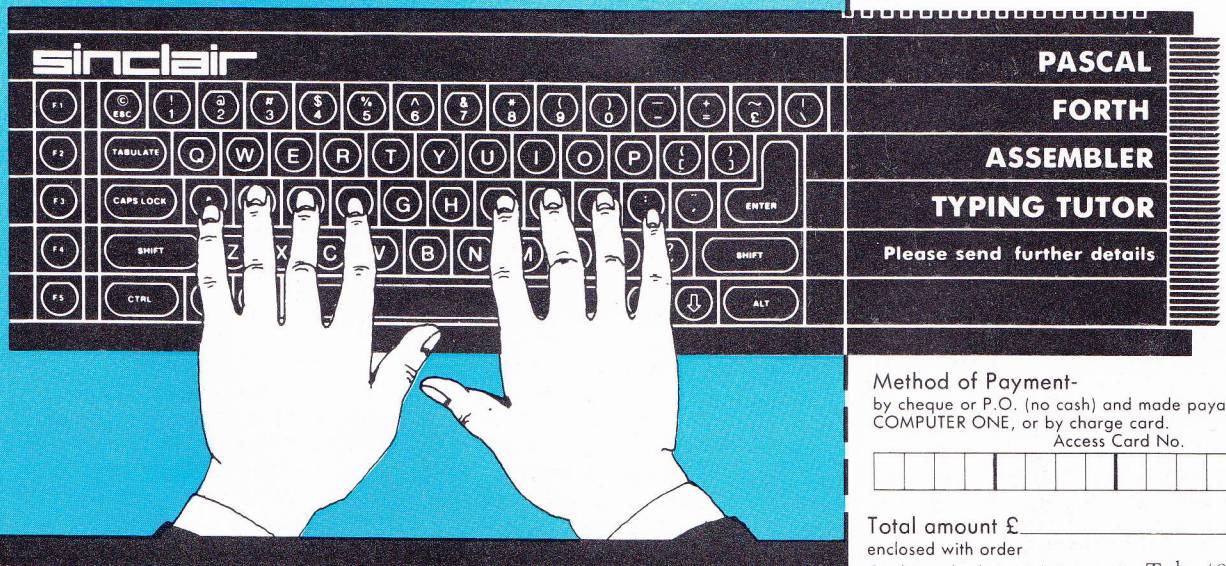
COMPUTER ONE Assembler is a comprehensive assembler program toolkit providing a professional multitasking assembler written in machine code for speed and compactness. An ideal package for the machine code programmer, allowing you easy access to the QDOS operating system, and integration to your Superbasic and Pascal programs.

INCLUDES: ★ 68008 Assembler ★ Full Syntax Checking ★ Full screen editor
★ Comprehensive User Manual ★ Integrates to Pascal and Superbasic

TYPING TUTOR £19.95 Inc. VAT & p&p.

Let your Sinclair QL teach you to type! Computer One's Typing Tutor is a fast, easy and enjoyable way of learning true touch-typing on your QL.

INCLUDES: ★ Self-paced tutorials
★ Speed and accuracy diagnosis
★ Interactive colour graphics
★ User Guide



computer ONE

SCIENCE PARK, MILTON ROAD, CAMBRIDGE

Name_____

Address_____

_____

Signed_____

	Tick Order
PASCAL	
FORTH	
ASSEMBLER	
TYPING TUTOR	
Please send further details	

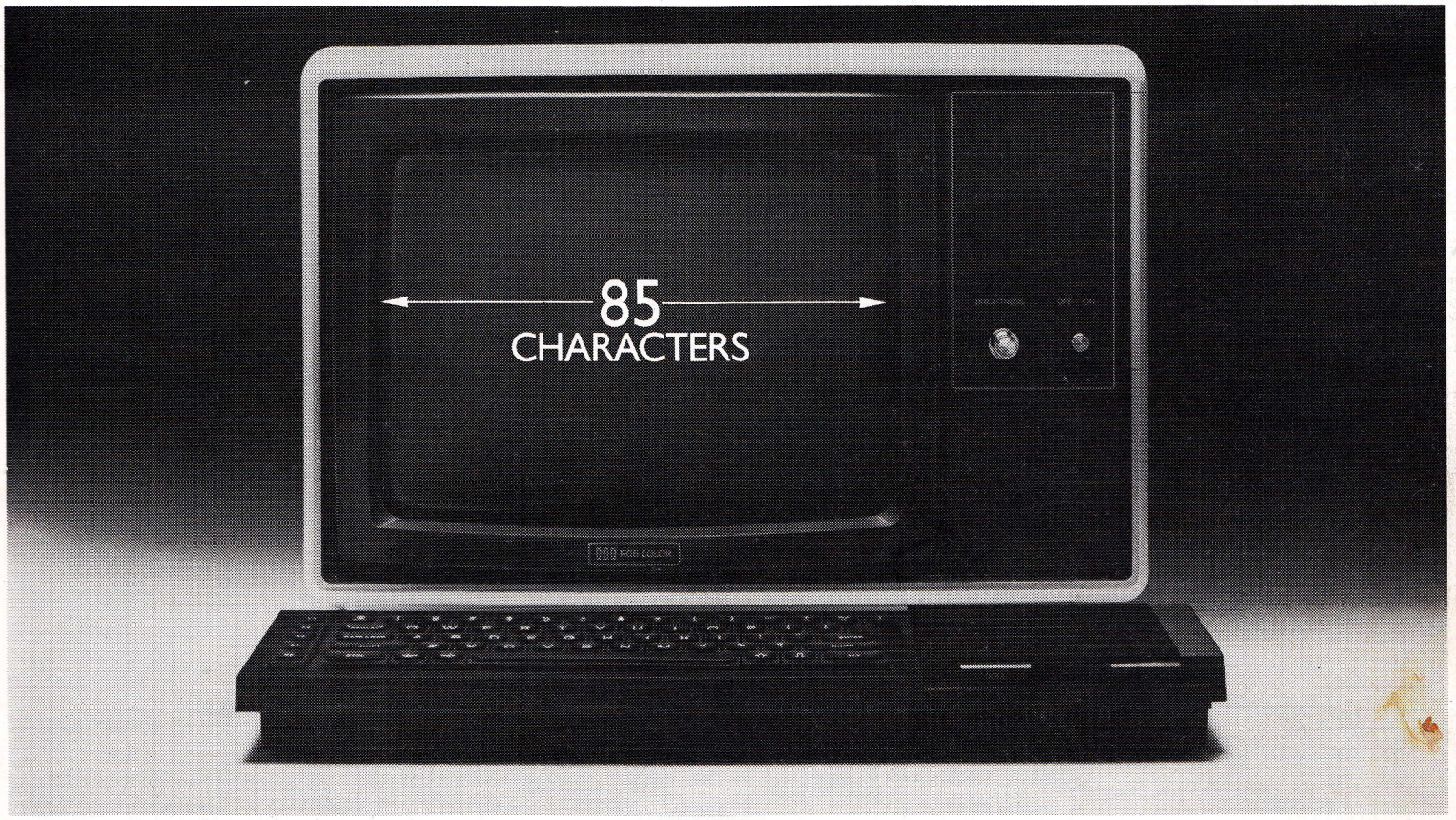
Access

Method of Payment-
by cheque or P.O. (no cash) and made payable to COMPUTER ONE, or by charge card.
Access Card No. QL/12/84

Total amount £_____
enclosed with order

Credit card sales: UK buyers only. Tel: (0223) 862616

A HIGH RESOLUTION MONITOR FOR THE SINCLAIR QL AT A LOWLY £299.95 (INC. VAT).



This is the much acclaimed JVC colour monitor from Opus.

And it gives Sinclair QL owners a lot more than just excellent high resolution.

As you know, the QL gives out information in 85 characters on the screen instead of the standard 80 characters.

And the loss of those five characters means that ordinary monitors simply can't show you the whole picture.

Opus, however, have overcome this problem by adapting the JVC, via a special interface and cable, so that the QL can now work 100% efficiently on it.

All for the lowly price of just £299.95 including VAT, carriage and, of course, a year's full guarantee.

You can order it by post by filling in the coupon below and sending it to: Opus Supplies Ltd., 158 Camberwell Road, London SE5 0EE. (You'll receive it within ten days by courier service.) Or you can telephone 01-701 8668 and pay by credit card.

Alternatively, you can buy at our showroom between 9.00 – 6.00pm Monday to Friday, or 9.00 – 1.30pm Saturday.

MODEL REFERENCE	1302-2 High Resolution
RESOLUTION	580 x 470 Pixels
C.R.T.	14"
SUPPLY	220/240v. 50/60Hz.
E.H.T.	Minimum 19.5kv Maximum 22.5kv
VIDEO BAND WIDTH	10MHz.
DISPLAY	85 characters by 25 lines
SLOT PITCH	0.41mm
INPUT: VIDEO	R.G.B. Analogue/TTL Input
SYNC	Separate Sync on R.G.B. Positive or Negative
EXTERNAL CONTROLS	On/off switch and brightness control

To: Opus Supplies Ltd., 158 Camberwell Road, London SE5 0EE.
Please send me:

_____ High Resolution Colour Monitor(s) at £299.95 each (inc. VAT).

I enclose a cheque for £_____ Or please debit my credit card account with the amount of £_____ My Access/Barclaycard (please tick) no. is_____

Name_____

Address_____

_____

_____

Telephone_____

QL/12/84

Opus.
Opus Supplies Ltd.

WHAT MAKES VIRTUALLY EVERY HOME COMPUTER CLICK?

Simple! A fully compatible printer or to be more precise the DWP1120 from Quen Data.

The DWP1120 is a real value for money, letter quality daisy-wheel that's so versatile, it can be interfaced with virtually any micro on the market.

Originally designed for professional users, the DWP1120 combines stylish good looks with total practicality. With features you'd only expect to find on much more expensive machines it's a powerful companion for any home micro, giving you complete word processing, documents and data listing facilities all at the touch of a button. In fact, whatever your particular printing requirements, the Quen Data DWP1120 is almost certain to make your computer click . . .

JUST SOME OF THE POPULAR MICROS EASILY INTERFACED WITH THE DWP1120

*SPECTRUM
BBC
*COMMODORE 64
ORIC
DRAGON
APPLE
SIRIUS
SAGE
OSBORNE
PET
NEW BRAIN
SINCLAIR QL
*Special interface.

SPECIFICATIONS

Printing speed of 18 cps, bi-directional and logic seeking.
Exchangeable 96 character print wheels.
Friction feed as standard, tractor feed and single sheet feed are easily attached options.
Interface examples; either 8 bit parallel/ centronics, or RS232C/V 24 on Sinclair QL or standard centronics.
Characters per line, 120c-15p with character spacings of 10.12.15 pitch and proportional spacing.
CONTACT ANY OF THE DEALERS LISTED BELOW OR QUEN DATA DIRECT FOR FULL SPECIFICATION DETAILS.

FLEXIBILITY AT THE TOUCH OF A BUTTON

★ HOME CORRESPONDENCE
★ BUSINESS USE
★ DOCUMENTATION
★ WORD PROCESSING

Instantly compatible with sophisticated word processing programmes such as Tasword 2, Wordwise and Wordstar, etc.
12 months full replacement warranty
DWP1120 RRP £295 or less
Also available: the DP1100 and shortly available a complete new range of competitively priced Dot Matrix printers.
Dealer enquiries welcome.

Available from: Datafax Systems Ltd, Basingstoke (0256) 464187; Rams Computer Centre, Bletchley (0908) 647744; Computer Intelligence, Weybridge (0932) 49723; Leigh Computer Systems, Hinkley (0455) 631579; Soft Shop, Chalfont St. Peter (0753) 889010; EEC Distribution, Barnes (01) 441 1590; Southern Software, Maidstone (0622) 51736
Plus Spectrum Dealers and all major computer retailers and department stores.

QUEN-DATA THE STATUS MACHINES.

Quen Data Business Machines UK Ltd.
25 Clarke Road, Mount Farm, Bletchley,
Milton Keynes, Bucks MK1 1LQ
Telephone: (0908) 649412
Telex 826343 QUEN UK G

Dialects on Demand

Adam Denning and Paolo Baccanello get to grips with four new languages – BCPL, LISP, Pascal and APL.

Up until now the first software available for any new micro has been games and 'applications' software, such as word processors and accountancy packages. In the case of the QL, the latter are rendered less important by the bundled software supplied with the machine. That leaves the games market, which could be expected to be flooded. However, this is not the case. The first major products have all been utilities. And after assemblers and operating systems, it's time to look at some languages.

Computer languages are an important part of any micro's repertoire, as there's always someone who won't touch a machine until language X is available for it. Well, the QL now has BCPL, Pascal, LISP and APL, and more are coming – fast. Lattice is reputed to be developing a version of its C compiler for the machine and Metacomco is developing a full ISO Pascal compiler and planning a version of the renowned Cambridge LISP system. In the meantime that company has used its BCPL knowledge to write a BCPL compiler and a LISP interpreter.

BCPL

The forerunner of C, BCPL is the hacker's delight. It is noted for its lack of data types and its brilliant compromise between a high level language and a low level systems language – many prefer it to C for just these reasons.

A BCPL program is liable to be very compact and generally quite efficient. Rather than typing its data structures, BCPL provides just one – the word. 'Word' length is implementation dependent, being 32 bits on the QL and 16 bits on the BBC Micro. The word can hold any sort of data, and the compiler does not need to know what the data refers to. This is because it is the programmer's responsibility to know what he is using a word to represent. This places the onus of meaningful programming firmly on the programmer's shoulders, which many feel is just how it should be. Words can hold addresses, acting as pointers, or numbers, making them integers, or characters, making strings. By declaring an area of store as being a contiguous series of words (of arbitrary length) and putting a reference to the first word of the series in a variable connected with an identifier, we have what is known as a Vector. This is the nearest equivalent BCPL has to an array. Variables and Vectors are declared with the LET statement, so that:

```
LET serial.port, count = ?,0
```

declares a variable **serial.port**, but does not initialise it. At the same time a variable **count** is declared, with an initial value of 0. Similarly:

```
LET input.vector = VEC 5
```

declares a vector of 6 words (BCPL vectors start at their zero'th element) pointed to by **input.vector**. To access elements of a vector we use indirection, with the (!) pling operator. Thus **input.vector!0** is the first element of the vector and **input.vector!5** is the last. The ! is associative in this context, so **input.vector!0** and **0!input.vector** refer to the same thing. Naturally enough, strings would be held in vectors, but if we had one character to each word we would waste three bytes (in this implementation) per character. For this reason various procedures exist to pack a string into a vector, one character per byte, and a further indirection operator, %, allows us immediate access to each character. In the case of a true string, **string%0** contains the length of the string and **string%1** to **string%(string%0)** contain the characters of the string.

The language is block structured in many respects, a block being defined as a group of statements and declarations between **section brackets**, **$(** and **$)**. Procedures can be defined with LET and AND, depending on the scope required, and variables can be local to each procedure. Procedures which return results are called functions, and the result is returned either by use of the = sign, or more usually by returning the value of a VALOF block containing one or more RESULTIS statements. Even BCPL's case statement SWITCHON can return a value, so that a routine to report an error could use SWITCHON to return a pointer to a string to print depending on the error generated.

Perhaps the most fundamental feature of BCPL programming is the **global vector**, an area of store dedicated to holding the values of variables and the addresses of routines which have been declared as global to the whole program. This mechanism allows for elegant overlays, where a program can load in another segment of code and communicate with it through the global vector. By the same token, assembler routines (or even routines written in another language which follow the right conventions) can be included in a program just by passing addresses, or whatever, through the global vector. Ever since this writer discovered BCPL, its clarity and ability to do anything have proved a constant delight.

The Metacomco QL Compiler comes on a microdrive cartridge with a manual which assumes you know how to program in a high level language already. BASIC programmers beware!

The compiler contains the standard syntax, translation and code generation stages, resulting in a compact code which must be linked with the global vector and runtimes using the supplied linker. The resultant program is native 68000 machine code and is executed as a job, meaning that it will, of course, multi-task.

It's a standard BCPL in most respects, thank God, but it also takes advantage of the floating point and graphics routines built into QDOS so that just about anything that can be done from BASIC or assembler can be done in BCPL. Procedures for setting the baud rate of the serial ports, and changing the screen mode are missing. However, all of these can easily be written (mainly in assembler).

The program shown is a simple serial file transfer system, used to send data to and from a BBC Micro (the disks are more secure!). It first reads in the standard header file, which declares all the global procedures and variables and defines a few constants ('Manifest' declarations in BCPL). All BCPL programs must have a procedure called START somewhere, as that is where execution begins. Our program declares a few variables and then proceeds to close the channels automatically opened for it. It then opens its own console device and selects it for both input and output.

The next few lines manipulate the screen and print a header. Then the user is asked whether he wishes to send or read a file. The conditional expression -> assigns a value of either TRUE or FALSE to **send.flag** depending on the key pressed. A filename is then collected and converted to a BCPL string, and if a file is being read in,

the end-of-file marker is collected.

Depending on the value of **send.flag**, one of two procedures is called; **SEND.FILE** or **READ.FILE**. Their functions are obvious, but notice that the sending of the file is a lot more efficient than the receiving, as we can send 'buffersful' of data at any time. We can't with **READ.FILE** because we need to check for the end-of-file marker.

That's BCPL. At £59.95 I would rate it as the most worthwhile of the bunch so far, but then I'm a BCPL convert.

LISP

Next up is LISP, again from Metacomco. This version of the language also costs £59.95 and comes with a manual describing the language and its implementation. It was written in the main by Dr Arthur Norman, who was responsible for the Acornsoft LISP interpreter on the BBC Micro. Consequently this version follows the same sort of path, with similar functions and methods of definition as the Acornsoft product. In fact, the Acornsoft publication *'LISP on the BBC Micro'* by Arthur Norman and Gillian Cattell, is probably the best book to buy as an accompaniment to the QL LISP package.

Naturally, this LISP takes advantage of the facilities available on the QL, coming with a full turtle graphics package. It also has most of the functions which were much missed on the BBC Micro.

LISP is an odd language which, like BCPL, is not really meant for number-crunching program development. It is, of course, fabled for its artificial intelligence ap-

BCPL

```
/*
This program copies files to or from the serial port (seric)
As there is no BAUD procedure in BCPL the current setting is used
As the program uses unbuffered keyboard input it must be run using EXEC_W
Written by Adam Denning Copyright (C) 1984 Adam Denning
Version 1.0 20th October 1984
*/

GET   "libhdr"

MANIFEST $(
          open.shared = 1
          open.new = 2
          max.len = 36
          buffer.size = 512
        $)

LET START() BE
$( LET send.flag,spare,temp,i = ?,?,?,?

   LET file.name = VEC 10
   LET pblock = TABLE 440,200,36,16          //window width 440
                                             //window height 200
                                             //window X pos 36
                                             //window Y pos 16
   ENDREAD() ; ENDWRITE()
   SYSIN := FINDINPUT("con_") ; SELECTINPUT(SYSIN)
   SYSOUT := SYSIN ; SELECTOUTPUT(SYSOUT)

   WINDOW(window.define,pblock,7,1)          //1 pixel white border
   SCREEN(screen.size,3,1)                   //set CSIZE 3,1
   SCREEN(screen.clear)                      //clear the screen

   WRITES("Serial File Transfer System")
   SCREEN(screen.size,0,0)                   //reset CSIZE 0,0

   SCREEN(screen.at,28,6)
   WRITES("By Adam Denning")

   SCREEN(screen.at,10,8)
   WRITES("Do you want to send a file or receive a file (S/R)?")

   send.flag := CAPITALCH(RDCH()) = 'S' -> TRUE,FALSE

   SCREEN(screen.at,10,10)
   WRITES("Filename: ")                      //collect filename for
   temp := READLINE(file.name,max.len+1)     //read / write
   TEST temp > 0 THEN
      temp := temp - 1
   ELSE
      temp := -temp

   FOR i = temp - 1 TO 0 BY -1 DO            //convert filename to string
      file.name%(i+1) := file.name%i
   file.name%0 := temp

   TEST send.flag THEN
      SEND.FILE(file.name)                   //send file to seric
   ELSE
      $(
         LET eof.marker = ?                  //get EOF marker
         WRITES("*NWhich character are you using as the end of file marker?")
         eof.marker := RDCH()
         READ.FILE(file.name,eof.marker)     //read file from seric
      $)

   SELECTOUTPUT(SYSOUT)

   WRITES("*N*NFile transferred*N")
$)

/*
SEND.FILE sends the named file to the serial port, using a buffer
allocated from the common heap. An implicit CTRL-Z EOF marker is sent
when the file has been sent.
*/

AND SEND.FILE(name.of.file) BE
$( LET out.stream,in.stream,in.bytes = ?,?,?
   LET buffer.vector = GETVEC(buffer.size/4)
   IF buffer.vector = 0 THEN STOP(err.om)

   in.stream := FINDINPUT(name.of.file)      //open file for input
   IF in.stream < 0 THEN
      $( WRITEF("*NCannot open %S*N",name.of.file)
         STOP(in.stream)
      $)

   out.stream := FINDOUTPUT("seric")         //open seric for output
   IF out.stream < 0 THEN
      $( CLOSE(in.stream)
         WRITEF("*NCannot open SERic*N")
         STOP(out.stream)
      $)

   WRITEF("*NSending %S...",name.of.file)

   SELECTINPUT(in.stream)
   SELECTOUTPUT(out.stream)

   $( in.bytes := READBYTES(buffer.vector,buffer.size)
      WRITEBYTES(buffer.vector,in.bytes)     //send file
   $) REPEATWHILE in.bytes > 0

   ENDREAD() ; ENDWRITE()
$)

/*
READ.FILE reads characters from the serial port (seric) and transfers
them to the named file until the specified end of file marker is met.
This end of file marker is not sent to the file.
*/

AND READ.FILE(name.of.file,marker) BE
$( LET out.stream,in.stream,in.byte = ?,?,?

   out.stream := FINDOUTPUT(name.of.file)    //open file for output
   IF out.stream < 0 THEN
      $( WRITEF("*NCannot open %S*N",name.of.file)
         STOP(out.stream)
      $)

   in.stream := FINDINPUT("seric")           //open seric for input
   IF in.stream < 0 THEN
      $( CLOSE(out.stream)
         DELETE(name.of.file)
         WRITEF("*NCannot open SERic*N")
         STOP(in.stream)
      $)

   WRITEF("*NReading %S...",name.of.file)

   SELECTINPUT(in.stream)
   SELECTOUTPUT(out.stream)

   in.byte := RDCH()
   WHILE in.byte NE marker DO                //read bytes until EOF
   $(
      WRCH(in.byte)
      in.byte := RDCH()
   $)

   ENDREAD() ; ENDWRITE()
$)
```

plications, but there's a whole lot more that LISP is capable of. A LISP program consists of function definitions, often using recursion, and the invocation of a function is the equivalent of running a program. Just about everything in LISP is an expression and can therefore return a value, but this value is not necessarily a number. LISP stands for LISt Processing, and a list is a series of items enclosed in parentheses (this is a very simple view, but it serves for description). LISP functions, like CAR and CDR, return various parts of a list as their values, and of course a list part can still be a list.

The function CONS creates lists out of more basic units, and in theory at least, every LISP program could be written in terms of these three functions. It would not be fun! As there is no clear definition of LISP, different implementations are often *very* different, and this version includes such loop constructs as WHILE and UNTIL. An interesting feature of the QL LISP, which wasn't present on the BBC version, is the ability to execute a source file and send its output to another file. This could be simulated on the BBC Micro using *EXEC and *SPOOL; though having one function to do it is nice.

Whether you should consider programming in LISP depends on where your interests lie. LISP is not as generally useful as say BCPL, but it can do some truly wonderful things. Our short example is the SORT function from Norman/Cattell (published with permission).

Pascal

One of the most popular languages at the moment is Pascal. Every variable must be associated with a type, such as CHAR, INTEGER or REAL. To assign from one type to another is generally an error, which means that the compiler can detect many more logical errors than an equivalent BCPL compiler, but it also means that the language is less flexible.

Pascal's sheer formality has caused it to be very popular amongst students and a surprisingly large number of systems programmers. It is not a language which one can ignore and its structured approach tends to make it very readable and easy to code. It is not capable of such low level interfacing as BCPL, but its ability to enumerate types and form complex data structures does make it more suitable for structured data processing and arithmetic, algorithmic problems.

Pascal programs can be written in such a way that functions and procedures are local to other functions and procedures, making the scope of temporary variables that much less, which obviously means that a student programmer will make fewer mistakes.

The Computer One Pascal compiler is a whole system, using a boot program to define a machine code procedure called PASCAL. Invocation of this procedure loads the system from Microdrive 1 and presents the user with a menu. Options from this menu allow program editing, compilation and running, along with a number of other useful utility options. The screen editor supplied is quite reasonable but is not in the same class as the Metacomco screen editor or GST's editor.

Source code is compiled into a compact interpretive code, similar to p-code or CINTCODE, which means that the object programs are small, but incapable of running outside of the system. The interpreter for this code is contained in the Pascal system, so it needs to be present to run a program. Nevertheless, the resultant programs are reasonably fast and able to take advantage of all the QL's facilities. This does seem to preclude the distribution of commercial software written in Pascal though, but no doubt Computer One is prepared to come to some arrangement here if necessary. Unfortunately, this Pascal is not really standard, so it is unlikely that anyone would consider writing applications in it anyway. When the full ISO compiler is available this will compile to pure 68000 code, so programs could be distributed without the need for any interpreter. The programs thus produced will also be standard QL programs, so could run with other programs.

Computer One Pascal costs £39.95, but is only suitable for users' own programs and not really intended for serious applications. Having said that, if Pascal is your language it will suit you down to the ground.

APL

APL is a high-level, interpreted language designed for non-computer professionals. It provides a concise and consistent notation which lends itself to writing ingeniously condensed programs.

Wherever possible, APL relegates the spadework to the computer leaving the programmer free to concentrate on how best to solve his problems using an arsenal of built-in mathematical functions.

APL provides extremely powerful, but simple array handling facilities. It dispenses with counts, loops and the like, allowing the programmer to manipulate com-

LISP

```
(defun make-node (val left right)
   (list left val right))

(defun left-subtree (tree)
   (car tree))

(defun right-subtree (tree)
   (caddr tree))

(defun item-in-node (tree)
   (cadr tree))

(defun print-tree (tree) (cond
   ((null tree) nil)
   (t (print-tree (left-subtree tree))
      (print (item-in-node tree))
      (print-tree (right-subtree tree)))))

(defun sort (item-list (tree))
   (loop
      (until (null item-list) (print-tree tree))
      (setq tree (add-item (car item-list) tree))
      (setq item-list (cdr item-list))))

(defun add-item (item tree) (cond
   ((null tree) (make-node item nil nil))
   ((orderp item (item-in-node tree))
   (put-in-left-subtree item tree))
(t (put-in-right-subtree item tree))))

(defun put-in-left-subtree (item tree)
   (make-node
      (item-in-node tree)
      (add-item item (left-subtree tree))
      (right-subtree tree)))

(defun put-in-right-subtree (item tree)
   (make-node
      (item-in-node tree)
      (left-subtree tree)
      (add-item item (right-subtree tree))))

(defun orderp (a b)
   (orderp1 (explode a) (explode b)))

(defun orderp1 (a1 b1) (cond
   ((null a1) t)
   ((null b1) nil)
   ((eq (car a1) (car b1))
      (orderp1 (cdr a1) (cdr b1)))
   (t (lessp (ordinal (car a1))
             (ordinal (car b1))))))
```

Pascal

```
PROGRAM Sieve;

CONST
   high = 2000;

VAR
   prime : ARRAY [1..high] OF BOOLEAN;
   divisor, next, i : INTEGER;

BEGIN
   divisor := 2;
   FOR i := 2 TO high DO
      prime[i] := TRUE;
   prime[1] := FALSE;
   WHILE SQR(divisor) <= high DO
   BEGIN
      next := SQR(divisor);
      WHILE next <= high DO
      BEGIN
         prime[next] := FALSE;
         next := next + divisor
      END;
      REPEAT divisor := divisor + 1 UNTIL prime[divisor]
   END;
   WRITELN('Primes up to ', high);
   FOR i := 1 TO high DO
   IF prime[i] THEN WRITE(i:8);
END.
```

plex data structures in the same fashion as he would single items. In this way matrix addition (which in other languages would require two loops and half a dozen statements) becomes, quite simply "A+B".

Typically, APL programs are made up of short, user-defined routines which may call each other or themselves. Each routine may be tested separately, prior to being tagged onto the body of a program. In this way errors are easily detected and programs modified.

Concise and easily debugged applications, written in APL, enjoy significantly faster development times than those written in other languages. Studies undertaken by the Data Control Corporation reveal that a 5000 line APL program, equivalent to some 20,000 lines in PL/1 and some 50,000 in COBOL, takes less than a third of the time to write.

APL, then, has traditionally been used as a kind of 'throwaway code', ideal for one-off 'what-if' analyses or for systems designed to evolve alongside the needs of their users. The emphasis here is on how fast a program can be written, rather than how fast it will run.

Whilst APL cannot compete with compiled languages in terms of the speed of execution, it is faster than some other interpreted languages. The reason for this is that APL stores all information used by an application in internal memory. This area of RAM is known as the 'workspace' and in it both data and functions rub shoulders. Workspaces alone can be saved and loaded, though items may be dropped from them to make room for others.

The concept of the 'workspace' coupled with an extensive command structure means that APL requires a relatively large amount of memory in which to run. In the past this has hindered the language's progress, making it available only on machines considerably more expensive than comparable systems running BASIC. However, because of advances in microprocessor capabilities and reductions in the cost of memory chips this is no longer the case.

Another reason why APL has made so little impact since its conception in 1962, is that the language makes use of a non-ASCII character set. APL statements resemble a mysterious shorthand made up of Greek letters and overstrike characters. This has proved particularly off-putting to users familiar with other languages (including English!) In addition, it has meant that APL requires the use of dedicated keyboards and printers thereby increasing cost of implementations.

Addressing themselves directly to this problem, MicroAPL Limited, at the request of Sinclair Research, will be bringing out a keyword version of APL alongside their standard version. Both are expected to cost in the region of £90, occupy 95K of memory and permit the use of a 28K workspace (which may be increased to 140K using a 128K RAM extension).

It should be noted that some of the more advanced APL.68000 features have not been implemented and also that multi-user filing features have been removed. These may well be available as extensions at a later time. Meanwhile the various extensions to take account of the QL's special features such as colour, windows, redirection of output, sound and graphics, more than make up for the loss. Finally, with the novice foremost in mind, the keyword IF similar to the APL '/' operator, but with the arguments reversed, has been added to simplify operations.

APL Keywords

APL CHAR.	KEYWORD						
==========		↑	UP	∨	OR	⊤	ENCODE
		⌽	ROTATE	∧	AND	○	TRIG
		⍉	TRANS	≠	NE	*	EXP
		⍟	LOG	/	ON	?	RAND
		⍒	LOCK	\	OVER	⍴	SIZE
		⍎	EXEC	⍺	PICFMT	⌈	MAX
⊞	FHOLD	⍕	FMT	⊥	DECODE	~	NOT
⊟	FDROP	⍞	PROMPT	⌊	MIN	↓	DROP
⊡	FGET	⌹	MATDIV	∈	IN	⍵	IF
⊡	FPUT	<	LT	∇	DEFN	↑	TAKE
⍱	NOR	≤	LE	⍳	INDEX	←	IS
⍲	NAND	=	EQ	∘	OUTER	→	GOTO
↓	DOWN	>	GT	|	ABS	≥	GE

APL Functions

Examples of simple functions written in keyword APL

```
      defn R is A CENTRE B
<1>   ! Centres text on page
<2>   ! A is page width, B is text as vector or matrix
<3>    A is min 0.5*A - -1 take size B is (-2 take 1 1,size B) size B
<4>    R is (((1 take size B),A) size' '),B
      defn

      defn R is A UNDERLINE B
<1>   ! Underlines a vector
<2>   ! A is underline character, B is vector
<3>    R is (2,size,B) size B,(size ,B) size A
      defn

      defn R is A NPV B;I
<1>   ! Calculates net present value of monthly flows B at rate A percent
<2>    I is index size B
<3>    R is +on B/((1+A/100) exp 1/12) exp I
      defn
```

RELEASE THE TRUE POWER OF THE QL!

Write programs to unlock the multi-tasking power of your QL!

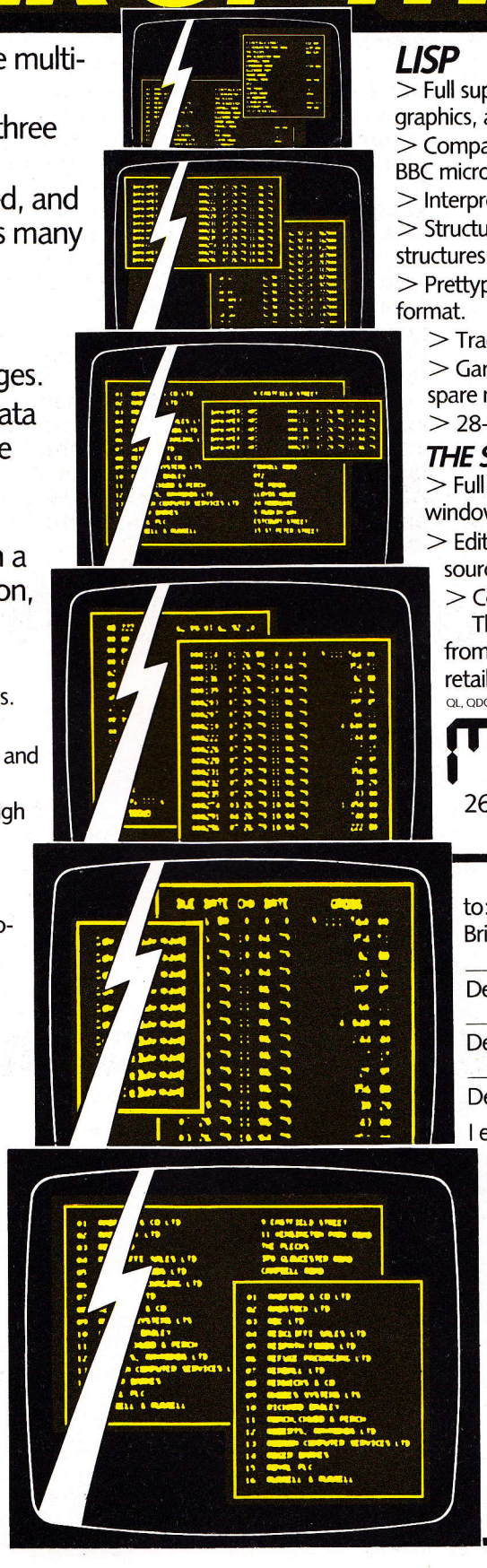
Metacomco now offers you three languages for the QL:

> Use **ASSEMBLER** for its speed, and for complete access to the QL's many features.

> Use **BCPL** for systems programming: writing games, utilities, and applications packages.

> Use **LISP** for manipulating data structures, and for exploring the world of artificial intelligence.

Each language comes as a complete development kit with a screen editor and documentation, for only £59.95.



THE ASSEMBLER

> Standard Motorola 68000 mnemonics.
> English error messages.
> Produces code which can be EXECed, and run as a concurrent job.
> External references allow linkage to high level languages and other assembler modules.
> Macro expansions.
> Position independent, absolute or relocatable code can be produced.
> Conditional assembly.
> Large range of directives.
> Fully formatted listings.
> 30 character variable names and 32-bit values.

BCPL

> True compiler.
> Multiple BCPL programs can run concurrently.
> Full runtime library includes interfaces to QDOS graphics, window handling, file operations, etc.
> Exception handling provides diagnostics for debugging.
> Link loader allows linkage of separately compiled segments.
> Assembler and BCPL modules can be linked together.
> 32-bit variables use the full QL address space.

LISP

> Full support of QL features including windows, graphics, and screen handling.
> Compatible with Acornsoft Lisp for the BBC micro.
> Interpreter.
> Structure editor allows alteration of data structures.
> Prettyprinter displays programs in structured format.
> Tracer to aid in debugging.
> Garbage collector automatically recovers spare memory space.
> 28-bit integers and 250-character names.

THE SCREEN EDITOR

> Full screen editor within a user defined window.
> Edits any ASCII file including SuperBASIC source.
> Comprehensive range of commands.

The Development Kits are available **now** from HMV, Menzies and other leading retailers, or direct from Metacomco.

QL, QDOS, and SuperBASIC are the trade marks of Sinclair Research Limited.

METACOMCO

26 Portland Square, Bristol BS2 8RZ.
Tel: Bristol (0272) 428781

Phone today, or post this coupon to: Metacomco, 26, Portland Square, Bristol BS2 8RZ. Please send me:

_____ copies of the QL Assembler Development Kit
_____ copies of the QL BCPL Development Kit
_____ copies of the QL Lisp Development Kit at £59.95 each.

I enclose a cheque for £_____
or please debit my ACCESS/VISA Account No.

Card expiry date _____

SIGNATURE _____

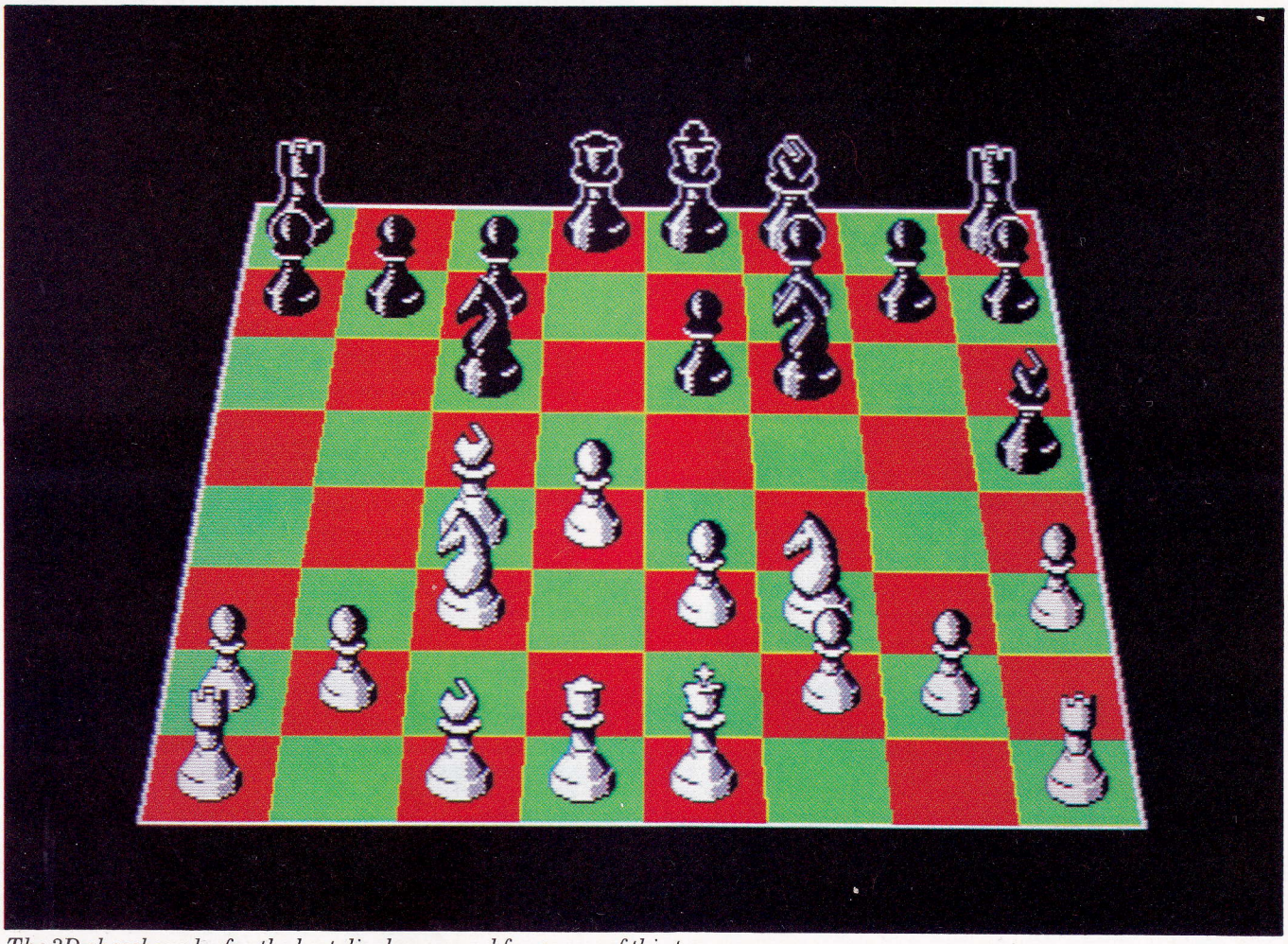
NAME _____

ADDRESS _____

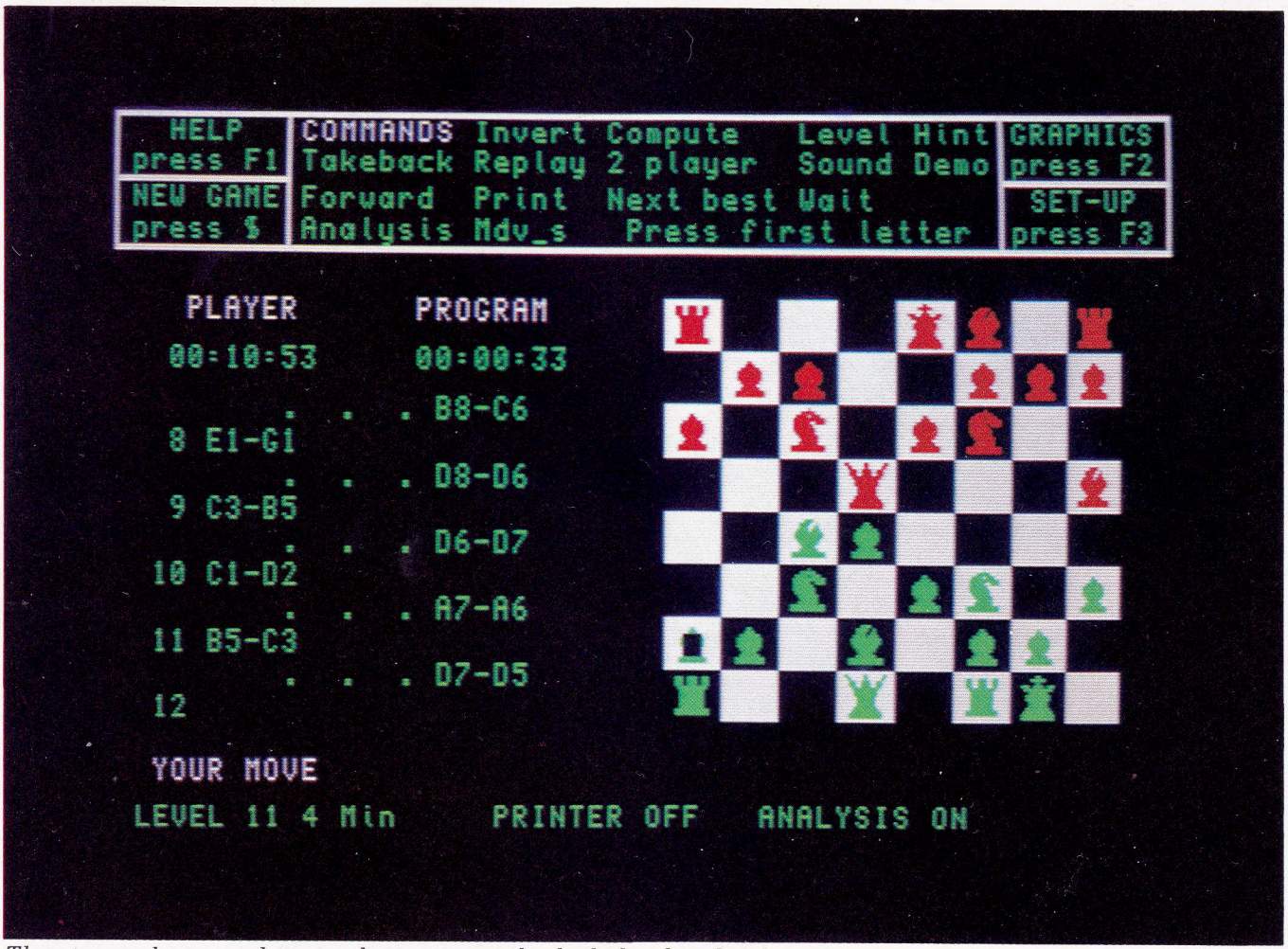
POSTCODE _____

TEL NO _____

Price includes VAT, postage and packing UK mainland only. Delivery allow 28 days. QL2



The 3D chessboard – far the best display around for games of this type.



The set-up and command screen where moves can be checked and analysed.

Psion Chess

One of the first high quality games to emerge for the QL had to be Chess – David Creasey checks *it out.*

Signs that the QL is at last coming of age lie in the software that's starting to appear. QL Chess is one such arrival, and significant because it is an example of superb programming that sets a very high standard for the market. It works so well that the user (player?) quickly becomes totally absorbed in the business of playing or analysing the game of chess.

Psion's QL Chess package consists of a single master microdrive cassette and a straightforward instruction leaflet. The instructions explain only the program's many features, they don't cover the basic moves and rules of chess. The information on the instruction leaflet is also summarised on three help screens which can be called up at any time when playing the game.

Psion recommend that the master cassette be backed up immediately and that the copy is used when running Chess. However, as the back-up copy won't run on its own – it needs the Master tape resident in drive two – it seems there's little point in making a back-up in the first place!

The Third Dimension

QL Chess has two main playing screens – a two-dimensional plan type board and a striking, three-dimensional graphics board.

The 2-D chess board is displayed on the right hand side of the screen when the program is booted. The various feature options are shown in a panel menu at the top of the screen, and information about the state of the game (such as whose turn it is), at the bottom. Two columns on the left hand side display the moves, in standard chess notation, as they're made on the 2-D board.

On booting, the board is set for a new game. You can start playing immediately, load an earlier game, or set up the board for problem solving or to simulate game positions from books.

During a game you can save a position, start a new game (in which case the pieces on the screen board are automatically reset and the clocks zeroed) or choose to interrupt a game and alter piece positions.

Moving pieces is simple. Using either the cursor arrow keys, or a joystick, move the flat-line cursor on the 2-D board to cover the relevant chess piece. Press Enter (or the spacebar or fire button), move the cursor to the target square and press enter. If it is a legal move it is carried out instantly on the screen board. At the same time the move is recorded under the Player column on the left of the screen.

Other features allow you to change level (this determines the length of time the program takes to calculate its own move), to take back an unlimited number of moves (either your own or the program's) or to replay a game from the beginning (in which case you can also interrupt the replay to resume it as a 'live' game).

If stuck for a good move, press H for hint. The computer flashes a recommended move on the screen board. You can also force the program to make a move or else to take back its latest move and, using the powerful Analyse option, get a taste of how the program is 'thinking ahead'.

When Analysis is selected the program displays a line of information at the bottom of the screen. The first figure indicates the computer's current advantage in the game judged by material and positional strengths, in terms of pawn values (eg, +3). Next is a record of the latest move made, followed by the program's prediction of the next moves ahead (up to seven).

It's interesting watching the computer play itself in demonstration mode; although the game moves so quickly that you have to pause it (using the Wait option) for proper study. Selecting the 2-player option allows two 'people' to play each other, using the QL screen as a board while the program records the moves and offers analysis.

You don't ever need to leave this 2-D mode. It incorporates everything necessary to keep chess lovers (beginners included) totally addicted and in itself 2-D QL Chess would be a prime contender for the best chess program available on any computer. However, Psion have also included a striking three dimensional 'animated' colour display of the board and the game pieces currently in play. It is a pleasure playing in this mode. To move or set up pieces use either the cursor control arrows or a joystick to select the piece to be moved and its destination square. In this case the on-screen cursor consists of the corners of a square – it does not obliterate the piece it covers. If it is a legal move, the piece glides into position very quickly. You can also alter piece positions during a 3-D game using the Set Up option (which generates a set-up cursor consisting of two small squares).

Modes And Options

Most of the QL Chess features available in 2-D mode (including hints for next best move) can be used in 3-D – although no messages are printed on-screen so you have to toggle back to 2-D mode (key F2) for analysis or a list of the moves made so far.

Also useful if you want to study games and improve your play are the printing options. Switch the printer on at the beginning of a game and each move will be printed out as it's made. A simpler solution is to wait until the end of a game, switch on the printer and then select the Replay option. CTRL-P produces a print out of the 2-D board but with no graphics (the pieces are represented by letters eg, K, Q, K, B etc).

Playing The Game

Psion claim that top club players will find the chess challenging and also point to the intriguing possibilities with the top levels of play. In level 12 the program takes approximately the same average time per move as its opponent while level 13 – infinite – leaves the program continuously calculating its best possible move until you, the poor opponent, forces it to play with the Move-now command (the chess algorithms, incidentally, were developed by Richard Lang).

At the bottom end of the playing scale, Easy novice level, the program only 'thinks' when it is its turn to do so (not while you are considering your own move) and, if it decides it is winning, it will start playing its second best possible move each turn, to give the beginner a chance to catch up!

In all there are 14 ordinary playing levels, 14 easy and eight problem modes (to solve problems up to Mate in eight moves).

You can learn a lot watching a demonstration game – you can 'cheat a lot' by going back moves (to the beginning if necessary), by forcing the computer to play your turn or by adding and removing pieces at random.

QL Chess will succeed because it is very easy to use, a pleasure to play and because of the powerful control it gives anyone over the play of a game.

The Super B

Adam Denning concludes his novel introduction to Sinclair's extended BASIC, with a round-up of lesser-known operators and features.

Most versions of BASIC include a group of symbols called 'logical operators'. SuperBasic has two varieties of each item in this group, known as the logical and the bitwise form. The logical form is the most used as it allows the formation of expressions which yield a logical value depending on certain conditions. A logical expression has one of two values – true or false. The QL and most other machines treat zero as being false and any other value as being true.

A relational expression is one which compares two or more items and returns a logical value depending on the result of the comparison. For example, if we had a variable called **a** and wanted to know if it was equal to or above a certain limit, we would use a relational expression:

If a >= limit THEN . . .

The **a >= limit** part is the expression, and this yields a value which the IF acts on. If **a** is greater than or equal to limit then the statement(s) after the THEN are executed.

Often we need to decide whether a number of items are true, such as deciding whether a certain character is an ASCII digit or not:

IF char >= '0' AND char <= '9' THEN . . .

In this example the statement(s) after THEN are executed only if char is greater than or equal to ASCII 0 AND less than or equal to ASCII 9; in other words, if it is an ASCII digit. Conversely we could test for the opposite condition, that is, if the character is NOT a valid digit. One way would be to write:

IF NOT (char >= '0' AND char <= '9') THEN . . .

Here the NOT inverts the value of the earlier conditional expression, giving us false if the character is a digit and true if it is not. This isn't particularly elegant, so we would normally use another logical operator to form the expression thus:

IF char < '0' OR char > '9' THEN . . .

Here we test for the character being less than ASCII 0 or greater than ASCII 9. If either of these conditions is satisfied then the character is not a digit.

The final logical operator provided by SuperBasic is XOR, which stands for 'exclusive or'. This is very rarely used in BASIC, but it is occasionally valuable. For example, if we needed to do something if one of two values was true, but not if both were true:

eg, IF char < '0' XOR char < CHR$(32) THEN . . .

Here we do the THEN part if the character is less than ASCII zero but greater than ASCII space (32). This circumstance is obviously rare, but it is there if we need it.

From these four logical operators (NOT, AND, OR and XOR) we can write what are known as truth tables which tell us what value to expect as a result, depending on the 'inputs' (see below).

The bitwise versions of these operations seem to have very different actions, but in fact they are closely linked. First of all, they can only have integer parameters between −32768 and 32767. They treat each number as being a binary string of 1s and 0s, taking 1 as true and 0 as false. The aforementioned truth tables are then applied to each pair of bits and the resulting value is put into the corresponding bit position of the result. For example, if we take two numbers, say 34 and 123, and perform a bitwise operation on them, we get the following results:

34 && 123 = 34 (bitwise AND)
34 !! 123 = 123 (bitwise OR)

NOT

Input:	Output
FALSE	TRUE
TRUE	FALSE

NOT simply reverses its input

AND

Input 1:	Input 2	Output
FALSE	FALSE	FALSE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
TRUE	TRUE	TRUE

AND will only give true if BOTH its inputs are true.

asic Special

34 ^^ 123 = 89 (bitwise XOR)

What is happening here? Well, let's take a look at the binary representations of 34 and 123. 34 = 0100010 in binary, and 123 = 1111011. Now, if we take each bit in turn from each number and apply the truth tables to them, we get the following results:

0100010 && 1111011 = 0100010
(= 34 in decimal)
0100010 !! 1111011 = 1111011
(= 123 in decimal)
0100010 ^^ 1111011 = 1011001
(= 89 in decimal)

It's obvious now, isn't it! The final logical bitwise operator is ~~ (NOT), which like its counterpart only works on one operand. It takes a number, negates it and subtracts one, so ~~123 is −124 and ~~34 is −35. Why does it do that? An integer never occupies more than 16 bits on the QL, as that is the defined SuperBasic integer size. Bitwise NOT takes each number as a binary string of 16 digits, so that 34 is 0000000000100010 and 123 is 0000000001111011, and inverts each bit. 34 then becomes 1111111111011101 and 123 becomes 1111111110000100. Converting these into decimal we get the results we've already shown. This is called two's complement form, and is really only used in machine code.

The bitwise forms of the logical operators are useful in BASIC when we want to 'mask out' certain parts of numbers. For example we could convert a number into hex by isolating four bits (a nybble) of the number at a time and converting that into a hex digit, like so:

```
10000 DEFine FuNction
      HEX$(number)
10010 LOCal a,b,num,an$
10020 num = number:an$ = ''
10030 FOR a = 1 TO 4
10040 b = num && 15:num =
      num DIV 16
10050 an$ =
      '0123456789ABCDEF'
      (b+1) & an$
10060 END FOR a
10070 RETurn an$
10080 END DEFine
```

This function takes a number between −32768 and 32767 and converts it into a hexadecimal string by isolating the lowest nybble and converting it into the corresponding digit. The number is then divided by 16 and the operation repeats for all four digits.

Integer mathematics is in fact quite useful on the QL, although it is limited by the fact that to fit into a SuperBasic integer variable, or to be operated on by an integer operator, a value must lie between −32768 and 32767. The 68000 has a data size twice as wide as this, which means that in theory QL integer arithmetic could have been written to use numbers between −2147483648 and 2147483647. The integer operators provided by SuperBasic are:

DIV this divides one number by another to give a whole number result.

MOD this divides one number by another and gives the whole number remainder.

~~ bitwise NOT
&& bitwise AND
!! bitwise OR
^^ bitwise XOR

Of course all the other operators such as multiplication and raising to the power may be used on integers, but the result is not guaranteed to be integral. If the result of an integer expression is too large to fit into an integer or if an operand of an integer operator is too large, the error message 'Overflow' occurs and the program stops.

Notice that the INT function,

OR

	Input 2	Output
Input 1: FALSE	FALSE	FALSE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
TRUE	TRUE	TRUE

OR gives true if either or both of its inputs is true

XOR

	Input 2	Output
Input 1: FALSE	FALSE	FALSE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
TRUE	TRUE	FALSE

XOR gives true if either of its inputs are true but not if they both are

which returns the whole part of a number, is not one of the integer mathematics range as it can work on much larger numbers. Machines normally supply integer operations because an integer requires less space in memory than a floating point number (and the operations are generally faster). This is not always the case with the QL's SuperBasic intepreter, unfortunately!

Many people have complained that although a large range of numbers can be held in the QL from 10^{-615} to 10^{615}, they are not held very accurately. Each floating point number is stored in the QL in six bytes, the first two forming the exponent and the next four holding the mantissa. The number that these six bytes represents is calculated from the following formula:

number = mantissa * (2^(exponent – 2079))

The exponent, in fact, only uses the bottom three nybbles of the two bytes allocated to it. If we look at a number held in this format, we'll see why they aren't held that accurately. To make things easier to read we'll show the number's internal representation in hexadecimal:

803 1000 0000

Applying our formula to this, we find that this number is equal to 1, and so are

802 2000 0000

and

801 4000 0000

and so on. All these representations equate to exactly one, but the smallest alteration that we can make to any of them is to increase the mantissa by one:

803 1000 0001
802 2000 0001
801 4000 0001

By doing this we get the numbers

1.000000004
1.000000002
1.000000001

Obviously, as the final value shows the smallest change, it must be the most accurate. So, that is the way in which we'd choose to represent a number – with the lowest exponent and therefore the highest mantissa. But it also shows that the smallest change we can make to a floating point number is one ten-millionth of a percent. This may seem terribly close to as accurate as we need, but some would say that it isn't close enough for scientific or accounting purposes; especially as the relative inaccuracy is compounded by each operation carried out on a number.

In much the same way that certain fractions cannot be accurately represented in decimal (a third, for instance), many numbers cannot be accurately expressed in this binary notation. Most of the time this is of no consequence to us, and certainly most programmers would choose to ignore it.

Last month's 'Adding to SuperBasic' got rather garbled during printing, and most of the machine code program was out of place! We apologise to those who typed it in and found that nothing went right. The corrections are as follows:

Page 55: The comment in the code starting 'A procedure to initialise the ...' is in the right place, but the code isn't. The code that should be here is on page 56 starting at the SETABLE label, down to EXIT_TAB. This should immediately be followed by the section of code on page 57 starting with the comment 'A procedure to show the current ...' down to the SPACE label on the same page. Below this comes the code from DIGITS on page 55 down to the comment starting 'Buffer for long integer to ...' on page 56.

Next comes the code starting at STRING on page 57 down to BUFFER on the same page, followed by the code from LINE_FEED to the RTS after NAMEPT on page 56.

All on page 58 is correct except that BUR_LEN after INPBUF at the bottom should be BUF_LEN (Phew!).

Microdrive Microdrive Microdrive Microdrive Microdrive

NEW

ZX Spectrum and QL Microdrive Storage Box £5.95

- ★HOLDS 20 CARTRIDGES
- ★FULLY INTERLOCKING
- ★CENTRE SECTION FOR INDEX CARDS
- ★DESIGNED TO MATCH SPECTRUM AND QL

Transform Ltd has now produced a smart new Storage Box for the ZX Spectrum and QL

The Transform Microdrive **Storage Box** is attractively designed to match both the Spectrum and QL

It will hold 20 Microdrive cartridges (enough to store **1.6 megabytes** of data!)

The Microdrive Storage Box is fully interlocking and will allow you to expand your system with all the new programmes on microdrive as well as enhance the 'smart' efficient look of your office/study.

Simply write or telephone:

TRANSFORM LTD.

DEPT X, 'SWATLANDS', LUCKS LANE
PADDOCK WOOD, KENT TN12 6QL 089 283 4783

For fast delivery quote your Credit Card/Access Barclaycard number

But before you leap at the likely cost of all this, just take a look at the price details in the order coupon. Because in the same way that the QL gives you so much for so little, the cost of communicating now takes a downward jump of the imagination.

nmunications.

Q MOD
V23 modem. Direct telephone line connection, 1200/75 baud full duplex operation for Prestel and other database access, 1200/1200 baud half-duplex operation for user-to-user communication.

Q CALL
Auto dial/answer unit for attended or unattended operation.

Q CON
Asynchronous multi-speed interface, single chip micro based with software on microdrive cartridge. Operates at 9600, 4800, 2400, 1200, 300 and 75 baud. Independent Tx and Rx rates. 25-way D-type, SER 2, power and expansion connectors. Provides VT100 emulation and software support for Q MOD and Q CALL. Can be used with any modem.

Tick	Quantity	Module	Price*	Value
☐	_____	Q CON	£79.95	_____
☐	_____	Q MOD	£79.95	_____
☐	_____	Q CALL	£49.95	_____
		Total order value		_____

☐ I enclose a cheque/P.O.† made payable to O.E. Ltd.
☐ Please charge my Visa/Access† card
Card No. ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐
Signed _____
☐ Please send me more information.
Name _____
Address _____
Tel: _____

oel

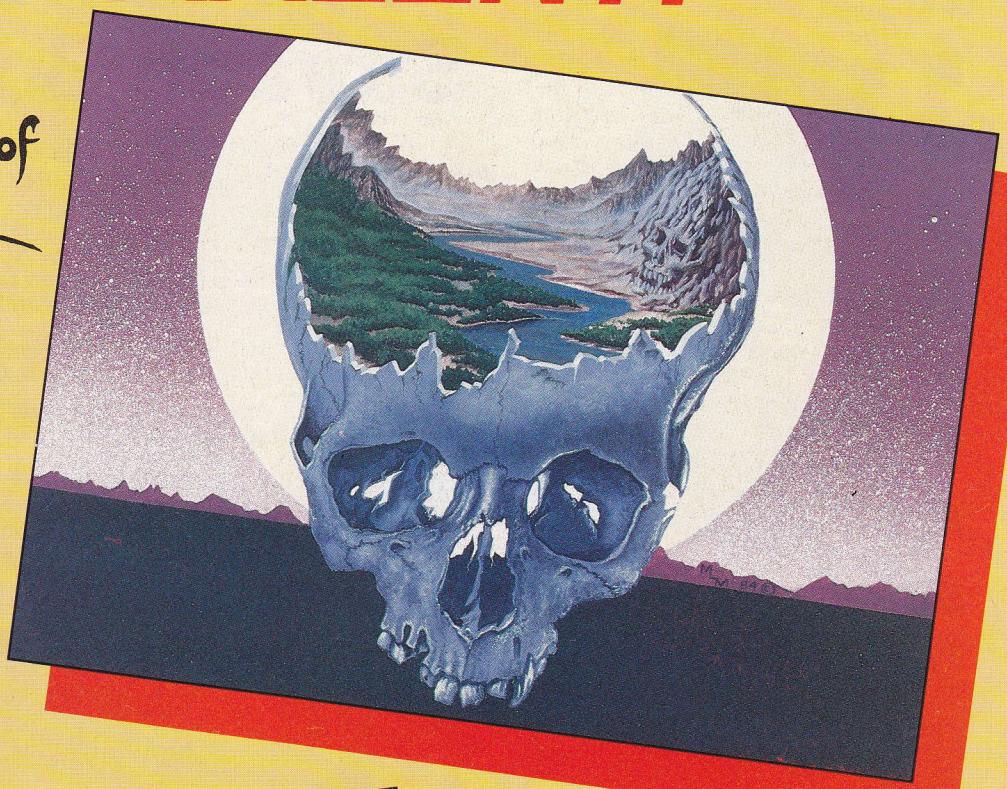
O.E. Ltd, North Point, Gilwilly Industrial Estate, Penrith, Cumbria CA11 9BN. Tel: 0768 66748 Telex: 64157

QL2

*Inc. VAT and P&P. Please allow 28 days for delivery †Please delete as appropriate.

The first QL adventures from TALENT!

the Lost Kingdom of ZKUL



Explore the ancient dwelling-place of the Dwarves – where the Wizard guards the last precious secret.

A classic and complex text adventure with hundreds of locations and a huge vocabulary. Set in real-time with traps, tasks, puzzles and mazes – and a special note-pad feature to aid you.

"the quality of the game is superb"
Micro Adventurer

PRICE
£19.95
+ 50p postage and packing

WEST

You are on the track of a notorious gang of bank robbers who have gone to ground near an abandoned mining town deep in Indian territory. Your task – to outwit and outgun the robbers, collect as much loot as possible and piece together clues on how to escape.

A demanding and exciting text adventure for the experienced games-player, using over 200 words and as many phrases. There are over 130 locations to search – not all of them easy to find. Events happen in real-time outside your control – Indians charge, rattle-snakes slither past and robbers appear and shoot at you. How long can you survive?

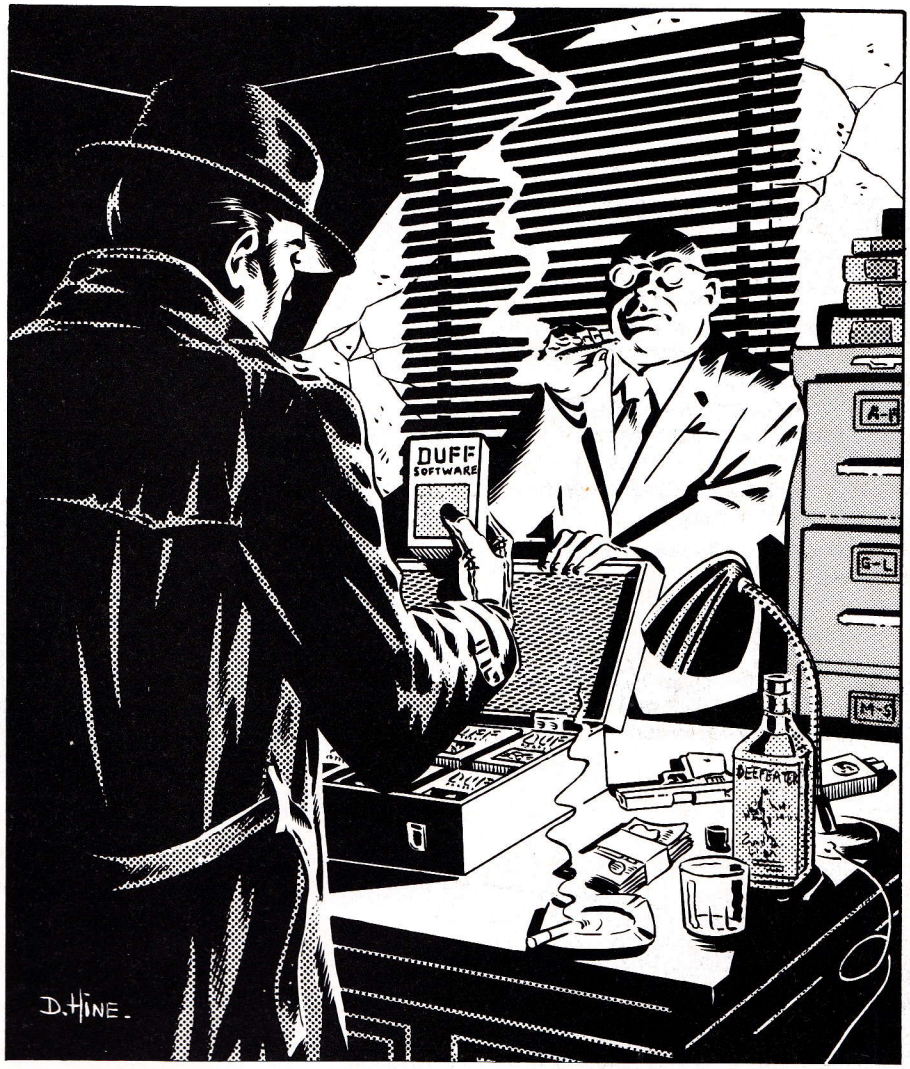
PRICE
£19.95
+ 50p postage and packing
(Also on CBM 64, BBC & Electron)

TALENT
COMPUTER SYSTEMS

CURRAN BUILDING, 101 ST JAMES ROAD, GLASGOW G4 0NS
24 HOUR CREDIT CARD HOT-LINE 041-552 2128

Games of Chance

Buying games software for the Sinclair QL is currently a risky business; but then the intrepid William John likes to live dangerously . . .



Games programmers' first reaction to the QL's 128K RAM and 68008 processor has been to revert back to the standards of the unexpanded ZX81. In short, games software for the QL is scarce and (often) pretty awful.

First there's an outfit called Quantum Soft, which supplies a microdrive cartridge containing one game for £9.95 or two for £14. Since the games are called *Hangman* and *Mastercode* (a Mastermind spin-off), and since neither stretches to more than 200 lines of BASIC, you'll see why Quantum Soft's products irresistably recall the days of membrane keyboards (plus ça change!) and wobbling RAM packs. They're outrageously over-priced and suit the QL like a nodding dog suits a Lamborghini *(unfair to nodding dogs! Ed)*. We suspect that Quantum Soft is run by an eager adolescent from his back-bedroom – otherwise our reaction would be indignation rather than sympathetic embarrassment. To be avoided like the plague.

Rodent Software is another company supplying 4K games for a 128K machine, but their selection is reasonably big and reasonably priced. A cartridge of seven programs costs £8.50, or you can have them copied onto your own cartridge for £3.50.

It's pretty traditional stuff, and (again) bears all the signs of a programmer who's graduated to the QL from a more modest Sinclair computer.

Fruit Machine is a respectable one-arm-bandit simulator, with a couple of interesting extra features. At this level of software quality it's probably futile to start wondering if there's any point in entirely random computer games. So we won't.

Mushroomaze is awful, *Basic Chess* and *Eliza* are (on our copy) bug-ridden, but *Mars Lander*, *Laserace* and *Orbiter* are mildly diverting. All the Rodent material is of the elementary kind you might come across in books of BASIC listings called something like 'Fifty Games for your ZX81'. However, if £3.50 seems a small price for keeping the kids quiet for an hour or two, Rodent are contactable at 3 Brook End Drive, Henley, Warwickshire.

Apart from the splendid *Psion Chess* (of which more elsewhere in this issue), the only decent QL games we've come across are a couple of adventures soon to be released by Talent Computer Systems. Their *Zkul* and *West* are text-only adventures, full of crossroads, assorted scattered objects and lots of sudden death – great fun if, like us, you care for that sort of thing.

West places you exactly where you'd expect, surrounded by vultures, tumbleweed and 'injuns'. Such a locale has great advantages in a text-only adventure, since we all carry in our heads a much more comprehensive library of images for cowboy country than for any sword and sorcery setting.

The basics of the adventure are well-regulated. The objects are scattered before every new game, but – if you're killed – sensibly left by the body where your reincarnated self can find them again. The locations are numerous and subject to random visitations from tumbleweed, rattlers, bad-tempered bank robbers and your trusty but fickle steed.

Our pre-production version of the game was innocent of all documentation, so our ramble amongst the clap-board buildings and trackless deserts was aimless if highly entertaining. However, if reviewer addiction is anything to go by (and it should be) then *West* will please, not just adventure addicts, but anyone who ever frowned into a puzzle of any sort.

From the same company comes the more traditionally-set *Zkul*, which sends you out into a landscape of wizards, dungeons and axe-wielding dwarves in pursuit of hidden treasure.

Zkul has a bewildering number of settings, some vexing logic tests, a full cast, and an entirely satisfactory store of treasures. It also boasts (in our pre-production version) a fair number of bugs, though we trust that these latter features will be removed by the time the program hits the streets.

West and *Zkul* cost £19.95 each, are due for release in December, and can be ordered from Talent Computer Systems on 041-552 2128.

MONITOR/PRINTER PACKAGES

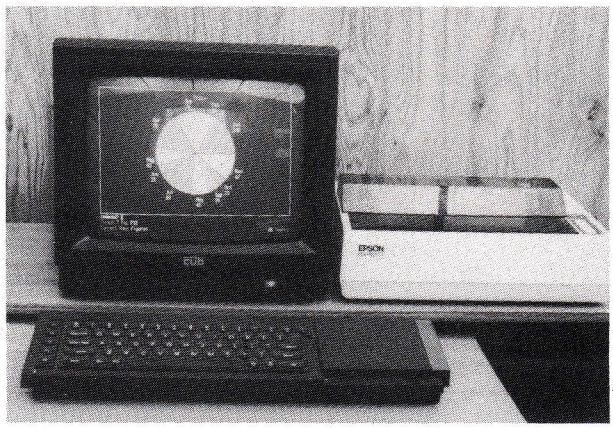
FOR

SINCLAIR QL

- **NO INTERFACING PROBLEMS. JUST PLUG IN AND GO!**
- **ALL SYSTEMS FACTORY-TESTED BEFORE DESPATCH TO YOU!**

PACKAGE DEALS

MONITOR	*with Printer*	*Package Price*
	Brother HR5 + mains adaptor	£425
Microvitec 14" colour 1451DQ3	Epson RX80 F/T	£585
	Epson FX80	£725
	Epson FX100	£869
	Epson DX100	£745
	Brother HR15	£715
	Canon PW1080A NLQ	£649
	Epson JX80 colour	£895
	Canon PJ1080A colour	£835



(Illustrated: QL with Microvitec 1451DQ3 and Epson RX80F/T)

Epson parallel printers supplied with RS232C I/F fitted Canon printers supplied with Miracle Systems I/F. All monitor and printer leads included.
Prices are inclusive of VAT. Add £12 for delivery.

PRINTERS		*ACCESSORIES*		*MONITORS*	
Dot matrix:		*Tractor feeders:*		*Colour:*	
Epson RX80	£260	FX80	£35	Microvitec	
RX80F/T	£290	LQ1500	£60	1431 Std. res.	£199
FX80	£425	DX100	£90	1431 MZ Spectrum	£235
FX100	£620	HR1	£95	1451 Med. res.	£299
LQ1500	£1,125	HR15	£90	1451 DQ. QL	£275
JX80 colour	£625	HR25/35	£99	1441 High res.	£506
Canon PW1080A NLQ	£375	*Cut-sheet feeders:*		1431 PAL/Audio	£259
PJ1080A colour	£529	DX100	£250	1451 PAL/Audio	£375
Daisy Wheel:		HR15	£250	*Monochrome:*	
Epson DX100	£495	HR25/35	£250	Philips	
Brother HR1	£675	*Keyboards:*		BM7502 green	£92
HR15	£450	DX100	£170	BM7522 amber	£92
HR25	£795	HR15	£170	Sanyo	
HR35	£995	RS232C I/F	£30	DM2112	£90
		RS232C 2K buffer	£65	DM8112	£125
				Novex 12/800	£120

Prices are inclusive of VAT

For Printers/Monitors add £10 for express delivery and for Accessories add £5 for express delivery

ZEAL MARKETING LIMITED

VANGUARD TRADING ESTATE, STORFORTH LANE, CHESTERFIELD S40 2TZ
Tel: 0246 208555 Telex: 547697

Have Fun With Alternative Graphics

Alec Short explains how to use the QL to draw directly onto the screen.

The QL provides some excellent graphic facilities via the normal SuperBasic input, but there may be times when it is preferable to draw directly onto the screen by an alternative method.

The QL manual provides details of the computer's memory map and some guide to applying the screen RAM which provides for direct access to the screen.

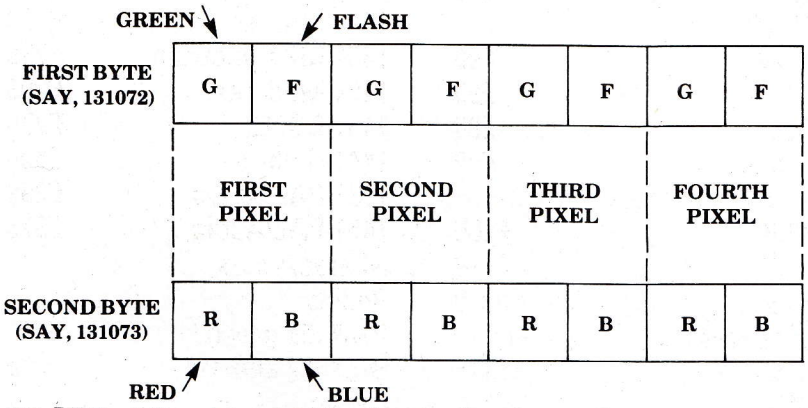
Let's examine how it works:

The screen picture is controlled by 32768 bytes of memory stored between addresses 131072 (20000 hex) and 163847 (28000 hex). This consists of 256 picture lines consisting of 128 bytes of data each. The bytes are used in pairs, and each pair controls four screen pixels. The first two bytes at addresses 131072 and 131073 control the first four pixels at the top left hand corner of the screen and the next two bytes the next four pixels to the right, etc (not right to left as your user manual tells you!). The 128 bytes controlling the top line, therefore, provide (128/2)*4 which equals 256 pixels across the screen. The next 128 bytes control the next line down in the same way until the bottom right-hand corner of the screen is reached. This enables any particular point on the screen to be readily located from the memory.

The pair of bytes controlling particular sets of four pixels control not only the location of the pixels but also their colour and whether or not they are flashing.

Experimenting can produce some fancy effects.

To understand how this happens it is necessary to consider the numbers stored at a pair of addresses in their binary state: each of the two numbers will consist of a set of eight binary digits, the first two digits from each of the pair of addresses is used to control the first pixel. The second two binary digits from each address controls the second pixel, the third two control the third pixel and so on. I have tried to illustrate this more clearly in the diagram below:



In the diagram the top row represents the eight bits of the first byte and the bottom row the eight bits of the second byte. The first two bits of the first byte determine whether the first pixel contains green and whether or not it is flashing. The first two bits of the second byte determine whether the pixel contains red or blue. In each case, if the digit is one, then the factor will be present, if the digit is zero then that factor will be absent from the pixel. If all of the four digits are zero, the pixel will be black (no colour) and if they are all one, the colour will be white and the flash turned on.

The second pixel is controlled in the same way by the third and fourth bit in each byte, the third pixel by the fifth and sixth bits and so on. Once the control process is understood it is easy to control the screen. Try colouring the screen with the following routine:

```
110 CLS
110 INPUT 'Enter two
    numbers, each between 0
    and 255',a,b
120 FOR n=0 TO 32768 STEP 2
130 POKE 131072+n,a : POKE
    131073 +n,b
140 END FOR n
```

Some good combinations of numbers are 170,0 or 0,170 or 0,85 or 0,255 or 170,170 or 170,85. Experimenting with other pairs can produce some more fancy effects.

To conclude here's a little program which will draw a black 'airplane' sprite against the normal red background, roughly at the centre of the screen. By poking this in with SuperBasic we have a system that would be rather slow for many games – machine-code entry would be preferable but it may give some ideas on how to extend personal programming as it can be moved by deletion with '0,170's and reprinted with incremented addresses (1's for horizontal, 128's for vertical).

```
100 REMark To print an 'airplane' sprite directly onto the screen
110 CLS
120 RESTORE
130 FOR n=0 TO 15
140 READ a,b,c,d
150 POKE (147500+(n*128)),a : POKE (1+(147500+(n*128))),b
160 POKE (2+(147500+(n*128))),c : POKE (3+(147500+(n*128))),d
170 READ e,f,g,h
180 POKE (4+(147500+(n*128))),e: POKE (5+(147500+(n*128))),f
190 POKE (6+(147500+(n*128))),g: POKE (7+(147500+(n*128))),h
200 NEXT n
210 DATA 0,170,0,168,0,42,0,170,0,170,0,160,0,10,0,170,0,170,0,128,0,2,0,170
220 DATA 0,170,0,0,0,0,0,170,0,168,0,0,0,0,0,42,0,160,0,0,0,0,0,10,0,128,0,160,0
230 DATA 10,0,2,0,10,0,160,0,10,0,160,0,170,0,160,0,10,0,170
240 DATA 0,170,0,168,0,42,0,170,0,170,0,168,0,42,0,170,0,170,0,160,0,10,0,170
250 DATA 0,170,0,128,0,2,0,170,0,170,0,8,0,64,0,170,0,168,0,40,0,40,0,42
```

Interfaces

The first of a two-part evaluation of interfaces and printers designed specifically for the QL.

There are basically two methods of sending information to or from a computer. The bits that make up a character or byte may be sent simultaneously down a parallel set of eight wires or they may be sent serially, one after another down a single wire. The former affords faster rates of transfer but is more susceptible to corruption over long distances. The latter, though slower, is more reliable and cheaper to implement.

The QL has two RS-232C serial ports, but no Centronics-compatible parallel port. Most low cost (dot matrix) printers, however, make use of the latter. So if you want hard copy at a reasonable price, you'll need some form of interface to convert information transmitted serially into being received in parallel. Such devices go under a variety of names, depending upon where they plug in and what capabilities they have. However, their number is steadily growing.

Cheap And Cheerful

At the bottom of the price scale are the 'plug-in-and-go' interfaces from Sigma Research and Miracle systems. Requiring no special software, these are unsophisticated devices that are easy to use.

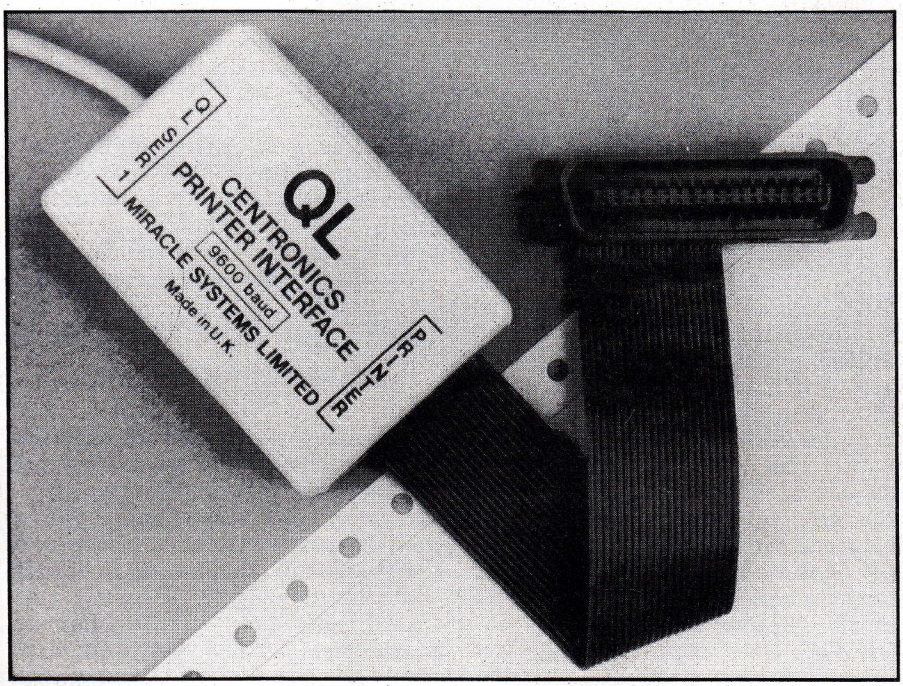
Aside from differing appearances, there is little to distinguish them externally. Both adopt the same configuration, with a lengthy serial cable at one end and a short ribbon cable at the other. The interface is located between these like a sort of junction box.

On the technical side, both units are also very similar. They convert information transmitted at the QL's default rate of 9600 baud. Also, presumably in the interests of cost and simplicity, neither permits this rate to be varied. Furthermore, both use the QL as a power source.

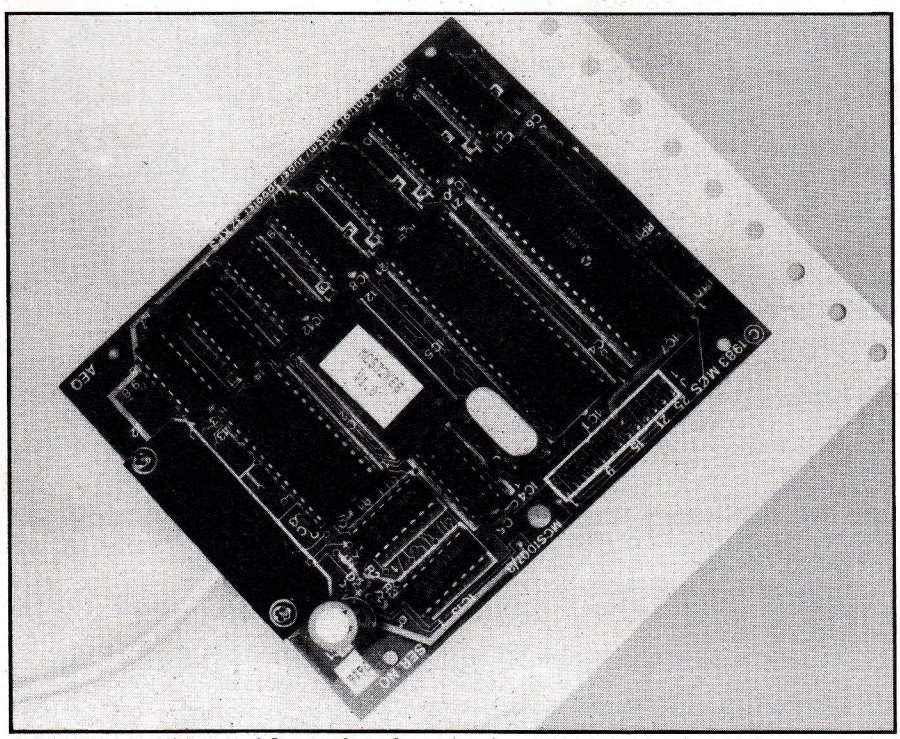
Overall, Sigma's interface, though marginally cheaper, appeared to be the better finished. The Centronics connector is metal as opposed to plastic. The UART specifically designed for inputting serial data and the PCB incorporates a ceramic resonator giving more accurate clock frequencies. Also the use of CMOS ICs throughout gives a very low current consumption.

Fixed Or Variable?

To get around the problem of fixed Baud rates, Care Electronics have



Miracle Systems parallel interface



Micro Control Systems' Super Spooler

in Print

recently released a 'serial to parallel converter'. Similar in design to the plug-in-and-go interfaces, it incorporates a switch that allows operation at 75, 300, 600, 1200, 2400, 4800 and 9600 baud. Competitively priced at under £38 it would appear to be very good value.

The facility to vary Baud rates is standard on the more expensive 'intelligent' serial interfaces, available from printer manufacturers themselves as well as independent suppliers. Using an RX-80FT printer to test such products, we looked at one of Epson's standard interfaces and one specifically designed for the QL from Micro Control Systems.

Installing either system requires that the printer be partially dismantled so that an additional circuit board can be fixed onto the PCB. Also DIP switches on the Epson and jumpers on MCS's board (determining such things as Baud rate, parity and handshaking) have to be set. Somewhat fiddly, users would be advised to leave installation to the retailer. However, failing this, Epson do provide a comprehensive manual covering each step of the procedure. MCS for their part have indicated that they will be dispensing with jumpers and hard wiring the options.

With the cable being an integral part of connecting the MCS, all that's left is to plug into the SER1 slot. To connect up the Epson, the lead supplied with the QL sufficed. However, if you do not have one of these then a BT/25way D-type lead will cost a further £12.

With the circuit boards in place, DIP switches or jumpers set at the QL's default rate and cables connected, the interfaces are as easy to operate as those previously examined. Their performance, however, is considerably better thanks to a device known as a buffer incorporated on the board.

The buffer acts as a reservoir, supplying a steady stream of characters to the printer, yet at the same time permitting the computer to transmit them at a much faster rate. When full, the printer signals to the computer to pause until the buffer is emptied. Then the process repeats itself.

Buffering output, occasionally known as 'spooling', has the effect of reducing the amount of time that a computer is 'printer bound'. The bigger the buffer, the quicker a computer can divest itself of all the characters to be printed and move onto another task leaving the printer to continue on its own. To get some idea of the savings involved, printing out 16K of text (four foolscap pages) would take nearly five minutes on an FX-80. With a 16K buffer this is reduced to 32 seconds.

An alternative to the external, buffered interface is provided by CST, who have neatly sidestepped the problems associated with the QL's serial ports by not using them at all. Instead, their Q-PI board fits into the expansion bus slot on the left hand edge of the QL, adding a further two inches of matching ribbed black plastic.

In effect, CST's device adds a Centronics compatible port to the QL's existing complement. A 2K ROM on the PCB supplies QDOS with the necessary drivers. This arrangement has certain advantages. First, it can harness the QL's own RAM to act as a print buffer; though, admittedly at the expense of memory in which programs may run. Secondly, transmission rates are limited only to the speed at which the printer is capable of digesting information.

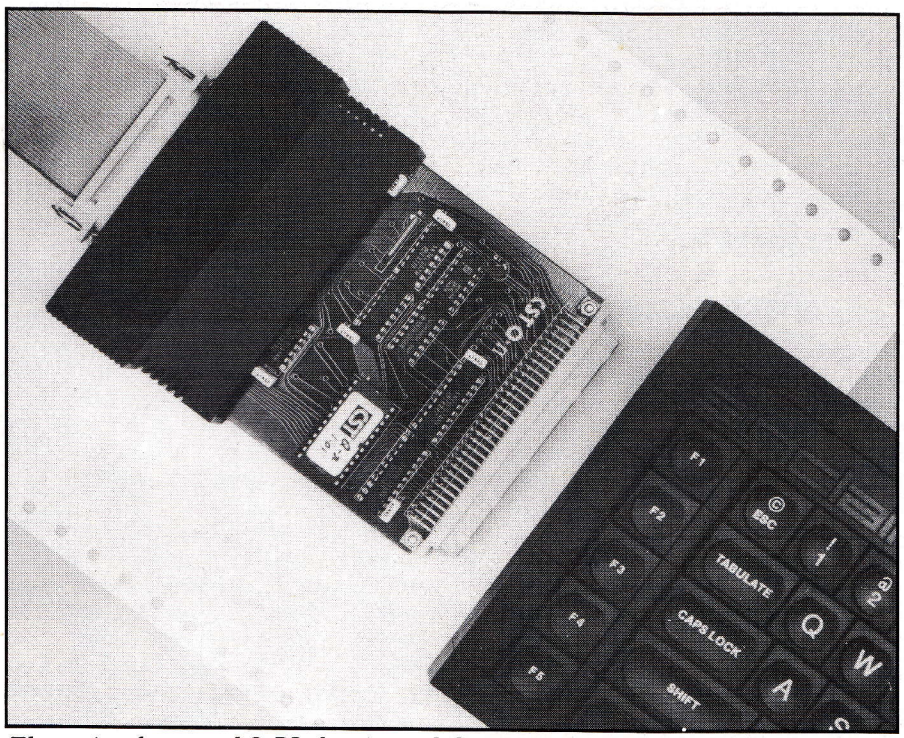
The major drawback with Q-PI is that the slot it occupies is the one designated for RAM extensions and disk drives. However, expansion modules are currently being produced which allow the slot to accommodate more than one device. Additionally, whilst Q-PI, with its own drivers does require some additional commands for it to run, they are consistent with those governing serial ports and as such the user need not worry about such things as baud rate, parity and handshaking. For example, to open a channel for printing you need only write "open #3,par". More interesting, to print out the contents of a Microdrive, simply enter the command "copy mdv1_ filename to par_5000".

Miracle Systems, Avondale Workshops, Woodland Way, Kingswood, Bristol BS15 1QL ***£3.95***

Sigma Research, 95 Vinery Road, Cambridge CB1 3DW ***£39***

Care Electronics, Unit 14, Peerglow Industrial Estate, Old's Approach, Tolpits Lane, Watford, Herts WD1 8SR ***£38***

A>Line Computer Systems, 1 Church Lane, Willoughby, Waterleys, Liecs LE8 3UD ***£61 (2K) to £173 (64K)***



The curiously-named Q-PI plug-in module

Making the Most of Multitasking

Multi-tasking – the ability to run more than one program concurrently – has never been so readily available on a micro as cheap as the QL, but somehow the technique of writing programs to multi-task has gained a false reputation for being difficult.

Of course, multi-tasked programs don't *actually* run at the same time, as a simple serial processor like the 68000 is not capable of that, but the illusion is very realistic. What happens is that each program, known as a job, is allocated a slice of the processor's time. By choosing a suitable length of time for each slice, it is easy to make each job seem to be continually running. A compromise must be made between the amount of time spent allocating the time slices and the length of each slice, otherwise nothing will ever get done!

On Schedule

The program controlling the time slice allocation is known as the scheduler. It runs in the 68000's supervisor mode, continually stopping and starting each job. These generally run in user mode, as stepping into supervisor mode stops the multi-tasking. When a job *does* enter supervisor mode it is generally through a QDOS routine which will at some point return to user mode.

A job can be in one of three well defined states. It can be *active* – it will, in time, be allocated its time slices – *inactive* – the job is present in memory but will never get an allocation of time – and *suspended* – the job is active but can't run (although it would dearly love to), because it is waiting for some other process to finish.

In the dark days before the QL, multi-tasking was the domain of the mini and mainframe. That has all changed now, as Adam Denning is quick to point out.

For example, a job talking to a printer will find itself suspended quite often as printers are very slow devices and cause the port to which they are connected to be periodically inactive. This in turn causes the job to be suspended. A job may also want to get some input from the keyboard, but is unable to at the moment because another job is using it.

A suspended job will eventually be *released*, and will then get its share of the time slices. A job awaiting input/output is quite likely to be released outside its time slice as the I/O subsystem is generally independent of the jobs.

High Priority

A programmer can decide how often a job is going to get a time slice by setting the job's *priority*. This varies between 0 and 127; with 127 being the highest possible. In theory, a job with a priority of 127 will be allocated time slices 127 times as often as a job with priority 1. A priority of zero is a special case. It means that a job will never get a time slice, but it doesn't necessarily mean that a job is technically inactive. A job may have been started at one point and then had its priority set to zero some time later.

There is also the concept of a job hierarchy, with one job owning another so that a job tree is created. All jobs created from BASIC (for example, using the EXEC or EXEC_W commands) are owned by the SuperBasic interpreter, which is the highest job and cannot be killed (removed). The SuperBasic interpreter itself has a priority of 32 until changed, and all jobs created by the interpreter are also given priorities of 32. When a job is killed, all jobs in the same tree below that job are also killed. The process of killing a job also releases any channels it may have open and any RAM it may have adopted from the system. It is, however, good practice to explicitly close a job's channels before killing the job (well, I think so anyway!).

Job ownership implies that it must be possible to uniquely identify a given job, and this is done by QDOS producing *job IDs* when jobs are created. In the current implementation the job ID is a long word (four bytes), consisting of a *job number* in the lower word and a *tag* in the higher word. The job number is related to the number of jobs currently in the machine. The BASIC interpreter is job number 0, and the next job created will be job number 1. Another job created will be given job number 2. Now, if we killed job number 1 and created a new job, it would be given job number 1.

The tag is rather different. It relates to the number of jobs which have been created since the system was last reset. The job created after the BASIC interpreter will be given tag 0, the next tag 1 and so on. Killing one of these jobs and creating another one will give it the next tag number, not the tag

corresponding to one of the killed jobs.

For example when a machine is first switched on, the only job present is job number 0, tag 0. This is the SuperBasic interpreter. If we loaded another two jobs they would be given numbers 1 and 2 with tags 0 and 1. Killing the first job and creating another will cause the new job to be given a job number of 1 but with a tag of 2.

The techniques for writing jobs involve taking account of two things: jobs are stopped and started automatically by the scheduler at undefined points in time, and QDOS creates a job in an arbitrary location in memory. So our code must be re-entrant and position-independent, neither of which is exactly hard to ensure.

Clocking On

The programs presented here demonstrate the techniques we have discussed, without being deviously difficult to understand. The first is a clock program which displays the current time and day at the top left hand corner of the screen. The source code is shown in the format required by the GST assembler, although as most assemblers follow the same general conventions the only words that may need changing are the INCLUDE and DATA directives (the Metacomco assembler uses the words GET and SIZE instead).

The first part of the program is known as the 'standard format code'. This declares that a word starting at byte 6 of the job must be $4AFB, and this is followed by the name of the job. The code proper starts at the C_START label. The first thing we do is set the priority of the clock to 1 so that it doesn't noticeably slow BASIC down. We then use the UT_SCR utility to open a screen window at the positions specified in PBLOCK.

Next enter a loop in which MT_MODE is called to check which screen mode the QL is in. This

Program 1

```
                        * Executable clock program for the QL
                        *
                        * By Adam Denning 22nd September 1984
                        * Copyright (C) 1984 Adam Denning
                        *
                                INCLUDE   'mdv1_header_asm'
                        *
00000100                        DATA      256               Assembler directive which sets
                        *                                   the job's data space to the
                        *                                   size specified in bytes. The
                        *                                   only effect is that this value
                        *                                   gets saved in the file header
                        *
0 00000000 6010                 BRA.S     C_START           Ignore standard format code
0 00000002 00000000             DC.L      0
0 00000006 4AFB                 DC.W      $4AFB             Standard format identification
0 00000008 0007                 DC.W      7                 Program name
0 0000000A 415F434C4F43         DC.B      'A_CLOCK',0
                        *
                        * MT_PRIOR sets the priority of the job whose ID is held in D1 to the
                        * value held in D2. It preserves all registers except A0 which ends up
                        * holding the base address of the job control area for this job
                        *
0 00000012 700B         C_START MOVEQ     #MT_PRIOR,D0      Set priority
0 00000014 72FF                 MOVEQ     #-1,D1            of this job
0 00000016 7401                 MOVEQ     #1,D2             to 1
0 00000018 4E41                 TRAP      #1
                        *
                        * UT_SCR is a vectored utility which opens a screen device (SCR_) using
                        * the parameters supplied in the definition block pointed to by A1. It
                        * returns with the ID of the opened channel in A0. D0 - D3 and A1 - A3
                        * are totally corrupted
                        *
0 0000001A 347B00C8             MOVE.W    UT_SCR,A2         Open screen window
0 0000001E 43FA0062             LEA.L     PBLOCK,A1         Using the definitions in PBLOCK
0 00000022 4E92                 JSR       (A2)
                        *
                        * MT_DMODE sets or reads the display mode depending on the keys sent as
                        * byte values in D1 and D2. -1 in D1 reads the mode, 0 sets the mode to
                        * 0 and 8 sets the mode to 8. -1 in D2 reads the display type, 0 sets it
                        * to monitor mode, 1 sets it to television mode. The routine returns with
                        * the display mode in D1, the display type in D2 and it corrupts A4
                        *
0 00000024 2F0C         RE_ENTRY  MOVE.L    A4,-(A7)          Preserve A4
0 00000026 7010                 MOVEQ     #MT_DMODE,D0      Read mode
0 00000028 72FF                 MOVEQ     #-1,D1
0 0000002A 1401                 MOVE.B    D1,D2
0 0000002C 4E41                 TRAP      #1
0 0000002E 285F                 MOVE.L    (A7)+,A4          Retrieve A4
                        *
                        * This section sets the window width held in DEFBLOCK to either 150 or
                        * 300 depending on the screen mode, whose value is currently in D1
                        *
0 00000030 43FA0054             LEA.L     DEFBLOCK,A1       Ready to alter window
0 00000034 32BC0096             MOVE.W    #150,(A1)         150 for mode 0
0 00000038 4A01                 TST.B     D1
0 0000003A 6702                 BEQ.S     MONITOR
0 0000003C E351                 LSL.W     (A1)              300 for mode 8
                        *
                        * Now alter the window accordingly using SD_WDEF. This trap requires A1
                        * to point to a parameter block containing four words :- the window
                        * width, its height, X origin and Y origin. A0 holds the ID of the
                        * channel being altered, D3 holds the timeout, D1.B holds the border
                        * colour and D2.W holds the border width. All registers except D1 and A1
                        * are preserved
                        *
0 0000003E 700D         MONITOR   MOVEQ     #SD_WDEF,D0
0 00000040 7200                 MOVEQ     #0,D1             No border colour
0 00000042 2401                 MOVE.L    D1,D2             No border width
0 00000044 76FF                 MOVEQ     #-1,D3            Infinite timeout
0 00000046 4E43                 TRAP      #3                Alter window
0 00000048 2F08                 MOVE.L    A0,-(A7)          Save channel ID
                        *
                        * MT_RCLCK reads the real time clock. It is passed no parameters but
                        * returns the long integer time in D1 and corrupts D2 and A0
                        *
0 0000004A 7013                 MOVEQ     #MT_RCLCK,D0      Read the clock into D1.L
0 0000004C 4E41                 TRAP      #1
0 0000004E 205F                 MOVE.L    (A7)+,A0          Retrieve channel ID
0 00000050 2F01                 MOVE.L    D1,-(A7)          Save the read time
                        *
                        * Make A1 point to a suitable stack address relative to A6 then call
                        * the vectored utility CN_DATE, which converts the long integer time in
                        * D1 to a string on the stack pointed to by A1. It preserves all
                        * registers except A1 which now points to the start of the string. The
                        * routine requires 'at least 22 bytes' of A1 stack
                        *
0 00000052 43EC0032             LEA.L     50(A4),A1         Put suitable stack address in A1
0 00000056 347B00EC             MOVE.W    CN_DATE,A2        A4 holds offset from A6 of start
0 0000005A 4E92                 JSR       (A2)              of data area; convert date to ASCII
                        *
                        * Make the A1 stack pointer absolute by adding A6 to it, and then use
                        * the UT_MTEXT routine to print the string. UT_MTEXT requires the ID
                        * of the channel to which the string is to be printed to be in A0 and A1
                        * must point to the base of the string. D0 - D3 and A1 - A3 are ruined
                        *
0 0000005C D3CE                 ADDA.L    A6,A1             UT_MTEXT needs A1 pointing to
0 0000005E 347B00D0             MOVE.W    UT_MTEXT,A2       an absolute address, not A6
0 00000062 4E92                 JSR       (A2)              relative. Print string
                        *
                        * IO_SBYTE sends the byte held in D1 to the channel whose ID is held in
                        * A0. D3 must hold the timeout and on return all registers except D0,D1
                        * and A1 are preserved
                        *
0 00000064 7005                 MOVEQ     #IO_SBYTE,D0      Send a space
0 00000066 7220                 MOVEQ     #' ',D1
0 00000068 76FF                 MOVEQ     #-1,D3            Infinite timeout
0 0000006A 4E43                 TRAP      #3
                        *
                        * CN_DAY is identical to CN_DATE except that it converts the long integer
                        * time in D1 to a day of the week as a string on the A1 stack. It only
                        * requires 'at least 3 bytes' on the A1 stack
                        *
0 0000006C 221F                 MOVE.L    (A7)+,D1          Retrieve clock time
0 0000006E 43EC0032             LEA.L     50(A4),A1         Suitable stack address in A1
0 00000072 347B00EE             MOVE.W    CN_DAY,A2         Convert to ASCII day string
0 00000076 4E92                 JSR       (A2)
                        *
                        * As for CN_DATE string printing, make A1 absolute and then use UT_MTEXT
                        * as before to print the day of the week to the screen
                        *
0 00000078 D3CE                 ADDA.L    A6,A1             Change from A6 rel to absolute
0 0000007A 347B00D0             MOVE.W    UT_MTEXT,A2       Print day
0 0000007E 4E92                 JSR       (A2)
                        *
0 00000080 60A2                 BRA.S     RE_ENTRY          Loop as scheduler permits
                        *
0 00000082 0000         PBLOCK  DC.W      0                 No border
0 00000084 1207                 DC.W      $1207             Stippled red/black paper, white ink
0 00000086 0096         DEFBLOCK DC.W     150               Window width
0 00000088 000A                 DC.W      10                Window height
0 0000008A 0024                 DC.W      36                Window X position
0 0000008C 0005                 DC.W      5                 Window Y position
                        *
                                END
```

Note: The machine code loader for these programs will appear next month.

information is used to alter the size of the window so that it is just big enough to hold the date information: 1984 Oct 27 15:34:15 Sat. Notice that register A4 is saved while we're doing this. This is because when a job is first activated registers A4, A5 and A6 are set up so that A6 holds the address of the start of the job, (A6,A4.L) points to the start of the job's data area and (A6,A5.L) points to the end of the job. MT_RCLCK reads the current time as a long integer into D1, and CN_DATE converts this into the string: 1984 Oct 27 15:34:15. CN_DATE needs A1 to point to a suitable stack area to put the time and date string, and A1 must be relative to A6 (in other words (A6,A1.L) points to the absolute address of the stack). UT_MTEXT is used to print this string to our window, followed by IO_SBYTE to send a space.

Finally, CN_DAY converts the time into the day of the week and UT_MTEXT prints this. The program then branches back to RE_ENTRY to do it all over again.

Sounding The Alarm

The next program demonstrates how to create subsidiary jobs – it's still a clock, but this time it looks for the third program – an alarm – and loads it as a job if it finds it. Things are written in such a way that the second clock program will run quite normally as a clock if it can't find the alarm, and the alarm can be loaded and run by itself using the EXEC command.

Most of the code in the second clock program is very similar to the first version, but it is preceded by a block of code from just after C_START to NO_AJOB. Here we try to open the file containing the alarm program using IO_OPEN. The value in D3 here specifies which mode of access to use. OPEN_INS means 'Open for shared input', which means that it

Program 2

```
                        * Executable clock and alarm program for the QL
                        * Copyright (C) 1984 Adam Denning
                                INCLUDE   'mdv1_header_asm'
           00000096             DATA      150
           0000000F     HDR_LEN EQU       15              Maximum header length
 0 00000000 6010                BRA.S     C_START         Ignore standard format code
 0 00000002 00000000            DC.L      0
 0 00000006 4AFB                DC.W      $4AFB           Standard format identification
 0 00000008 0007                DC.W      7               Program name
 0 0000000A 415F434C4F43        DC.B      'A_CLOCK',0
 0 00000012 700B      C_START   MOVEQ     #MT_PRIOR,D0    Set priority
 0 00000014 72FF                MOVEQ     #-1,D1          of this job
 0 00000016 7401                MOVEQ     #1,D2           to 1
 0 00000018 4E41                TRAP      #1
 0 0000001A 41FA00C0            LEA.L     ALARM_JOB,A0
 0 0000001E 72FF                MOVEQ     #-1,D1
 0 00000020 7601                MOVEQ     #OPEN_INS,D3
 0 00000022 7001                MOVEQ     #IO_OPEN,D0
 0 00000024 4E42                TRAP      #2
 0 00000026 4A80                TST.L     D0
 0 00000028 664A                BNE.S     NO_AJOB
 0 0000002A 740F                MOVEQ     #HDR_LEN,D2
 0 0000002C 76FF                MOVEQ     #-1,D3
 0 0000002E 43FA00CA            LEA.L     D_SPACE,A1
 0 00000032 7047                MOVEQ     #FS_HEADR,D0
 0 00000034 4E43                TRAP      #3
 0 00000036 2F08                MOVE.L    A0,-(A7)
 0 00000038 43FA00C0            LEA.L     D_SPACE,A1
 0 0000003C 2411                MOVE.L    (A1),D2
 0 0000003E 26290006            MOVE.L    6(A1),D3
 0 00000042 72FF                MOVEQ     #-1,D1
 0 00000044 227C00000000        MOVE.L    #0,A1
 0 0000004A 7001                MOVEQ     #MT_CJOB,D0
 0 0000004C 4E41                TRAP      #1
 0 0000004E 4A80                TST.L     D0
 0 00000050 6708                BEQ.S     AJOB_OK
 0 00000052 205F                MOVE.L    (A7)+,A0
 0 00000054 7002                MOVEQ     #IO_CLOSE,D0
 0 00000056 4E42                TRAP      #2
 0 00000058 601A                BRA.S     NO_AJOB
 0 0000005A 2248      AJOB_OK   MOVE.L    A0,A1
 0 0000005C 205F                MOVE.L    (A7)+,A0
 0 0000005E 2F01                MOVE.L    D1,-(A7)
 0 00000060 7048                MOVEQ     #FS_LOAD,D0
 0 00000062 76FF                MOVEQ     #-1,D3
 0 00000064 4E43                TRAP      #3
 0 00000066 7002                MOVEQ     #IO_CLOSE,D0
 0 00000068 4E42                TRAP      #2
 0 0000006A 221F                MOVE.L    (A7)+,D1
 0 0000006C 7401                MOVEQ     #1,D2
 0 0000006E 7600                MOVEQ     #0,D3
 0 00000070 700A                MOVEQ     #MT_ACTIV,D0
 0 00000072 4E41                TRAP      #1
 0 00000074 347800C8  NO_AJOB   MOVE.W    UT_SCR,A2       Open screen window
 0 00000078 43FA0074            LEA.L     PBLOCK,A1       Using the definitions in PBLOCK
 0 0000007C 4E92                JSR       (A2)

 0 0000007E 2F0C      RE_ENTRY  MOVE.L    A4,-(A7)        Preserve A4
 0 00000080 7010                MOVEQ     #MT_DMODE,D0    Read mode
 0 00000082 72FF                MOVEQ     #-1,D1
 0 00000084 1401                MOVE.B    D1,D2
 0 00000086 4E41                TRAP      #1
 0 00000088 285F                MOVE.L    (A7)+,A4        Retrieve A4
 0 0000008A 43FA0066            LEA.L     DEFBLOCK,A1     Ready to alter window
 0 0000008E 32BC0096            MOVE.W    #150,(A1)       150 for mode 0
 0 00000092 4A01                TST.B     D1
 0 00000094 6702                BEQ.S     MONITOR
 0 00000096 E3D1                LSL.W     (A1)            300 for mode 8
 0 00000098 700D      MONITOR   MOVEQ     #SD_WDEF,D0
 0 0000009A 7200                MOVEQ     #0,D1           No border colour
 0 0000009C 2401                MOVE.L    D1,D2           No border width
 0 0000009E 76FF                MOVEQ     #-1,D3          Infinite timeout
 0 000000A0 4E43                TRAP      #3              Alter window
 0 000000A2 2F08                MOVE.L    A0,-(A7)        Save channel ID
 0 000000A4 7013                MOVEQ     #MT_RCLCK,D0    Read the clock into D1.L
 0 000000A6 4E41                TRAP      #1
 0 000000A8 205F                MOVE.L    (A7)+,A0        Retrieve channel ID
 0 000000AA 2F01                MOVE.L    D1,-(A7)        Save the read time
 0 000000AC 43EC0032            LEA.L     50(A4),A1       Put suitable stack address in A1
 0 000000B0 347800EC            MOVE.W    CN_DATE,A2      A4 holds offset from A6 of start
 0 000000B4 4E92                JSR       (A2)            of data area; convert date to ASCII
 0 000000B6 D3CE                ADDA.L    A6,A1           UT_MTEXT needs A1 pointing to
 0 000000B8 347800D0            MOVE.W    UT_MTEXT,A2     an absolute address, not A6
 0 000000BC 4E92                JSR       (A2)            relative. Print string
 0 000000BE 7005                MOVEQ     #IO_SBYTE,D0    Send a space
 0 000000C0 7220                MOVEQ     #' ',D1
 0 000000C2 76FF                MOVEQ     #-1,D3          Infinite timeout
 0 000000C4 4E43                TRAP      #3
 0 000000C6 221F                MOVE.L    (A7)+,D1        Retrieve clock time
 0 000000C8 43EC0032            LEA.L     50(A4),A1       Suitable stack address in A1
 0 000000CC 347800EE            MOVE.W    CN_DAY,A2       Convert to ASCII day string
 0 000000D0 4E92                JSR       (A2)
 0 000000D2 D3CE                ADDA.L    A6,A1           Change from A6 rel to absolute
 0 000000D4 347800D0            MOVE.W    UT_MTEXT,A2     Print day
 0 000000D8 4E92                JSR       (A2)
 0 000000DA 60A2                BRA.S     RE_ENTRY        Loop as scheduler permits
 0 000000DC 000F      ALARM_JOB DC.W      15
 0 000000DE 6D6476315F61        DC.B      'mdv1_alarm_exec',0
 0 000000EE 0000      PBLOCK    DC.W      0               No border
 0 000000F0 1207                DC.W      $1207           Stippled red/black paper, white ink
 0 000000F2 0096      DEFBLOCK  DC.W      150             Window width
 0 000000F4 000A                DC.W      10              Window height
 0 000000F6 0024                DC.W      36              Window X position
 0 000000F8 0005                DC.W      5               Window Y position
           000000FA   D_SPACE   EQU       *               Start of job data space
                                END
****** TOTAL ERRORS    0 (line    0)
**** TOTAL WARNINGS    0 (line    0)
      memory usage    8 kbytes
```

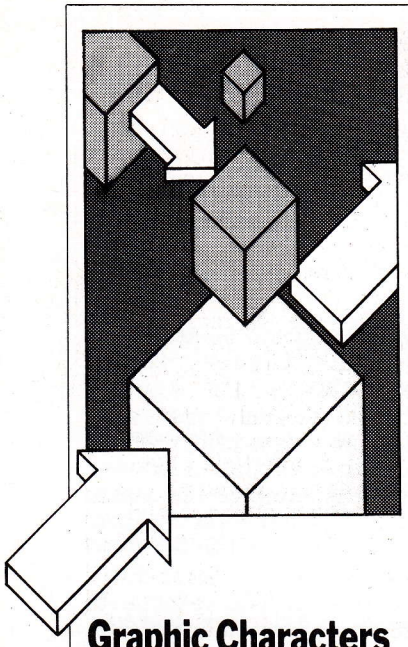
will open the file for reading and leave it in such a state that other programs can read the same file at the same time. The name of the file is held as a QDOS string (*1-word length, the rest the characters of the string*) at ALARM_JOB. Here we've called it **mdv1_alarm_exec**, but any name will do. If this open fails, then D0 will not be zero after IO_OPEN, so we jump straight to the main part of the clock program. Otherwise we read the alarm job's file header into memory. This header contains two pieces of important information – the length of the job and the length of its data space. We load these values into D2 and D3 and use MT_CJOB to create the job. *THIS DOES NOT ACTIVATE IT*. The ID of the newly created job is returned in D1 and the address at which it is to be loaded in A0. We use FS_LOAD to load the file into the allocated area and then close the file as it is no longer needed.

Finally, activate the job with a priority of 1. Notice that we put 0 in D3 before calling MT_ACTIV, as we want the clock to continue running once the alarm has been started. If −1 was used instead, the effect would have been similar to EXEC_W – the clock wouldn't start until the alarm had finished. After that the clock program is the same as we used before.

The alarm program opens a console device and prompts for an alarm time. This should be in the form 15:00:00 or whatever time you want. It now uses MT_RCLCK to continually read the time until the real time is equal to the alarm time. When this happens the program sends a command to the 8049 co-processor to tell it to make a disgusting racket. The routine to send a command is MT_IPCOM, and the command for sound is shown in I_SOUND. The program then continually scans the keyboard until F1 is pressed, when it sends another command to the IPC telling it to shut-up. Finally, the job kills itself.

Program 3

```
                        * A program to demonstrate communication with the 8049 co-processor
                        * A multi-tasking alarm!
                        * By Adam Denning 15/10/84 Copyright (C) 1984 Adam Denning

00000096                DATA    150

                        INCLUDE 'mdv1_header_asm'

00000032    STACK       EQU     50              Safe place for stack
00000008    TIME_LEN    EQU     8               Length of time string

0 00000000 601A                 BRA.S   A_START         Ignore standard format code
0 00000002 00000000             DC.L    0
0 00000006 4AFB                 DC.W    $4AFB           Standard format ID word
0 00000008 0005                 DC.W    5               Job name
0 0000000A 414C41524D00         DC.B    'ALARM',0

                        * The definition block for a console device to be opened with UT_CON

0 00000010 0000     PBLOCK      DC.W    0               No border
0 00000012 0007                 DC.W    7               Black paper with white ink
0 00000014 012C                 DC.W    300             Window width
0 00000016 0014                 DC.W    20              Window height
0 00000018 0024                 DC.W    36              Window X origin
0 0000001A 000F                 DC.W    15              Window Y origin

0 0000001C 700B     A_START     MOVEQ   #MT_PRIOR,D0    Set priority
0 0000001E 72FF                 MOVEQ   #-1,D1          of this job
0 00000020 7401                 MOVEQ   #1,D2           to 1
0 00000022 4E41                 TRAP    #1

0 00000024 43FAFFEA             LEA.L   PBLOCK,A1       Open specified console device
0 00000028 347800C6             MOVE.W  UT_CON,A2
0 0000002C 4E92                 JSR     (A2)
0 0000002E 66000080             BNE     ALARM_END       Leave if error

0 00000032 43FA00B2 GET_TIME    LEA.L   T_MESS,A1       Print 'Alarm time: ' to console
0 00000036 347800D0             MOVE.W  UT_MTEXT,A2
0 0000003A 4E92                 JSR     (A2)

0 0000003C 43FA00B4             LEA.L   BUFFER,A1       Fetch a string from the console
0 00000040 7409                 MOVEQ   #TIME_LEN+1,D2
0 00000042 76FF                 MOVEQ   #-1,D3
0 00000044 7002                 MOVEQ   #IO_FLINE,D0
0 00000046 4E43                 TRAP    #3

0 0000004B 0C410009             CMPI.W  #TIME_LEN+1,D1  Have enough characters been
0 0000004C 670C                 BEQ.S   VAL_TIME        collected?
0 0000004E 43FA0074             LEA.L   ERR_MESS,A1     No; print 'Invalid time!' to
0 00000052 347800D0             MOVE.W  UT_MTEXT,A2     console and try again
0 00000056 4E92                 JSR     (A2)
0 00000058 60D8                 BRA.S   GET_TIME

0 0000005A 5341     VAL_TIME    SUBQ.W  #1,D1           Remove terminating line feed
0 0000005C 43FA0092             LEA.L   BUF_POS,A1      Store string length at start
0 00000060 3281                 MOVE.W  D1,(A1)         of string buffer

0 00000062 7002                 MOVEQ   #IO_CLOSE,D0    Close console channel
0 00000064 4E42                 TRAP    #2

0 00000066 7013     CHK_TIME    MOVEQ   #MT_RCLCK,D0    Read the clock
0 00000068 4E41                 TRAP    #1
0 0000006A 43EC0032             LEA.L   STACK(A4),A1    Put safe stack address in A1
0 0000006E 347800EC             MOVE.W  CN_DATE,A2      Convert time to ASCII
0 00000072 4E92                 JSR     (A2)

0 00000074 D3FC0000000C         ADDA.L  #12,A1          Make A1 point to time in string
0 0000007A 3DBC00089800         MOVE.W  #TIME_LEN,0(A6,A1.L) Make time string time only
0 00000080 41FA006E             LEA.L   BUF_POS,A0      Point to alarm time string
0 00000084 91CE                 SUBA.L  A6,A0
0 00000086 347800E6             MOVE.W  UT_CSTR,A2      Compare the two strings
0 0000008A 7003                 MOVEQ   #3,D0
0 0000008C 4E92                 JSR     (A2)
0 0000008E 4A80                 TST.L   D0              Equal?
0 00000090 66D4                 BNE.S   CHK_TIME        No; try again

0 00000092 7011                 MOVEQ   #MT_IPCOM,D0    Send a command to 8049 IPC
0 00000094 47FA003E             LEA.L   I_SOUND,A3      'Start sound' parameter block
0 00000098 4E41                 TRAP    #1

0 0000009A 7011     CHK_ROW     MOVEQ   #MT_IPCOM,D0    Send a command to 8049 IPC
0 0000009C 47FA004A             LEA.L   KEYROW_0,A3     'Read keyrow 0' parameter block
0 000000A0 4E41                 TRAP    #1

0 000000A2 02010002             ANDI.B  #2,D1           Is F1 being pressed?
0 000000A6 67F2                 BEQ.S   CHK_ROW

0 000000A8 7011                 MOVEQ   #MT_IPCOM,D0    Send a command to 8049 IPC
0 000000AA 47FA0038             LEA.L   K_SOUND,A3      'Kill sound' parameter block
0 000000AE 4E41                 TRAP    #1

0 000000B0 7005     ALARM_END   MOVEQ   #MT_FRJOB,D0    Now kill the job
0 000000B2 72FF                 MOVEQ   #-1,D1
0 000000B4 4E41                 TRAP    #1

0 000000B6 000C     T_MESS      DC.W    12
0 000000BB 416C61726D20         DC.B    'Alarm time: '

0 000000C4 000E     ERR_MESS    DC.W    14
0 000000C6 496E76616C69         DC.B    'Invalid time!',10

0 000000D4 0A       I_SOUND     DC.B    $A              Initiate sound command
0 000000D5 08                   DC.B    8               Eight parameters
0 000000D6 0000AAAA             DC.L    $AAAA           All 8 parameters are 8 bits
0 000000DA 00                   DC.B    0               Pitch 1
0 000000DB 32                   DC.B    50              Pitch 2
0 000000DC 2000                 DC.W    $2000           Interval between steps
0 000000DE 0000                 DC.W    0               Infinite duration
0 000000E0 11                   DC.B    $11             step of 1; wrap of 1
0 000000E1 00                   DC.B    0               no randomness or fuzziness
0 000000E2 01                   DC.B    1               no reply
0 000000E3 00                   DC.B    0               round to even address

0 000000E4 0B       K_SOUND     DC.B    $B              Kill sound
0 000000E5 00                   DC.B    0               No parameters
0 000000E6 01                   DC.B    1               no reply
0 000000E7 00                   DC.B    0               round to even address

0 000000E8 09       KEYROW_0    DC.B    9               Read keyboard row command
0 000000E9 01                   DC.B    1               One parameter
0 000000EA 00000000             DC.L    0               lowest four bits only
0 000000EE 00                   DC.B    0               row 0
0 000000EF 02                   DC.B    2               Reply of 8 bits

0 000000F0 0000     BUF_POS     DC.W    0
           000000F2 BUFFER      EQU     *

                    END
****** TOTAL ERRORS    0 (line    0)
**** TOTAL WARNINGS    0 (line    0)
      memory usage  12 kbytes
```



THE PROGS

This is the place to look for readers' QL programs. So, if you've got a computational masterpiece, why not send it in for evaluation. The address is 'The Progs', QL User, Scriptor Court, 155 Farringdon Road, EC1R 3AD. We pay for everything published.

Graphic Characters

For a machine as powerful as the QL, it's surprising that there is no facility for user-defined character graphics. This program fills that gap in two ways: firstly, a machine code routine which plots character graphics quicker than would be possible in SuperBasic; and secondly, a SuperBasic program to generate those character graphics initially.

The SuperBasic program calls the machine code routine so that characters generated can be displayed at their normal size. The screen starts at memory location $20000 and ends at $28000 (decimal 131072 to 163840). $20000 is the top left hand corner of the screen and successive locations run left to right across the screen, then down to the next line and so on. In Mode 8 (Low Resolution mode), a 16-bit word controls four pixels; the three primaries (R, G & B) and 'flash'. To set pixel N to white is a simple matter:

$$G_N = R_N = B_N = 1$$

The mixture of Red, Green and Blue used to give a pixel any particular colour is tabulated in the QL manual, (concepts section) under 'Colour' (the flash bits, Fx, are not used in this program).

To maintain versatility, the machine code routine plots the required character at any screen position in any of the eight paper and ink colours (the codes for the colours are the same as those used by SuperBasic). This makes the routine more complex than would be necessary if, for instance, the paper colour was kept as black.

The nature of the screen mapping readily falls into a pattern of four pixels (P_0 0 P_1 0 P_2 0 P_3 0)$_2$. The data for the character graphics is stored like this with four pixels per byte. Px = 1 means ink and Px = 0 means paper.

For each of the three primary colours the program has one of four options, depending on the paper (**P**) and ink (**I**) colours. This is summarised in the truth table below.

(I)	(P)	BINARY
NO	NO	0 00 00 00 00 0_2
NO	YES	$\bar{P}_0 0 \bar{P}_1 0 \bar{P}_2 0 \bar{P}_3 0_2$
YES	NO	$P_0 0 P_1 0 P_2 0 P_3 0_2$
YES	YES	1 01 01 01 0_2

The colours for (**I**) and (**P**) are Green, then Red, then Blue in turn. For instance, if we want a Black character on White paper: for the Green pixels we have no colour in (I), but colour in (P); which is line 2 of the truth table. Hence we require ($\bar{P}_0 0 \bar{P}_1 0 \bar{P}_2 0 \bar{P}_3 0)_2$.

```
MAIN LOOP :
MOVEQ  23,D1                                    114,23
MOVEQ   2,D0                                    112,2
BSR     40          }                           97,40
ASL     D6,8        } GREEN                     225,70
MOVE    D6,D7       }                           62,6
MOVEQ   1,D0                                    112,1
BSR     32          }                           97,32
MOVE.B  D6,D7       } RED                       30,6
MOVEQ   0,D0                                    112,0
BSR     26          }                           97,26
ASR     1,D6        } BLUE                      226,70
ANDI    $55,D6      }                           2,70,0,85
ADD     D6,D7       }                           222,70
MOVE    D7,(A1)+    PLOT ON SCREEN              50,199
ADDQ.L  1,A2                                    82,138
BTST    D0,D1                                   1,1
BNE     +4                                      102,4
ADDA #124,A1        NEXT ROW                    210,252,0,124
DBCS    -40,D1                                  85,201,255,216
RTS                                             78,117
```

(Main routine as called over page).

```
######################
# GRAPHICS PROGRAM #
######################
```

```
1010 start=RESPR(1000) : MODE 8 : PAPER 4 : INK 7
1020 PAPER #0,0 : INK #0,7 : CLS : CLS#0 : MCODE : DIM c$(12,8)
1030 LBYTES mdv1_char,261616
1040 AT 3,0: PRINT " SPACE:"\"  changes"\"  colour"\\" CURSOR"\"  red/blue"
1050 PRINT \" ESC:"\"  escape"\\" SAVE:"\"  stores"\"  graphics"
1060 CSIZE 3,1 : AT 1,19 : PRINT "GRAPHICS" : CSIZE 2,0 : DRAWPIC
1070                                                    REPeat character
1080                                          REPeat mug_trap
1090 BEEP 5000,5
1100 CLS #0:INPUT #0,\("    Character to be changed? ");m$
1110 IF m$="SAVE" OR m$="save":                          EXIT character
1120 m$=m$ & "  " : m$=m$(1 TO 2)
1130 IF m$(1) INSTR "0123456789" =0 :          NEXT mug_trap
1140 IF m$(2) INSTR "0123456789 "=0 :          NEXT mug_trap
1150 CLS #0 : n=m$ : IF n>20 : n=20
1160 PRINT #0,\,"    CHARACTER ";n : g=261616+n*24 : gc=g : char=n
1170 FOR p=1 TO 12 : STRING:c$(p,1 TO 4)=a$ : STRING:c$(p,5 TO 8)=a$
1180 DRAWPIC : x=4 : y=6 : ox=x : oy=y
1190                                                    REPeat movement
1200 CALL start,1,y+1,x+9,2,2 : IF c$(y,x)="X" : CALL start,1,y+1,x+9,1,1
1210 FOR n=1 TO 5 : k$=INKEY$
1220 k$=INKEY$(-1) : IF k$=CHR$(10) OR k$=CHR$(27) :       EXIT movement
1230 x=x+(x<8 AND k$=CHR$(200))-(x>1 AND k$=CHR$(192))
1240 y=y+(y<12 AND k$=CHR$(216))-(y>1 AND k$=CHR$(208))
1250 colour=7 : IF c$(oy,ox)="X" : colour=0
1260 IF x<>ox OR y<>oy : CALL start,1,oy+1,ox+9,colour,colour : ox=x:oy=y
1270 IF k$=CHR$(32) AND c$(y,x)="X" : BEEP 200,0 : c$(y,x)="." : k$=""
1280 IF k$=CHR$(32) AND c$(y,x)="." : BEEP 200,0 : c$(y,x)="X"
1290                                                  END REPeat movement
1300 BEEP 1000,4 : CLS #0 : PRINT #0,\,"LEFT/RIGHT symmetry? (Y/N)";
1310 FOR n=1 TO 30 : k$=INKEY$
1320 k$=INKEY$(-1) : BEEP 1000,9 : CLS#0
1330 IF k$="Y" OR k$="y"
1340  FOR m=1 TO 12
1350   FOR n=1 TO 4 : c$(m,9-n)=c$(m,n)
1360  NEXT m
1370 END IF
1380 DUMP : DRAWPIC
1390                                                END REPeat character
1400 CLS : CSIZE 3,1 : AT 5,5 : PRINT "SAVING" : BEEP 10000,3
1410 DELETE mdv1_char
1420 SBYTES mdv1_char,261616,500
1430 CSIZE 2,0 : CLS : BEEP 10000,2 : CLS#0 : PRINT #0,\,"OK  !"
1440 REMark ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
1445 REMark ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
1450                                              DEFine PROCedure STRING
1460 LOCal b$,d,n
```

THE PROGS

This is obtained by reading the graphics character information $(P_0\emptyset P_1\emptyset P_2\emptyset P_3\emptyset)_2$, inverting to $(P_0\bar{1}P_1\bar{1}P_2\bar{1}P_3\bar{1})_2$, then ANDing with $(10101010)_2$.

The first section of the machine code routine sets up Address Register A_1 to be the starting location of the screen printing position, and Address Register A_2 to be the start of the graphics look-up table for the character selected. A_1 is a function of the start of the screen RAM and the row/column co-ordinates selected. A_2 is a function of the start of the graphics characters and the actual character selected.

The main loop of the routine is run 24 times by Data Register D_1. D_0 tells the subroutine which colour is being selected. Each pass through the loop generates a 16-bit pattern in D_7 for a line of four pixels. The character is eight pixels across and 12 pixels down, which produces a square. Green pixels go into the Most Significant Byte of D_7, Red pixels into the Least Significant. Blue pixels have to be shifted one place to the right before adding into the Least Significant Byte. This is facilitate interlacing with the Red pixels.

The subroutine sorts out the necessary coding of graphics information in accordance with the truth table given. D_6 is loaded with the graphics pattern and the modified value of this is returned to the main loop.

The SuperBasic program enables up to 21 characters to be defined and displayed. The displays of enlarged characters are not square since they are composed of one square per pixel – the graphics characters are eight pixels across and 12 pixels down.

Before the program is run for the first time, line 1420 must be executed. This saves 500 bytes of graphics characters on mdv1. If this is not done, the program will drop out at line 1030 where it looks for the graphics information. Subsequently, these random characters can be replaced by the user generated versions.

Program flow is easy to follow. Array C$ (row, column) contains the pattern of the graphics character being changed. 'X' corresponds to ink and '.' corresponds to paper. Line 1010 reserves 1000 bytes for the machine code routine and the graphics characters. 'Start' is the beginning of the machine code routine. On line 1020, MCODE calls the procedure of that name which writes the machine code routine into RAM, ready for execution.

STRING generates a four character string, a$, derived from the stored graphics information. However, line 1470 is unusual. What d = 170 && PEEK(g) means is make d the decimal value of the bitwise AND of binary forms, PEEK(g) and 170_{10} (ie, $170_{10} = 10101010_2$). This masks out every other bit from address g.

DRAWPIC simply draws the enlarged character being changed, with all 21 characters printed underneath.

DUMP places the finished character back into machine code form using procedure WRITE (WRITE performs the opposite function to STRING).

Operating the program couldn't be simpler. To change the colour of a pixel, move the cursor over it and press the space bar. The colour of the cursor reflects

MACHINE CODE ROUTINE

```
FORMAT: CALL, START, CHARACTER, ROW, COLUMN, PAPER, INK
                                  D1      D2      D3      D4      D5
MOVEA   #$020304,A1                                       34,124,0,2,3,4,
MOVEA   #$03FDF0,A2                                       36,124,0,3,253,240
ADDA    D3,A1                                             210,195
ADDA    D3,A1                                             210,195
ADDA    D3,A1                                             210,195
ADDA    D3,A1                                             210,195
MOVE    #$600,D3                                          54,60,6,0
SUBA    D3,A1                                             146,195
ADDA    D3,A1                                             210,195
DBCS    -4,D2                                             85,202,255,252
MOVE    #24,D3                                            54,60,0,24
SUBA    D3,A2                                             148,195
ADDA    D3,A2                                             212,195
DBCS    -4,D1                                             85,201,255,252
```

SCREEN START

GRAPHICS TABLE START

SUBROUTINE :

```
MOVEQ   0,D6                                              124,0
MOVE.B  (A2),D6                                           28,18
BTST    D0,D5                                             1,5
BNE     +16                                               102,16
BTST    D0,D4                                             1,4
BNE     +4                                                102,4
MOVEQ   0,D6            NO COLOUR IN EITHER               124,0
RTS                     INK OR PAPER                      78,117

NOT     D6                                                70,70
ANDI    $00AA,D6        COLOUR IS PAPER                   2,70,0,170
RTS                                                       78,117

BTST    D0,D4                                             1,4
BEQ     +2                                                103,2
MOVEQ   $AA,D6          COLOUR IN INK AND PAPER           124,170
RTS                     COLOUR IS INK                     78,117
```

```
1470 a$="" : d=170 && PEEK(g) : d=d/2
1480 FOR n=1 TO 4
1490  b$="." : IF d MOD 2=1 : b$="X"
1500  a$=b$ & a$ : d=INT(d/4)
1510 NEXT n
1520 g=g+1
1530
                                                        END DEFine
1540 REMark XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
1550                                                DEFine PROCedure DRAWPIC
1560 LOCal c,r,inc
1570 FOR r=1 TO 12
1580  FOR c=1 TO 8
1590   inc=7 : IF c$(r,c)="X" : inc=0
1600   CALL start,0,r+1,c+9,inc,inc
1610  NEXT c
1620 NEXT r
1630 FOR n=0 TO 20 : CALL start,n,15+(n MOD 2),n+4,7,0
1640
                                                        END DEFine
1650 REMark ******************************************************************
1660                                                DEFine PROCedure DUMP
1670 LOCal n,t
1680 t=gc
1690 FOR n=1 TO 12 : WRITE c$(n,1 TO 4) : WRITE c$(n,5 TO 8)
1700
                                                        END DEFine
1710 REMark $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
1720                                                DEFine PROCedure WRITE(d$)
1730 LOCal a$,b,n
1740 b=0 : a$=d$
1750 FOR n=1 TO 4 : b=4*b : IF a$(n)="X" :b=b+1
1760 b=b*2 : POKE t,b : t=t+1
1770
                                                        END DEFine
1785 REMark %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1780 REMark %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1790                                                DEFine PROCedure MCODE
1800 LOCal a,n
1810 RESTORE 1830
1820 FOR n=start TO start+121 : READ a : POKE n,a
1830 DATA  34,124,  0,  2,  3,  4, 36,124
1840 DATA   0,  3,253,240,210,195,210,195
1850 DATA 210,195,210,195, 54, 60,  6,  0
1860 DATA 146,195,210,195, 85,202,255,252
1870 DATA  54, 60,  0, 24,148,195,212,195
1880 DATA  85,201,255,252,114, 23,112,  2
1890 DATA  97, 40,225, 70, 62,  6,112,  1
1900 DATA  97, 32, 30,  6,112,  0, 97, 26
1910 DATA 226, 70,  2, 70,  0, 85,222, 70
1920 DATA  50,199, 82,138,  1,  1,102,  4
1930 DATA 210,252,  0,124, 85,201,255,216
1940 DATA  78,117,124,  0, 28, 18,  1,  5
1950 DATA 102, 16,  1,  4,102,  4,124,  0
1960 DATA  78,117, 70, 70,  2, 70,  0,170
1970 DATA  78,117,  1,  4,103,  2,124,170
1980 DATA  78,117
1990
                                                        END DEFine
2000 REMark <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
```

the true state of the block underneath.

When a character is completed, pressing ESC gives the user the choice of LEFT/RIGHT symmetry. When asked for the next character, you can enter "SAVE", which updates the machine code file mdv1_char containing the graphics data.

Once you've typed in the whole program, save it onto two separate microdrive cartridges before attempting to run it. The SuperBasic part should be easy to debug since error messages will appear. The machine code, on the other hand, will probably crash the machine if there is any slight error. If both the machine code and SuperBasic parts are bug ridden, you can eliminate the machine code bugs by using 1830 DATA 112, 0, 78, 117, 0, 0, 0, 0. This makes the machine code routine do nothing except return immediately. Hence the SuperBasic section can be debugged.

Characters printed 'below' the screen (ie, into memory locations after $28000) crash the QL. It is easy to do this by exceeding the limits of the row/column co-ordinates. No provision has been made in the machine code routine to trap such errors. Columns go up to 31 before they wrap around, and rows can go up to 20 (note that on row 20 the maximum column size is 23).

In And Out The Window

C Harker

As the QL is aimed at the serious end of the market, there is a need for a set of versatile input and output routines which will accept and print data at particular positions inside any window within a defined format.

This set of routines are designed to work in any SuperBasic program. They are self contained and accessed via a number of parameters.

All the routines need to know the window, and the row and column where printing must finish.

The first routine is a function called GET_FIGURE which allows you to accept numbers:

ie, X = GET_FIGURE (a,b,c,d,e)

where a = the window, b = the row where the number is to be input, c = the column where the number is to be input, d = the maximum length of the integer part and e = the maximum number of decimal places.

The value input is returned in X.

eg, x = get_figure (2,5,10,4,2)

This inputs a number at 5,10 in window 2 with the format 9999.99. During input of the number you can delete data already input by using the Left Arrow key. Only valid numbers can be entered (and a decimal point if applicable). The routine is especially useful for inputting data from a table, since if you set the values 'd' and 'e' correctly it is impossible for the user to corrupt your table.

The second routine is similar to the first except it inputs text.

ie, X$ = GET_TEXT$ (a,b,c,d,e)

where a,b,c = the same as for GET_FIGURE, d = the size of the frame which shows the test and e = the maximum length of the string of text.

This routine inputs a string of text up to a maximum length defined by 'e'. Not all the text is shown; only a small frame is shown, dictated by the 'd' parameter. The portion of the text displayed scrolls across the frame as you enter or delete (Left Arrow) text. The routine is useful for entering a long string (ie, an address) where you only have a small area on the screen in which to accept the data.

The third routine prints out numbers in a format as follows:

PFORMAT a,b,c,d,e

where a = the window, b = the y co-ordinate of the last decimal place, c = the x co-ordinate of the last decimal place, d = the value to be printed and e = the number of decimal places.

For instance, pformat 2,5,10,fred,2 will print at 5,10 in window 2 12.25 (if fred = 12.2476). Because the routine does not use strings it is only accurate to 6 significant figures.

Incidentally, the line with test$ in it, is only needed because of a bug (AH version) on my QL. It seems that if you find len(int(string)) before len(int(number)) you're OK. This bug only occurs when the machine is first turned on and has been fixed on the latest 'JM' ROM.

The last routine, PTEXT, is simple and has only been included to complete the set.

PTEXT a,b,c,d,e,f

where a,b,c = as before, d = the text or string to be printed, e = position in text where printing to start and f = length of text portion to be shown.

ie, ptext 2,2,2,fred$,6,12 will print 6789012 (if fred = '123456789012').

```
32000 DEFine FuNction get_figure(windo,row,column,int_length,dplaces)
32005 LOCal max_length,dec_length,number$,current_len
32010   number$="" : still_int = 1
32015   dec_length = 0 : current_len = 0
32020   max_length = int_length + dplaces + (dplaces > 0)
32025   REPeat get_digit
32030     current_len = LEN(number$)
32035     AT#windo,row,column - current_len : PRINT#windo; number$
32040     key=CODE(INKEY$(-1))
32045     SELect ON key
32050       =48 TO 57
32055         digit=key-48
32060         IF  (still_int AND current_len < int_length) OR (NOT still_int AND
current_len < max_length)
32065           number$ = number$ & digit
32070         END IF
32075       =46
32080         IF dplaces AND still_int AND current_len < max_length
32085           number$ = number$ & '.'
32090         END IF
32095         still_int = 0
32100       =10
32105         IF current_len = 0
32110           RETurn 0
32115         ELSE
32120           number = number$ : RETurn number
32125         END IF
32130       =192
32135         IF current_len > 0
32140           IF number$(current_len) = '.' : still_int = 1
32145           number$ = number$ (1 TO current_len - 1)
32150           AT#windo,row,column - max_lengtn
32155           PRINT#windo;FILL$(' ',max_length)
32160         ELSE
32165           still_int = 1
32170         END IF
32175     END SELect
32180   END REPeat get_digit
32185 END DEFine get_figure
32190 :
32195 DEFine FuNction get_text$(windo,row,col,space,max_length)
32200   LOCal back,front,i,lastkey$
32205   text$=''
32210   AT#windo; row,col-space+1 : PRINT#windo; FILL$(' ',space)
32215   back=1 : front=1
32220   REPeat text_loop
32225     key$=INKEY$(-1)
32230     key = CODE(key$)
32235     IF front<max_length+1
32240       SELect ON key
32245         =32 TO 127
32250           text$ = text$ & key$
32255           front = front + 1
32260           IF front > space : back = front - space
32265           AT#windo;row,col-(front-back)+1
32270           PRINT#windo; text$(back TO front-1)
32275       END SELect
32280     END IF
32285     IF key=192 AND front > 1
32290       text$ = text$(1 TO front-2)
32295       back = back - (back > 1)
32300       front = front - (front > 1)
32305       IF front=1 : text$=''
32310       AT#windo; row,col-space+1 : PRINT#windo; FILL$(' ',space)
32315       AT#windo; row,col-(front-back)+1
32320       IF front > 1 : PRINT#windo; text$(back TO front-1)
32325     END IF
32330     IF CODE(key$) = 10 : EXIT text_loop
32335   END REPeat text_loop
32340   RETurn text$
32345 END DEFine
32350 :
32355 DEFine PROCedure pformat (windo,row,col,value,dp)
32360   test$="yes" : test$=INT(LEN(test$))
32365   power=10^dp
32370   form_value = ABS(INT((value*power)+.5)/power)
32375   int_value  = INT(form_value)
32380   dp_value   = INT(((form_value - int_value )*power)+.5)
32385   IF dp>0 THEN AT#windo,row,col-dp-1 : PRINT#windo;"0.";FILL$('0',dp)
32390   IF (value >= 1) OR (value <= -1) OR (value=0)
32395     AT#windo,row,col-LEN(int_value)-((dp+1)*(dp>0))+1
32400     PRINT#windo ; int_value;
32405     IF dp > 0 THEN PRINT#windo ; '.';dp_value
32410   ELSE
32415     AT#windo,row,col-LEN(dp_value)+1
32420     PRINT#windo ; dp_value
32425   END IF
32430   IF value < 0
32435     AT#windo,row,col-(LEN(int_value)-(dp+1)*(dp>0))
32440     PRINT#windo; '-'
32445   END IF
32450 END DEFine
32455 :
32460 DEFine PROCedure ptext(windo,row,col,text$,start,length)
32465   AT#windo; row,col-length+1
32470   PRINT#windo; text$(start TO start+length-1)
32475 END DEFine
```

Recommended by QL-User magazine

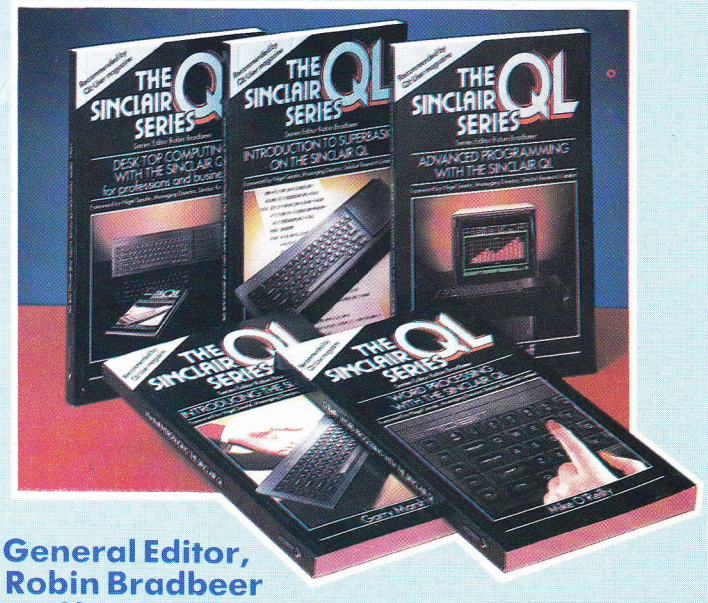
At Last! THE SERIES THAT MAKES THE QUANTUM LEAP

THE SINCLAIR QL SERIES

"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** @ £6.95 each
..... copies of **Introduction to SuperBASIC on the QL** @ £6.95 each
..... copies of **Advanced Programming with the Sinclair QL** @ £6.95 each
..... copies of **Desk-top Computing with the Sinclair QL** @ £6.95 each
..... copies of **Word Processing with the Sinclair QL** @ £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 _____

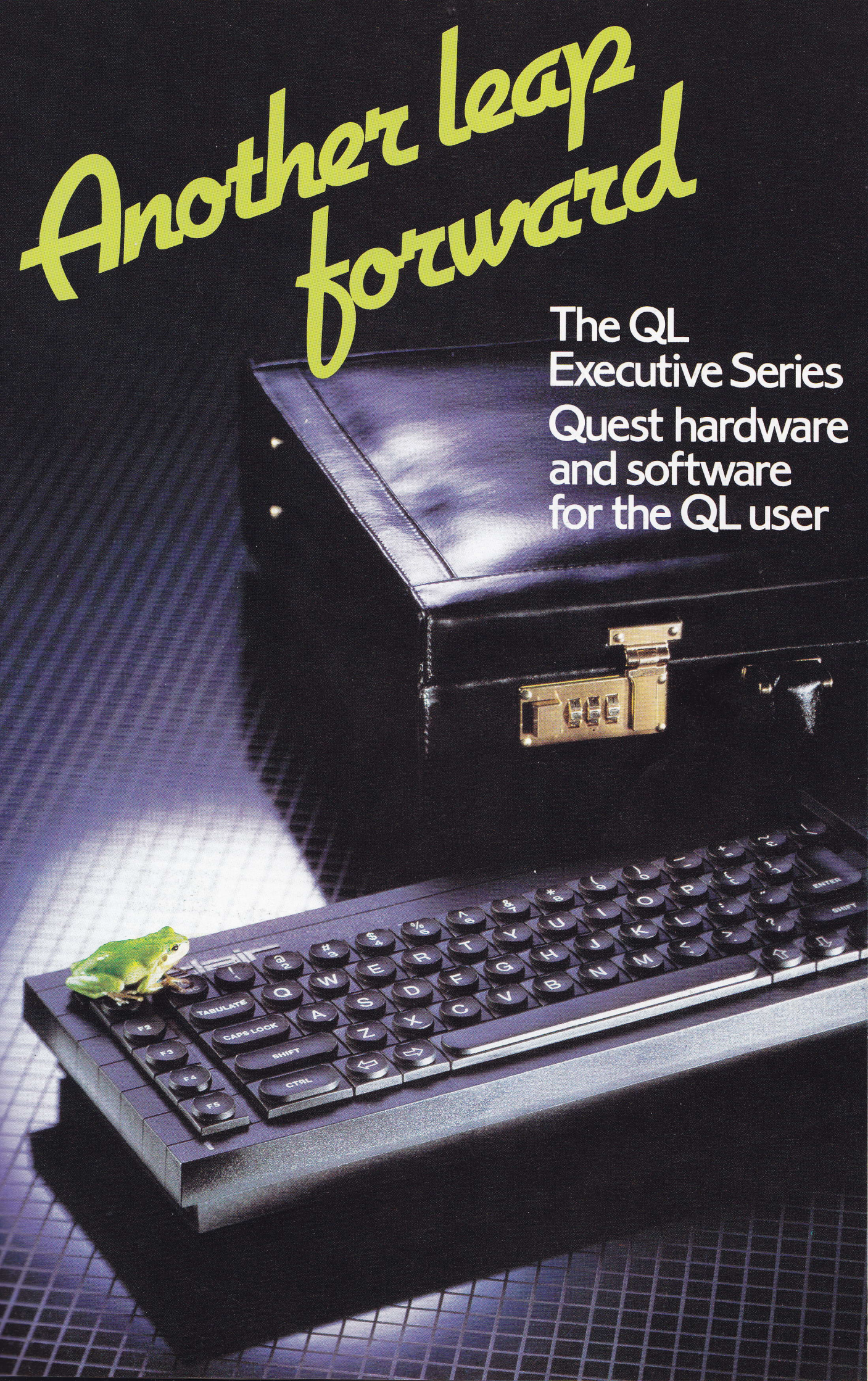
_____

_____

_____ Signed _____

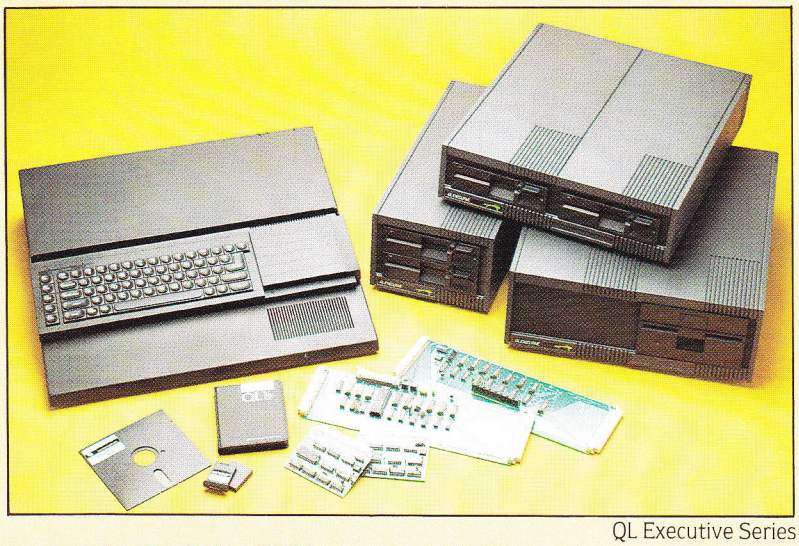
Dealer enquiries to:
Doug Fox, Hutchinson, 17-21 Conway Street, London W1

HUTCHINSON

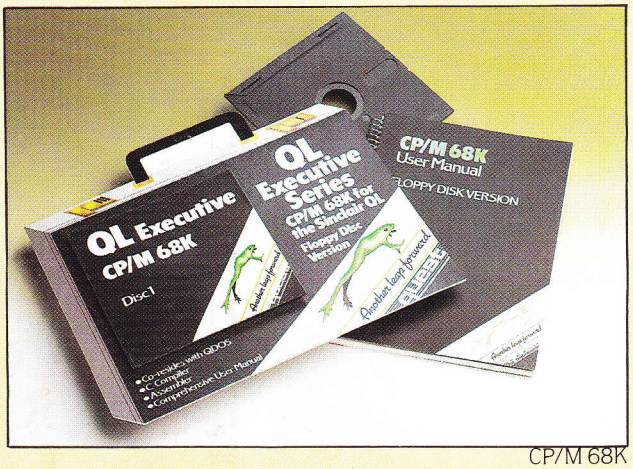

Another leap forward
The QL Executive Series
Quest hardware and software for the QL user
TABULATE
CAPS LOCK
SHIFT
CTRL
ENTER
SHIFT
F2
F3
F4
F5

The full potential of the QL can now be realised: Quest's QL Executive Series of quality peripherals and software enables the QL effortlessly to bridge the gap between the home and professional computer.

By providing industry standard operating systems and storage devices, Quest have opened up the way for the adoption of the QL by not only the serious home user but also by the business user.



QL Executive Series



CP/M 68K

CP/M - 68K

The Easy Route to Industry-Standard Applications Packages

Quest has the exclusive rights, worldwide, to CP/M 68K for the QL and has converted this industry standard operating system to run on the machine. The world's most widely used operating system, CP/M offers a proven and familiar environment in which to develop applications software.

- Co-resides with QDOS
- Users can switch between Operating Systems
- Includes Assembler and C Compiler
- Supplied on microdrive cartridges or floppy disk
- Supports 3", 3½", 5¼" and 8" floppy disks

N.B. CP/M requires either an Executive Series floppy disk system or an OScard to be present.

CP/M on Floppy Disk £59.50
CP/M on Microdrive £99.50

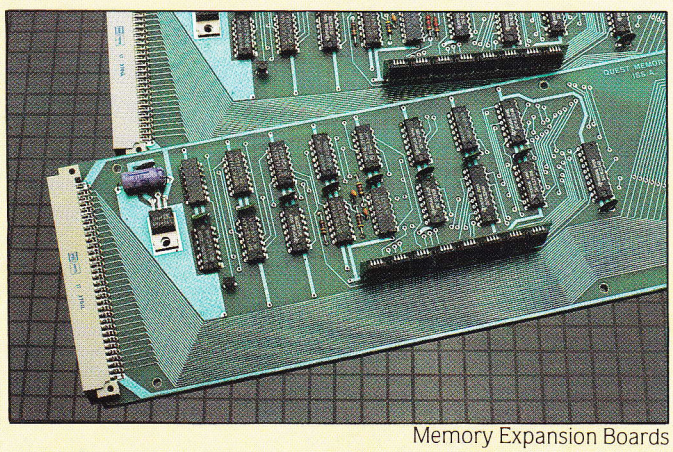
Memory Expansion Boards

The QL's standard 128K RAM is impressive, but because of the possibilities the machine offers, the programs used will frequently require additional capacity. Consequently, Quest has designed and produced a range of memory expansion cards to suit any requirement.

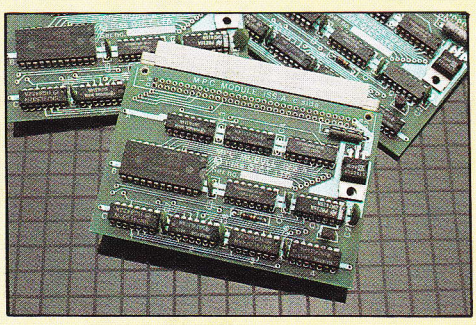
- Based on the latest 64K D RAM Modules
- Fully buffered
- Automatic refresh
- 150ns access to RAM contents
- Fit the standard QL and Quest's Expansion Console

The 64K unit can easily be powered by the QL's internal power source: larger units may need external power and provision for this has been made in their design.

64K RAM £115.00 **256K RAM £349.00**
128K RAM £185.00 **512K RAM £579.00**



Memory Expansion Boards



OScard

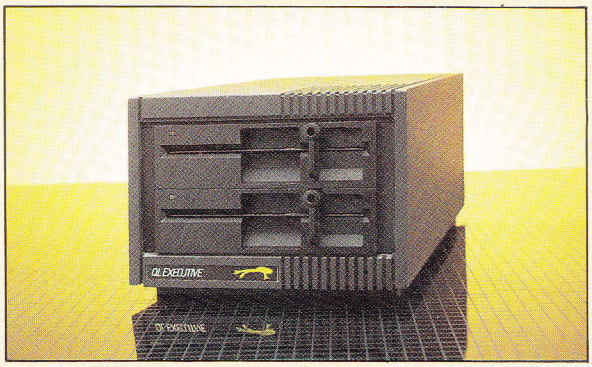
OScard

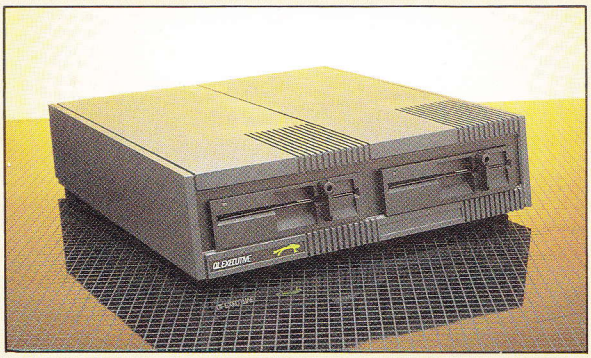
To enable microdrive or floppy disk versions to be used without the QL Executive Series floppy disk controller, this circuit board provides the necessary memory switch and RAM capacity. A full years warranty. **OScard £49.50**

Power Supply Unit

Specially designed to power Quest's memory expansion cards and the Expansion Console, this plug-in, stand-alone unit is covered by a full year's warranty.

Power Supply Unit £64.50





Disk Drives

With years of experience in disk drive te
have been able to harness their technica
commercial purchasing power to offer
capacity and reliability of disk storage a
prices.

Complete with disk interface controller/
unit, power supply and cables – no hidd
attractively styled and using state-of-th
the Executive Series of disk drives bring
professional mass storage to the QL.

- Latest half-height technology
- All necessary equipment included
- Operate under CP/M and QDOS
- Full year's warranty cover

200K Floppy Drive* £295
400K Floppy Drive £419
800K Floppy Drive £499
2 x 200K Floppy Drives £469
2 x 400K Floppy Drives £579
2 x 800K Floppy Drives £695

*Full height technology

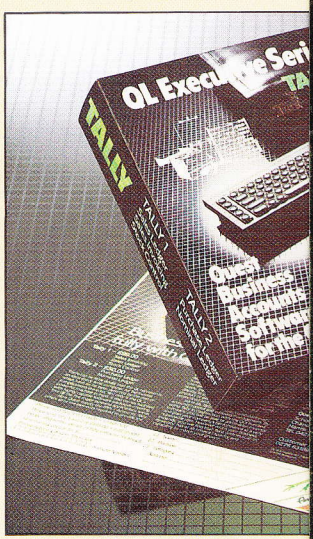
Tally

Business Accounts Software

Double Integration is the key to Tally, for the accounts modules contained in it link not only with each other but also with the four Psion packages – Quill, Archive, Easel and Abacus – supplied with the QL.

Tally I, comprises Sales Invoicing, Sales Ledger and Stock Control and Tally II consists of Purchase and Nominal Ledgers. These Quest programs enable the user to integrate, for example, Sales Ledger and Quill to produce standard letters without re-entering any data, or to use Easel to display in graphic form the data entered under Tally's Stock Control module.

Tally I £115.00
Tally II £59.50



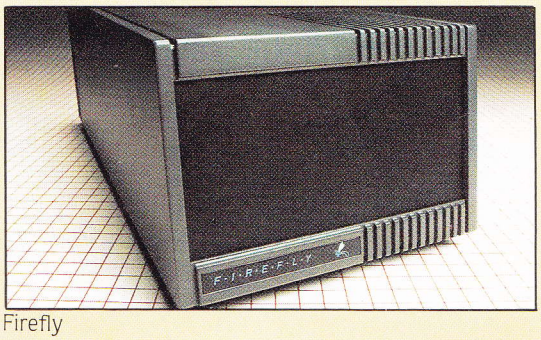
QL Expansion Console

Expansion Console

The Professional Desk Top System

Your QL and extra expansion cards can be c
stylishly housed in the QL Executive Expans
also serves as an ergonomically designed m
wrist-rest. Made of durable and attractive b
the console is available simply as the housi
module ready to take cards and including a

Expansion Console £139
Monitor Stand/Handrest £69.50



Firefly

Firefly QL

The Firefly QL Winchester disk offers a full 7.5 mb of data and program storage with extremely fast access times. Ideal for large databases such as stock and customer lists, these products represent true value for money – larger capacities coming.

Firefly £1149.00

See the QL Executive Series on Stand Number 3311 at The Which Computer Show, NEC Birmingham, 15th-18th January 1985.

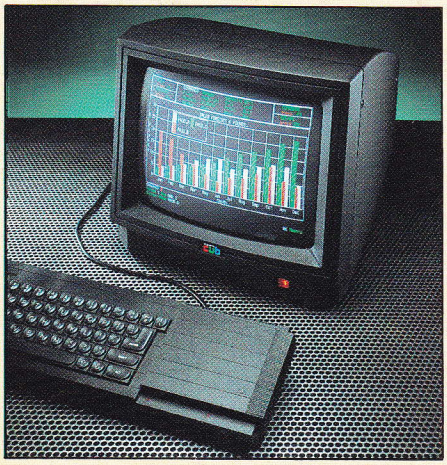
Executive Extras

As the major source of all QL related products, Quest is pleased to present a quality range from some of the world's leading manufacturers.

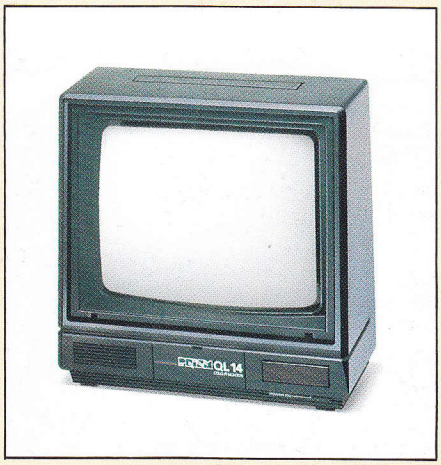
The Sinclair QL

Superb value for money, with 128k of user memory, 32 bit processor chip and 4 software packages (Word Processing, Spreadsheet, Graphics and Data base) included in the purchase price.

£399



Microvitec 1451/DQ3



Prism 14QL

Monitors

Microvitec 1451/DQ3

- 14" medium resolution colour monitor specifically designed for the QL
- 85 column text and outstanding graphics
- RGB TTL input

£274

Prism 14QL

- Portable, 85 column display
- Detachable anti-glare tinted glass
- Full A4 width on Quill
- Standard resolution for Easel graphics

£235

Printers

Brother HR-5

- Full 80 column printing
- Uses plain or thermal paper
- Compact and lightweight

£189

Brother HR-15

- Versatile daisy wheel printing
- Text re-printing
- Prints red and black
- Attachable cut sheet feeder

£519

Brother M-1009

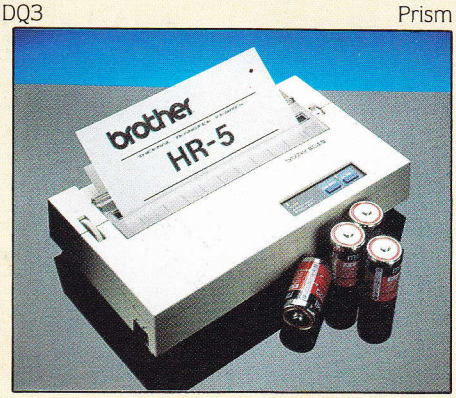
- Dot matrix printing at 50 cps
- Bi-directional, logic seeking printing
- Excellent value for business or home use

£224

Epson RX-80 F/T

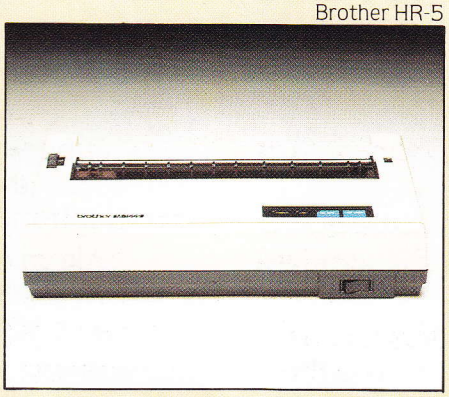
- 80 column printing at 100 cps
- Six user selectable graphics modes
- Six different densities
- Text and graphics on the same line

£334

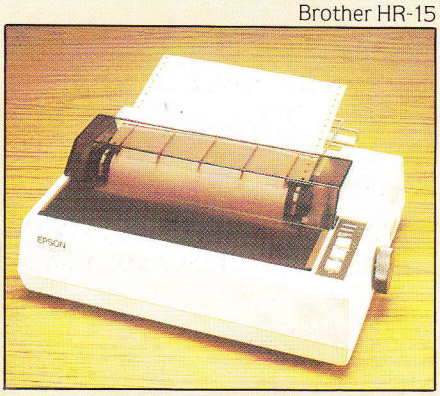


Brother HR-5

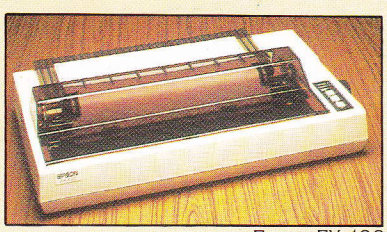
Brother HR-15



Brother M-1009



Epson RX-80 F/T



Epson FX-100

Epson FX-100

- High speed (160 cps) matrix printer
- 136 column capability
- Bi-directional, logic seeking printing

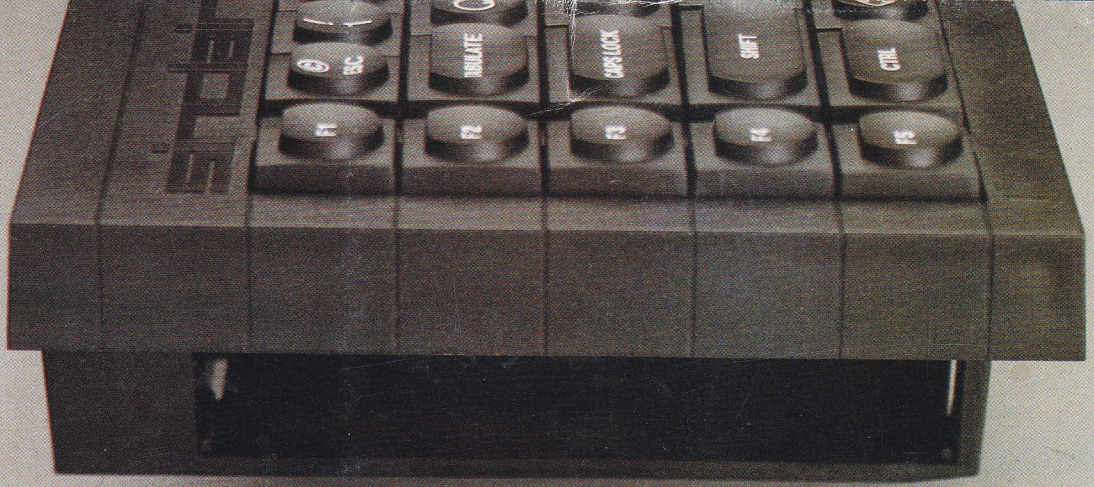
£664

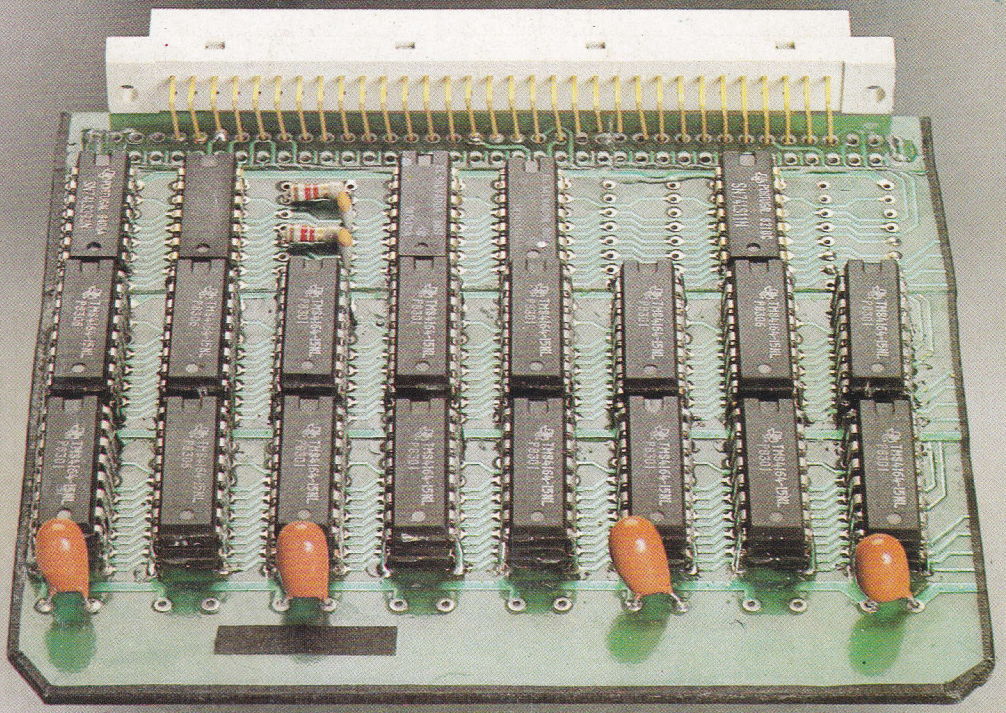
Consumables

A full range of consumables is available for your business and personal use, from microdrive cartridges and floppy disks through line printers and personalised stationery to desks and chairs. Please phone for our 48 page catalogue.

TELEPHONE SALES DESK 04215 66488
ALL PRICES INCLUDE VAT, POSTAGE AND PACKING

A Quest Automation plc Group Company
School Lane, Chandler's Ford, Hants. SO5 3YY
Tel: 04215 66321 Telex: 47326





Quarter Megabyte More

Designed and produced for the Sinclair QL enthusiast, this ingenious little memory expansion board gives you an extra 265k bytes of computing power to your existing system.

The majority of home computers have an in-built potential of using more memory than they are actually fitted with. The new memory expansion board from Simplex Data Ltd, takes full advantage of that wasted potential enabling you to expand your system — creatively! Two quarter megabyte memory expansion boards can be slotted together to expand your system to a further 512k bytes, giving you half a megabyte more.

Costing only £198.00 (inc. VAT, postage and packing), this unit is simple to use, just remove the port cover located at the end of the Sinclair QL Keyboard and slot the unit in, replacing the port cover after insertion.

Exclusively available from Simplex Data Ltd.

TO: SIMPLEX DATA LTD, FREE POST, 432 GREENFORD RD, GREENFORD, MIDDLESEX, UB6 8SG

Please send me: _____ (qty) QL Memory Expansion Unit(s) at £198.00 (inc. VAT, Post and Packing) I enclose Cheque/Postal Order for £ _____

Name: _____

Address: _____ Tel: _____

1 year warranty included. Please allow 28 days for delivery

Payment by ACCESS or BARCLAYCARD phone 01-575 7531