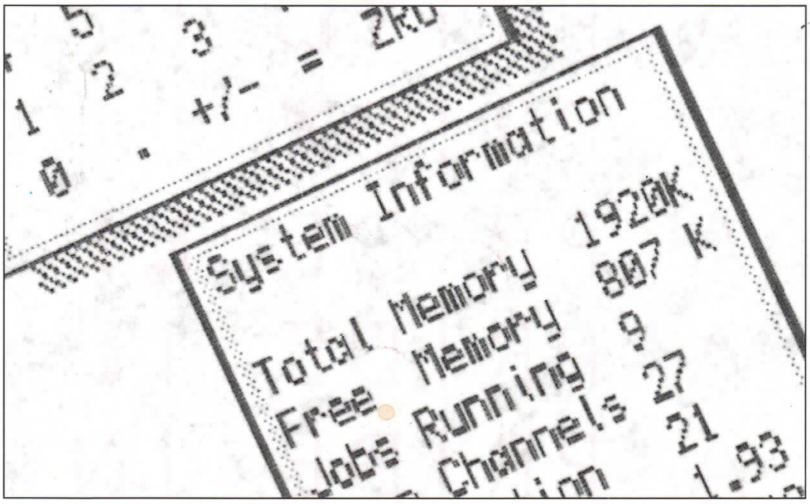
SINCLAIR QL WORLD

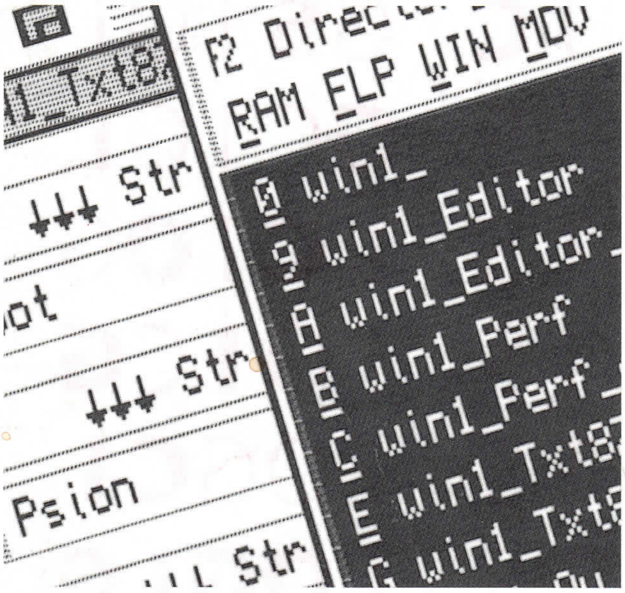
February 1993 Price £2.50

THE POINTER ENVIRONMENT



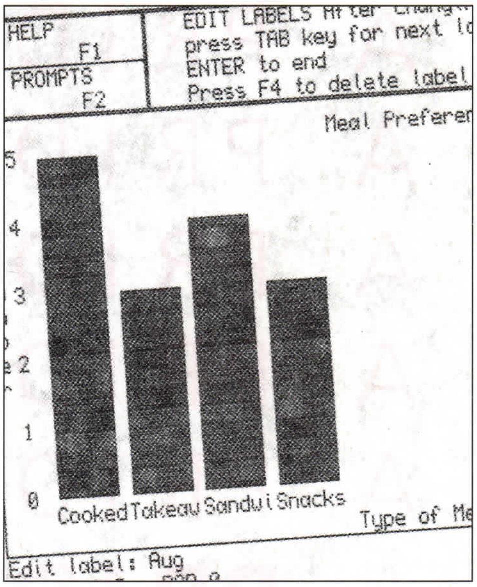
System Information
Total Memory 1920K
Free Memory 807 K
Jobs Running 9
Channels 27
21
...ion 1.93

FILE FINDER - REVIEW



0 win1_
9 win1_Editor
A win1_Editor_
B win1_Perf
C win1_Perf_
E win1_Txt8
G win1_Txt8

EASY WITH EASEL PART TWO



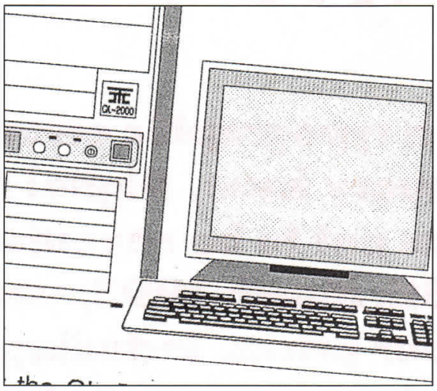
EDIT LABELS After chang...
press TAB key for next la
ENTER to end
Press F4 to delete label

Edit label: Aug

MACHINE CODE FOR BEGINNERS

```
LISTING 2

100 z=RESPR(28)
110 LBYTES flp1_LISTING1_code,z
120 OPEN#3,scr_512x256a0x0
130 CLS#3
140 CALL z
```



QL SCENE

ISSN 0951-9335 02
9 770951 933023

D IGIT**A** LP REC ISION

A **I**GITD L**P** REC ISION

A P**G**ITD LI **R**EC ISION

A PRI**TD** LI GEC ISION

A PRID**T** LI G**E**C ISION

A PRIDE **LI** GTC ISION

A PRIDE I**L** GTC ISIO**N**

A PRIDE IN **G**TC **I**SIOL

A PRIDE IN IT**C** G**S**IOL

A PRIDE IN ITS **G**CIO**L**

A PRIDE IN ITS L**C**I**O**G

A PRIDE IN ITS LO**I**C**G**

A PRIDE IN ITS LOG**CI**

A **PRIDE** IN ITS **LOGIC**

Due to a typographic error (not by Digital Precision) in the January issue, Digital Precision's New Year offer that was not to be repeated or extended - is Extended! Until 15th March 1993, you may deduct an extra 10% off all our prices, including even the existing Special Deals where you can get upto 25% off already! SUPER!! Don't hesitate: this offer DEFINITELY won't be repeated or extended again.

Contents

February 1993

8 **REVIEW - FILE FINDER**
A pointer-driven file-seeker takes Bryan Davies hunting.

10 **OPEN CHANNEL**
Blank pages! ... QL clock ... UnConquerered ... Newsagents ... Machine code ... Ancient history ... Damped off ... Lost files.

12 **QL SCENE**
International QL ... QL 2000 Falkenburg ... Qbits ... EEC have Philips monitors ... Screen Dazzler ... Canadian Club.

16 **SUPERBASIC IN ACTION**
Simon Goodwin continues with part two of Revive, the file retriever.

18 **EASY WITH EASEL - PART 2**
Henry Orlowski with the second part of his useful Easel series.

20 **TROUBLESHOOTER**
Bryan Davies immerses himself in readers' letters.

23 **THE NEW USER GUIDE - part 23**
The Keyword Index from PRINT to PROCEDURE.

27 **MACHINE CODE FOR BEGINNERS - PART 1**
This time, Alan Bridewell starts right at the bottom.

31 **DIY TOOLKIT**
Simon Goodwin continues with part 2 of Flexynet.

34 **THE POINTER ENVIRONMENT**
The long and the long of it - Ian Bruntlett introduces the QJump masterpiece - the start of a long conversation.

46 **MICRO ADS**

Coming Soon

Yet more on Qpac2 ... Archive statistics ... games! ... Abacus notes.

Editor
Helen Armstrong

Publisher
Mark Kasprowicz

Advertising Manager
Jim Peskett

Creative Director
John Stanley

Graphic Artist
Stevie Billington

Magazine Services
Linda Miller, Frances Maxwell,
Pauline Wakeling

**Sinclair QL World,
Published by Arcwind Ltd.
The Blue Barn,
Tew Lane, Wootton,
Woodstock,
Oxon. OX7 1HA
Tel: 0993 811181
Fax: 0993 811481
ISSN 026806X**

If you have any comments or difficulties please write to the editor and we will do our best to deal with your problem in the magazine, though we cannot guarantee individual replies.
Back issues are available from the publisher price £2.50 UK, £2.99 Europe. Overseas rates on request.

Subscriptions from:
Arcwind
The Blue Barn, Tew Lane,
Wootton, Woodstock,
Oxon. OX7 1HA
UK: £23.40
Europe: £32.90
Rest of World: £40.90

Reprographic Services:
Colorfields, Eaton Socon,
St. Neots.
Distributed by: Seymour Press Ltd.,
Windsor House, 1270
London Road, Norbury,
London, SW16 4DH

PERFECTION SPECIAL EDITION

POWER

PERFECTION SPECIAL EDITION has 253 (two hundred and fifty three) direct/menu commands (not counting options in sub-menus), plus 32 special characters (like Bold on) that can be inserted 'directly' plus intelligent (and now excellently documented) macros. Comparisons with other word processors on the subject of power are hence quite unnecessary.

EASE OF USE

Independent reports, customer feedback and published reviews (of its less able but still excellent predecessor, PERFECTION) leave one in no doubt as to which word processor is friendliest – PERFECTION SPECIAL EDITION, with its intuitive, silky handling. Uniquely, it has two operating modes, with both menus (visible or invisible – they even look like Quill's) and direct commands (for when you familiarise yourself with the system). Uniquely, both modes are 're-entrant' (so you can use any menu option or direct command while you are in the middle of performing another option or command – block handling, etc, becomes a dream). Uniquely, PERFECTION SE has fully automatic memory management, grabbing and releasing RAM instantly as your document grows or shrinks – programs without this don't take full advantage of the multi-tasking abilities of the QL! Uniquely, PERFECTION SE leaves you in the driving seat, not juggling things around 'underfoot' while you are typing. Uniquely, PERFECTION SE allows up to nine different documents to be handled simultaneously from one copy of the program – each with totally independent margin, tab, justification, control panel, etc, settings. Uniquely, each document can itself have up to six environment settings, each settable or recallable instantly with a single keypress combination. Each document can have any number (up to 500,000 on GOLD CARD) of candidate blocks! Each document can have two independent windows (of any depth, of any (but same) width across) 'on to' it, even with overlapping text – that allows you to edit in one place while viewing another, to compare 'before editing' with 'after editing' (you can arrange to have one window remain 'frozen' in time), etc. Uniquely, we realise how much faster it is to type in something like CTRL/SHIFT/F5 than (say) F3 D U – both involve three keys, but as the former doesn't require the keys to be pressed in just one specific order, or to be released in any order at all (together will do), it is in practice twice as fast as the latter, where no key may be pressed until its predecessor is released. Also, sequences like CTRL/T (top) and then CTRL/G (go to next occurrence of string in set direction) can be accomplished by holding down CTRL and then tapping T and G. Uniquely, by providing eight user-definable strips, PERFECTION SE allows you to cope with printers of the future, not just the printers that now exist – you can attach the strips to any printer features. Uniquely, PERFECTION SE's status lines give full information on all relevant global settings. And the manual has an index. Also, it has all the important bits at the front.

PC CONQUEROR GOLD SPECIAL EDITION – This terrific new product for QLs with 1.5 Mb or more makes your QL system into a PC. A well-equipped PC too, with about a megabyte of expanded RAM installed, and the ability to read, write and format SD/DD/HD/ED disks (the last by making them into pseudo hard disks). Disk performance is up to 5 times faster. Other performance is up to 55% faster than standard CONQUEROR on GOLD CARD. There are many extra features too – see our ads in June - September 1992 QLW for full details.

DR-DOS V 6.0 – The latest and most capable DOS of all!

QMATHS MATHEMATICAL SYSTEM PART TWO – An excellent complement to QMATHS, with loads of 'functionality' – fractals, function evaluation, terrain plotting, masses of maths & stats, etc.

QUICKLASER – The definitive output tool from PRO PUBLISHER to HP LaserJet II (or compatible) printers. Printed output quality subjectively exceeds that from any other QL product.

TRANSFER UTILITY SPECIAL EDITION – Does everything – 16 case change options, 14 types of sorting (multiple sorts possible), auto string translations, etc.

LIGHTNING SPECIAL EDITION GOLD CARD VERSION – See June-Aug 1992 QLW for details: optimal speed from GOLD CARD, ST/QL, THOR XVI. Free upgrade from the ROM SE version (return ROM + disk) if you are ordering something else at same time: if not, £10 charge.

SUPERB PRINT QUALITY & FLEXIBILITY

Uniquely, using the aforementioned automatic link, you can output PERFECTION SE documents using over a thousand fonts (a huge variety of styles and sizes, supplied on the PUBLISHER and TOOLBOX disks) on virtually any printer – from the humblest Epson RX80, Brother M1009 or Star LC10 (which are all single font machines when used with most word processors) to top-end lasers. *You are not limited to the fonts built into the printer!!* All PERFECTION SE **bold**/underlined/ *italics*/$^{\text{super}}$/$_{\text{sub}}$, etc, settings are preserved. Proportional spacing and micro-justification are automatic, even when you mix fonts of differing widths *and heights* (even on the same line), vary line spacings, etc. Uniquely, you are not trapped with one type of micro-justification (ie adding all the space between words, and using the predefined widths of characters as their separation) – with our system, you can vary (in 5% steps) the ratio of micro-spaces added between words to that added between characters (the latter in proportion to their *individual* widths). Ratios around 65%-35% – not the 100%-0% forced upon you by some other word processors – seem to give the most pleasing results. Uniquely, you are not limited to mere rectangular columns plus headers/footers – that's all the rest can do – you can output in any sequence to any number of frames (text flowing from one to the next), each of any shape – irregular polygons of up to 66 sides, circles, multi-column or part-column boxes (hundreds of types of borders, thousands of textures), doughnuts, wrap-around shapes, even re-entrant ones ('join-the-dots' type borders, even with intersecting edges) – all with micro-justification and proportional spacing! Look at the example on this page. Of course, if super-fancy effects (like wraparound windows and mixing different font widths on the same line while maintaining right justification) are not of the essence, PERFECTION SE's direct printer output is excellent with all your printer's capabilities supported.

THE FASTEST

For benchmarking, we've used a public domain version of the first book of The King James Bible, all fifty chapters of the book of Genesis. This came to **one hundred and forty pages**, well over **forty two thousand words** excluding headers and footers, well over **two hundred and twelve thousand characters** excluding justification ones, **fifty full chapters** and **one thousand five hundred and thirty three indexed verses**!! We didn't use a smaller file (as used to benchmark other programs) as PERFECTION SE's timings for most operations then become impossible to stopwatch (too fast!). The hardware used for all timings was GOLD CARD: speeds would be **further improved by over three times** using the ST/QL 030. Of course, LIGHTNING SE was used. File operations were to ramdisk: normal slave blocks would give identical times. All settings on **everything** were for maximum speed, except where indicated to the contrary – we do not force full speed upon you in operations like scrolling and global Search & Replace. PERFECTION SE's speed for these is switchable (at run-time and when configuring), as too great a speed may cause overshoot (with scrolling) or fatal alteration (if there is human error inputting the target or replace strings). Here are the benchmarks for this huge document:

Load 140 pages: 0.6 seconds (yes 0.6, not 6!) ☆ Import 140 pages: 0.6 seconds (yes 0.6, not 6!) ☆ Save 140 pages: 0.5 seconds (yes 0.5, not 5!) ☆ Export 140 pages: 0.5 seconds (yes 0.5, not 5!) ☆ Case-sensitive search from top for word at bottom: 0.4 seconds (yes 0.4, not 4!) ☆ The same, but case-insensitive: 0.5 seconds (yes 0.5, not 5!) ☆ Case-sensitive search backwards from bottom for word at top: 0.4 seconds (yes 0.4, not 4!) ☆ The same, but case-insensitive: 0.5 seconds (yes 0.5, not 5!) ☆ Automatic Search & Replace, in Fast (No Query) mode, of last 600 occurrences: 7.4 seconds (same length replace string); 7.7 seconds (shorter replace string); 10.5 seconds (longer replace string – longer time as we deliberately chose a high *density* of replaces to handicap PERFECTION SE into auto-managing memory – without causing any heap fragmentation, but still with only a 0.005 second overhead per replace!) ☆ Automatic Search & Replace in Slow ('Querying') mode: arbitrarily slow, typically 30 times slower – because we deliberately allow for human response time (in case you want to abort) before proceeding from one replace to the next. ☆ Scrolling 100 lines of text, up or down, by full-width screen page: 1.5 seconds ☆ Scrolling 100 lines of text on full-width screen, line by line, in slow (full) mode: 5.7 seconds (down)/5.8 seconds (up) ☆ As above, but in medium speed mode: 4 seconds ☆ The same, but in fast mode and default settings: 13.5 seconds to scroll through the whole massive document, averaging 0.23 seconds per 100 pages (!) – and this could be made up to ten times faster by reconfiguring PERFECTION SE ☆ Reformatting paragraphs, changing margins, justification, etc, of existing text: c5 times faster than predecessor ☆ Inserting (or undoing) emphasised, underlined, italics, superscript, subscript, 8 strips, 6 environment settings: Instant (i.e. immeasurable) ☆ Navigation to line or page or to top or bottom or to 8 markers or to highlights/blocks: Instant ☆ Setting new margins, justification, etc: Instant ☆ Deleting block of 100 pages: 0.3 (yes, 0.3 not 3!) seconds ☆ Copying/moving block of 100 pages (not just 10!), downwards or upwards: 3.4 seconds (yes, including all the time for automatic memory management and anti-fragmentation – other programs are light-years behind) ☆ Spellcheck as you type: Ten times faster than anyone can possibly type ☆ Spellcheck all 140 pages in the document using the 350,000 word Mega Dictionary: 3.9 seconds (20 'errors' – like 'pluckt'!) ☆ And using our tiny dictionary (well, tiny by our standards – large by comparison with most others): 5.1 seconds (566 'errors') ☆ Time taken to create user dictionary from the results of the second spellcheck (566 errors): 0.8 seconds to extract all 'errors' from document and clean document; 1.9 seconds to create a full user dictionary therefrom and also a sorted, duplicate-free wordlist file (for browsing) ☆ Spellcheck file (ASCII or native): Even faster. ☆ Print first 10 pages to file: 3.5 seconds. ☆ Change every occurrence in 140 pages of God to ***God*** in bold underlined italics, strip 8 – 9.5 seconds! ☆ Virtually everything else: instant.

For prices, see the coupon page of our ad. For more info, read our detailed QLW ads in early 1991 for PERFECTION, plus the extra features of the SE (well, about half of them) listed in the June-August 1992 issues. You can upgrade from the standard PERFECTION (or PLUS) to the SPECIAL EDITIONs for the difference in current price, plus £10: no manuals or dictionary disks to be returned – we'll send a supplement to the manual.

PERFECTION
PERFECTION PLUS

Perfection is the finest word processor available for any computer. We have received dozens of letters from happy users saying just this... and all of these letters were unsolicited. "Superb" was used most often.

Perfection manages to achieve all the sophistication of the most complex PC word processors while still using a user interface as friendly as Quill's. Perfection has a dual system of user control: menus while you are familiarising yourself with the program, and direct commands for the time when you feel ready for more adventurous things. The two systems can be used interchangeably and even simultaneously. Even more exciting – both systems are iterative. In case you don't understand what this means, let us give you an example: suppose you wished to move a block of text using the menus. You would choose Block Move (yes, it is right in the first menu) and the screen would then tell you to move your cursor to the start of the block. On most word processors you would have to navigate manually to this position: indeed, on many of them (Quill included) only a subset of the normal navigation commands would be available. On Perfection, not only can you use all the manual navigation commands (viz all 28 permutations of CTRL, ALT, SHIFT and the arrow keys!) but in addition you can use direct commands like GoTo Line or Page or any of eight markers. Even more amazingly, you can use Search (either as a direct command or from the menus) *even though you are already 'within' a menu option.*

Perfection has about 200 commands, but the layout of menus and the choice of keys for the direct commands makes it very easy to master. Though a 100+ page manual is provided (with all the important bits right at the front), you should only need to consult it for specialised operations like macros.

Even if speed is not particularly important to you, we assure you that Perfection's lightning performance will enable you to use the word processor in sensible ways that you would not have dreamed possible before. For example, scrolling 100 pages or so is accomplished so quickly using the normal navigation commands that you do not need to bother using a menu option to do the move. Spellchecking, assuming you have Perfection Plus, is accomplished virtually instantly: to spellcheck this whole ad (all the pages) would take under 1.5 seconds... Searching (you can switch case sensitivity, as well as equivalences between tabs, soft spaces and hard spaces) is at the rate of about 100 A4 pages per second.

Moving from one word processor to another is usually very traumatic. With Perfection, this will not be the case. Not only can Perfection read in Quill _doc and _exp files directly (you do not even need to tell it they are Quill files!) but it can make direct and immediate use of your existing Quill printer driver. File re-export is also possible.

Perfection is truly WYSIWYG: this means that bold appears bold on screen, italics appear as italics, underlined as underlined, and so on. Of course, your printer may have functions we do not know about (upside down?). To deal with these, Perfection provides a number of on-screen shaded strips: these can be attached to any printer function you wish, and will not upset justification as a translate would. Of course, translates are provided as well!

A variety of statistics on the document being processed are available: some of them are on view all the time, the rest can be toggled to instantly. Not only is there a word count, but also page, line, character and special character (like Superscript Off) counts. There are also a dozen status indicators, letting you know whether you are in Insert or Overwrite mode, whether a block is defined, whether interactive spellchecking is enabled etc. Current line (from top as well as within page) and column positions and character codes are also available.

A terrific feature of Perfection is the dual screen mode. You can view one part of the document while editing another. The sizes of the two windows are themselves adjustable, both in real-time or via the configurator. We should devote more space to the configurator: however, it must suffice to say that everything that could be dynamically set within Perfection may also be preset with the configurator. The configurator can, for example, allow you to select any of 256 colours for any of a dozen parameters (like paper colour, border colour, status window ink and paper colour etc).

Perfection is fully multitasking without need for any external accessory: however, if you already use QPAC or Taskmaster or similar and are happy, you may go on doing so.

There is absolutely no way that we can prepare you for the quality 'feel' of Perfection. We have a great deal of experience using PC word processors costing many hundreds of pounds: with absolutely no exception, Perfection is far easier to use and master.

So if you thought Perfection was unattainable, you have a very pleasant surprise coming to you!

LIGHTNING SPECIAL EDITION
LIGHTNING

These programs accelerate QL operation by up to 10x (2x –4x is typical) without having any adverse effect whatsoever on compatibility or anything else. Lightning SE is typically 40% faster than the standard version. This acceleration is totally independent of, and in addition to, any speed-up obtained by hardware means. So if you have Gold Card, your need for Lightning SE is just the same as if you had only an unexpanded QL – Lightning SE will accelerate both by the same ratio.

The Lightning programs achieve their acceleration by automatically paging out sections of the QL's operating system and replacing these with optimal, concise code written by us.

Lightning installation is a completely automatic and one-off: no knowledge of computing or programming is required. Once installed, Lightning can be completely forgotten about – you will soon get used to the superb speed! Knob twiddlers are catered for too.

Lightning technology is not built in to any of our other programs. Perfection users (as well as users of all other QL software) should therefore use Lightning all the time.

In summary: if you do not have Lightning, you are wrong. Buy this one **FIRST OF ALL!**

PROFESSIONAL PUBLISHER

The Professional in Professional Publisher refers to the quality of output from that program, and is not meant to suggest any complexity of operation. Few programs are as easy to use as this one: > 99% of users will be able to do without using a manual!

Professional Publisher is by far the best DTP program for the QL. It is fully compatible with Perfection, Editor, Quill, Eye-Q & the ASCII editors. It allows you to *both* create and import *both* text and graphics. Text can be 'poured' into boxes of any shape , size and number, automatically maintaining justification and hyphenation settings. So flowing text around graphics is a doddle.

Professional Publisher is supplied with a generous selection of fonts of various sizes, as well as clip art.

Justification is by pixel, not by character. This gives a much smoother effect.

It is pointless for us to try to list all of Professional Publisher's features – we would end up filling half the magazine! We will concentrate on just a few 'points': Professional Publisher is extremely precise, performing all its computations accurate to a small fraction of a millimetre. All its features can be preset by you using its configurator, ruling out the need for repetitive key strokes.

The program is extraordinarily versatile while remaining intuitive in its user interface. Buy it!

PROFESSIONAL PUBLISHER
TOOLBOXES

Toolbox I is an excellent collection of high definition fonts, clip art and utility programs for Professional Publisher. While the fonts supplied with Professional Publisher are excellent, many users will feel the need for a wider range of typefaces and styles.

Toolbox II starts where Toolbox I leaves off, providing an even better – and different – font collection.

The two Toolboxes complement each other and are available together at a special price.

FONT ENLARGER
GRAFIX

Font Enlarger does exactly what you would expect it to from its name. While Professional Publisher is also capable of enlarging fonts, it does them 'on the fly' and consequently is not able to remove the jaggedness caused by magnification. Font Enlarger is much cleverer, and enhances detail without any step effect.

While the built-in printer driver for Professional Publisher is excellent with 9-pin printers, it is not optimal with 24-pin or laser printers. Grafix is.

DIGITAL PRECISION LTD

EYE-Q
ULTRAPRINT

Eye-Q is the finest graphics program for the QL. While there may be other graphics programs with a few more features, no other program comes anywhere close to Eye-Q in sheer enjoyability. Eye-Q develops a pleasurable tactile relationship with you, and makes you feel like an artist (even if you aren't). Eye-Q graphics can be read in by Professional Publisher, and the latter's pages can be exported to Eye-Q (using Toolbox I). Everything in Eye-Q is menu-driven and there is context-sensitive help.

While Eye-Q has its own printer driver, Ultraprint allows you 22 distinct styles/sizes of printer output. The reasoning is that the scale of gradation suitable for pictures is probably unsuitable for text or line drawings.

PC CONQUEROR
SOLUTION

PC Conqueror makes your QL into a PC-compatible machine,automatically. It does this by software means only, so there are no screws to undo or wires to fiddle with. Your QL stays a QL too.

Why, might you ask, should you wish to make your QL into a PC-compatible? The reason is simple: you may wish to run the same programs at home as you do at work. Alternatively, you may wish to tap into the vast storehouse of PC software of every type and description you could imagine.

Using PC Conqueror could not be easier. Just boot up your machine with the PC Conqueror disk in floppy 1 and within 10 seconds your QL will be transformed into a PC that is just waiting to be switched on. From this point on you will do exactly the same as you would if you were running a 'real' PC – this means putting a DOS disk (any version) into one of your drives and pressing a key. If you do not already have legal access to a copy of DOS, we can provide you with one at reasonable cost (see our price list).

PC Conqueror runs as fast as it is possible for a PC emulator to run: we have used all our skills to make it work quickly. Of course, you can make the emulation must faster by using Gold Card and Lightning SE. With this combination, you should get speed noticeably better than that of a PC XT.

PC Conqueror allows you to fine-tune the operating environment of the PC in order to improve performance. If you get a hard disk or other high capacity floppy system, you can utilise part or all of it as a PC hard disk.

PC Conqueror occupies under 80K and leaves 667K free for DOS when run on a Trump Card. This is more than you will get on a 'real' PC.

Solution does what Conqueror does but is about half as fast and is not quite as compatible.

SPELLCHECKER
MEGA DICTIONARY

Spellchecker is what makes Perfection into Perfection Plus. We have made it available as a separate item for two reasons: (a) to allow Perfection owners to add it later (b) to allow users of other word processors to benefit from the very best in spellchecking technology.

Spellchecker is supplied complete with three dictionaries of differing sizes as well as a system for building, reviewing and maintaining user dictionaries.

Spellchecker's ultimate accessory is the Mega Dictionary, which gives the user a vocabulary of over 350,000 words!

3D PRECISION CAD SYSTEM

This program allows you to manipulate shapes and figures in 2D and 3D at a speed that will leave you breathless. Irrespective of whether your interest is in CAD, in animation or in just having fun, this program should not be missed. You can output to plotters directly from it, or alternatively create graphics screens to be manipulated and output by Eye-Q, Ultraprint or Professional Publisher.

SUPER SPRITE GENERATOR

SSG moves things about the screen very fast and very smoothly, without flicker. Sprites can have up to 16 frames.

MEDIA MANAGER SPECIAL EDITION
MEDIA MANAGER

Media Manager Special Edition (MMSE) is a program to be used both when things have gone wrong as well as when things are perfectly OK. It allows for automatic, semi-automatic and manual correction of a huge variety of disk and tape problems. It allows you to explore disks and tapes to your heart's content, producing all sorts of different diagnostic reports. MMSE is very simple to operate, being menu-driven and assuming no degree of computer knowledge whatsoever.

MMSE also allows you to tidy, catalogue, sort and order your disks and cartridges.

The standard Media Manager is both less powerful and less user-friendly, but manages to work on an unexpanded QL.

Both programs allow for data transfer between PC and QL. With MMSE, this transfer is at file and directory level, is bi-directional and is completely automatic.

SPECIAL DESKTOP PUBLISHER
DESKTOP PUBLISHER

These programs are quite primitive compared to Professional Publisher. However, if you have not experienced that program as yet, you will find both of these very competent. Both are capable of producing excellent results. The cheaper one has fewer features but is able to run on smaller systems.

EDITOR SPECIAL EDITION
THE EDITOR

With the sole exception of Perfection, this is the best word handling system on the QL. Editor's features include an unrivalled degree of programmability and the ability to cope with the entire 256 character ASCII set. The Special Edition has enhanced document-type facilities, including column blocks and on-screen page break displays. Neither program is suitable for computing novices. Until Perfection, Editor Special Edition would have been our 'Desert Island Program'.

Editor SE can do a few things that Perfection can't, so the ideal combination is to have both (they are compatible at file level and can multitask). If you order Editor SE at the same time as Perfection, you can have Editor SE at half price.

PROFESSIONAL ASTROLOGER
PROFESSIONAL ASTRONOMER

The Astrologer program teaches you Astrology from scratch and enables you to automatically produce text narrative on personality delineation, year-to-year and minute-to-minute life predictions, compatibility interpretations and so on. Whether or not you believe in astrology – indeed, especially if you do not – this program is one that you cannot afford to have. You can tailor the readouts (both in terms of quantity and what is said) to your own particular requirements. The amount of fun you can have with this program is endless. Do not blame us if you start believing in astrology, though!

Astronomer is an extremely fast and accurate solar system calculator, with planetarium views, planet faces, eclipses, cinerama display etc..

TURBO BASIC COMPILER

Turbo is the finest BASIC compiler for the QL and arguably the finest BASIC compiler for any computer!

Turbo automatically converts working BASIC programs into optimised machine code, usually with no need for human intervention. The benefits of this conversion are vastly enhanced running speed (as well as much faster loading, encryption and automatic bug fixing for a variety of QL interpreter oddities). Typical speed-up is 40x – 100x.

Turbo is provided with a 200 command toolkit, adding many useful commands to BASIC. Most of these commands will be of immediate use to the programmer, whether he is a novice or an expert. There are commands to load strings and floats into RAM, and to extract them automatically; to search memory and to move its contents; to control jobs and change their priorities, manage pipes, allocate and deallocate memory, to control both rubber and virtual arrays, to present INPUT with an editable default, to have random access to files and much more.

TOOLKIT III

Toolkit III starts where Toolkit II stopped, adding about 60 new commands and enhancing many existing dual functions. Toolkit III is available either on disk or on ROM, and works whether or not you have Toolkit II.

Toolkit III commands can, with only a couple of exceptions, be compiled using Turbo.

QFLICK CARD INDEX

All QL owners have a copy of Archive, supplied free with the QL. While Archive is competent, it is very hard to get to grips with and is not particularly fast. QFlick presents a very convenient alternative – a snappy, simple-to-use, pointer-controlled card file database. You can move data between QFlick and Archive in either direction.

QFlick is not itself programmable but we document its data structure and give guidance on how to program it using Turbo.

ARCHDEV + RTM
DATABASE ANALYSER
ARCHIVE TUTORIAL
NAMES + ADDRESSES
MAILMERGE
DAT–APPOINT
SEDIT
SCREENPRINT
RECOVER

This suite of utilities will greatly enhance your use of the Archive database system.

Archdev + RTM is a straight replacement for Archive: it gives enhanced speed, greater workspace and a much cleaner boot-up. All your existing applications will work.

Database Analyser provides very fast and comprehensive statistics about your Archive databases.

Archive Tutorial proceeds systematically through the whole philosophy and grammar of Archive, providing you with expert and patient guidance.

Names + addresses, Mailmerge and Dat-Appoint are ready-to-run, off-the-shelf Archive applications, providing an address database, mailmerging and appointment diary respectively. You now have no excuse not to use Archive.

SEdit allows you to create and edit screen format files in Archive. Screenprint allows you to print them out.

Recover allows you to get back lost Archive databases, created when you switched off the computer without properly exiting from Archive.

XREF
SUPERBASIC MONITOR
BETTERBASIC EXPERT SYSTEM

XRef analyses the structure of a BASIC program, providing detailed reports on things like variable usage, what calls what, dynamic call hierarchy of procedures and functions, and so on.

SuperBasic monitor actually monitors and reports on the performance of BASIC programs as they run under the interpreter.

BetterBasic analyses and automatically corrects structural flaws in your programs and allows you to customise things like indentation, number of statements per line, filtering out of noise words, etc.

The three programs together provide a matchless diagnostic and auto-correcting facility for BASIC programs.

TRANSFER UTILITY

This program copies files at high speed between devices, performing translates as it goes along. Ideal for all sorts of applications, including transfers from microdrive to disk.

QMATHS SYSTEM

This is an incredible mathematical compendium for the QL. Pride of place goes to the symbolic problem solver: this can solve equations, simplify expressions, factorise, expand, etc, all symbolically. If you could sneak this one into a maths examination, you would have a formidable ally. QMaths knows about all the algebraic operators, powers, roots, brackets, trigonometry, matrices, determinants, vectors, factorials, permutations, combinations, binomials, exponentials, logarithms, hyperbolics, inverse functions, infinite series including Taylor & Maclaurin expansions, complex numbers, conversions, Fourier series, and lots of calculus: both differential and integral, including integration by parts and definite integrals. QMaths optionally displays its workings and comes with a superb interactive tutorial.

The package also contains an interpretive, fractal, image-generating language with loads of beautiful fractal programs supplied for you to use and edit – no programming skill is required.

There is also a multiple precision floating point maths package, giving calculations at precisions up to over 600 decimal digits of accuracy.

There is even more to this system, but we think we have told you enough.

QMON MACHINE CODE MONITOR

The latest version of Tony Tebby's superb monitor: an absolute must for those who really want to know what is going on in the QL. No other machine code monitor even comes close.

Do not confuse this program with SuperBasic monitor, which monitors SuperBasic, not machine code.

COMPARE

This program compares files – data or program – at colossal speed. Where a mismatch is detected, the relevant areas are highlighted and you can shuffle, displace and align very easily.

CASH TRADER WITH ANALYSER
PAYROLL

Cash trader with Analyser is an accounts system designed by businessmen and not by wretched accountants! Consequently, it has excellent reporting and management facilities, and is very flexible. It is aimed primarily at the layman, probably a sole trader running a small or medium sized business. All the features you would expect – including audit trail – are present.

Payroll is a reasonably flexible system designed to automate the payroll function in small businesses.

Both programs are configurable, with editable defaults letting you adapt the programs from year to year.

HARDBACK WITH FINDER

This is the ultimate hard disk backup and management utility, with all the sophisticated features you could want. User dialogue is via overlapping pop-up windows – the whole program just feels right. It is possible to scan the disk at great speed, too.

DISKTOOL WITH QUICKDISK

This permits you to add password protection to disks, to optionally increase disk storage capacity on DSDD drives by 36K and to increase speed of access by as much as 30%. All this is done while maintaining full compatibility. Automatic file management is also provided.

DIGITAL C SPECIAL EDITION
DIGITAL C

These are extremely fast and efficient C compilers, complying with and surpassing the Small C definition. The Special Edition goes much further, including support for structures, pointers, long pointers, >64K code size, direct access to QDOS traps, etc. The Special Edition C generates code that runs about twice as fast as the other.

SPECIAL DEALS

5% off total if you buy 2 programs/upgrades;
10% off 3; 15% off 4; 20% off 5; 25% off 6+
Upgrades cost difference in price + £10
Non-UK Europe add 5%, rest of world 10%

For full terms and conditions, please refer to any of our QL World ads from Jan-Nov 1990, or write in including a SAE

CPORT IMPROVED VERSION

A brand new CPORT system, enabling you to rapidly convert your SuperBASIC programs into C (ANSI or Lattice). The new (October 1992) version is now as close to being fully automatic as makes no difference – you must get it!

Owners of our earlier CPORT versions should return disk + SAE for a free upgrade.

SUPERFORTH COMPILER WITH REVERSI

Forth is the most logical computer language. This compiler produces multitasking code. The manual teaches you Forth-83 from scratch.

IDIS SPECIAL EDITION IDIS

These intelligent disassemblers make the otherwise terrifyingly complex task of understanding other people's machine code programs absurdly easy. The SE version, which has a higher hardware requirement, sorts out some routines, replaces addresses with names, untangles data from code and much more.

QKICK FRONT END SYSTEM

This is a simple, easy-to-master, pull-down menu controlled multitasking front end. QKick runs in the background and can be called up at any time. It provides you with notepads, sophisticated file/sector/RAM handling, backing up facilities, a clock, diary, calculator, mini-database and so on.

ADVENTURE CREATION TOOL SPECIAL EDITION

ACT is a must for every programmer. The name of the program is misleading, insofar as it has capabilities far beyond the 'mere' creation of adventures. ACT has utilities providing animated graphics, data compression, language design, parsing, maps, object-oriented control etc. If all you want to do is generate adventures, though, you do not need to be a programmer to use it. This is a purchase you will never regret.

PEDIT

A fast, modern and capable printer driver for the programs bundled with the QL.

MICROBRIDGE

Superb contract bridge bidder (ACOL etc) and player, using millions of random but reconstructable hands. Microbridge also includes a state of the art interactive bidding tutor and a clear instruction manual. There is nothing like this anywhere else!

SUPER ASTROLOGER

A very cut-down version of Professional Astrologer – still great fun, though!

SUCCESS CP/M EMULATOR

Allows your QL to run CP/M programs at great speed.

	£	
3-D PRECISION CAD SYSTEM	49.95	d
ADVENTURE CREATION TOOL SPECIAL EDITION	49.95	e
ARCADIA GAME	9.95	f
ARCHIVE DEVELOPMENT SYS + RUN-TIME MODULE	29.95	a
ARCHIVE TUTORIAL	19.95	a
BETTER BASIC EXPERT SYSTEM	24.95	a
BLOCKLANDS GAME	9.95	f
CASH TRADER v3.3 + ANALYSER	99.95	c
COMPARE FILE UTILITY	19.95	a
COPY UTILITY FOR BACKING UP	9.95	a
CPORT BASIC TO C TRANSLATOR WITH C68K COMPILER	99.95	c
CPORT BASIC TO C TRANSLATOR	89.95	c
DATABASE ANALYSER FOR ARCHIVE	19.95	a
DAT-APPOINT APPOINTMENT DATABASE SYSTEM	19.95	d
DESKJET/LASERJET DRIVER FOR PRO PUBLISHER	19.95	c
DESKTOP PUBLISHER SPECIAL EDITION	39.95	c
DESKTOP PUBLISHER	24.95	d
DIGITAL C COMPILER	29.95	a
DIGITAL C SPECIAL EDITION	49.95	a
DISKTOOL WITH QUICKDISK	19.95	b
DROIDZONE GAME	9.95	f
DR-DOS v6.0 STATE OF THE ART PC OPERATING SYSTEM	79.95	e
EDITOR SPECIAL EDITION	49.95	d
EDITOR	29.95	a
EYE-Q GRAPHICS SYSTEM	39.95	a
FONT ENLARGER	19.95	e
GAMES COMPENDIUM (ALL FIVE GAMES)	29.95	a
GRAFIX 24-PIN UNIVERSAL PRINTER DRIVER	14.95	e
HARDBACK + FINDER HARD DISK MANAGER	49.95	c
IDIS INTELLIGENT DISASSEMBLER	24.95	a
IDIS SPECIAL EDITION	39.95	d
LIGHTNING ACCELERATOR	24.95	a
LIGHTNING SPECIAL EDITION ACCELERATOR	49.95	h
LIGHTNING SPECIAL EDITION FOR GOLD CARD	39.95	a
MAILMERGE DATABASE SYSTEM	19.95	a
MEDIA MANAGER SPECIAL EDITION	49.95	d
MEDIA MANAGER	24.95	a
MEGA DICTIONARY	29.95	g
MICROBRIDGE	39.95	d
NAMES AND ADDRESSES DATABASE SYSTEM	19.95	d
PAYROLL SYSTEM	49.95	c
PC CONQUEROR GOLD SPECIAL EDITION WITH DR-DOS v6.0	179.95	g
PC CONQUEROR GOLD SPECIAL EDITION	99.95	g
PC CONQUEROR WITH DR-DOS v6.0	139.95	e
PC CONQUEROR	59.95	e
PEDIT PRINTER DRIVER FOR XCHANGE	19.95	a
PERFECTION PLUS SPECIAL EDITION WITH SPELLCHECKER	139.95	c
PERFECTION PLUS WITH SPELLCHECKER	99.95	e
PERFECTION SPECIAL EDITION WORD PROCESSOR	99.95	c
PERFECTION WORD PROCESSOR	59.95	d
PERFECT POINTER TOOLS	29.95	a
PROFESSIONAL ASTROLOGER WITH ASTRONOMER	69.95	a
PROFESSIONAL ASTROLOGER	59.95	a
PROFESSIONAL ASTRONOMER	29.95	f
PROFESSIONAL PUBLISHER TOOLBOXES (PARTS ONE & TWO)	49.95	e
PROFESSIONAL PUBLISHER TOOLBOX PART ONE	29.95	e
PROFESSIONAL PUBLISHER TOOLBOX PART TWO	29.95	e
PROFESSIONAL PUBLISHER	89.95	c
QFLICK CARD INDEX SYSTEM	29.95	a
QKICK FRONT END SYSTEM	24.95	a
QMATHS MATHEMATICAL SYSTEM PART ONE	69.95	c
QMATHS MATHEMATICAL SYSTEM PART TWO	59.95	c
QMATHS MATHEMATICAL SYSTEM (PARTS ONE & TWO)	99.95	c
QMON MACHINE CODE MONITOR v2.05	39.95	a
RECOVER ARCHIVE UTILITY	19.95	a
REVERSI GAME	9.95	f
SEDIT + SCREENPRINT ARCHIVE UTILITIES	39.95	a
SOLUTION PC EMULATOR WITH DR-DOS v6.0	109.95	e
SOLUTION PC EMULATOR	29.95	e
SPELLCHECKER WITH DICTIONARIES	49.95	d
SUCCESS CP/M EMULATOR	49.95	b
SUPERBASIC MONITOR	24.95	a
SUPERFORTH COMPILER WITH REVERSI	39.95	a
SUPER ASTROLOGER	24.95	f
SUPER BACKGAMMON GAME	9.95	f
SUPER SPRITE GENERATOR	29.95	a
TOOLKIT III WITH ROM	49.95	h
TOOLKIT III	29.95	a
TRANSFER UTILITY SPECIAL EDITION	29.95	b
TRANSFER UTILITY	9.95	b
TURBO BASIC COMPILER + TOOLKIT	79.95	a
ULTRAPRINT SCREEN DUMP UTILITY	19.95	a
XREF SUPERBASIC PROGRAM ANALYSER	29.95	a

NEW!

KEY>>>>>>> Available either on cartridge or disk	*a*
Available only on disk	*b*
Minimum 512K exp:only available on disk	*c*
Minimum 256K exp:either cartridge or disk	*d*
Minimum 256K exp:only available on disk	*e*
Available only on cartridge	*f*
Minimum 1.5Mb RAM:only available on disk	*g*
As well as cartridge or disk, you get a ROM	*h*

VISA

Send to: DIGITAL PRECISION LTD, 222 THE AVENUE, CHINGFORD, LONDON E4 9SE or phone 081-527 5493

Please rush me: ....

Name: .... Address: ....

.... Postcode: ....

Encl: CHEQUE/VISA/ACCESS/MASTERCARD/PO/MO/CASH for £ ....

Card No: .... Expiry:............ Signed: ....

Delete as appropriate: (QL/THOR/STQL)(MDV/3.5"/5.25")(Disk=360/720/1440/2880/6400 sectors; nb 1 sector=0.5K)(If 720 sectors: SSDD/DSSD)

File Finder

Bryan Davies tries a Pointer Interface-driven File Hunter from a major supplier.

INFORMATION
Program: File Finder
Price: £19
Supplier: Software87, 33 Savernake Road, London NW3 2JU.

Software 87 has taken on the selling in the UK of several programs developed in Germany, and distributed there by Jochen Merz. From the samples sent to QL World, it can be taken that these programs meet a decent standard of presentation. They each have good, well-printed A5-size instruction booklets, and the instructions themselves are generally clear and sufficient. The programs all have in common the QJump Pointer Interface, which is as would be expected, bearing in mind Jochen's obvious enthusiasm for Tony Tebby's graphical interface for the QL. A mouse is not essential, but does make use of the program easier (for the novice, at least). Alternative key combinations are available for some functions and they make operation faster and simpler.

As noted in previous reviews, prior knowledge of the workings of the Pointer Interface is always helpful when getting started with a new program that utilises it, and this certainly applies with File Finder. There is a brief section in the instructions devoted to explaining the PE to those who are new to it, but the notes do no more than give some idea of what keys to hit to get action. The instructions are spread over a total of eight pages, most of them consisting largely of "white space". There is really not a lot to explain, however, as the sole function of the program is to locate "lost" files.

Introduction

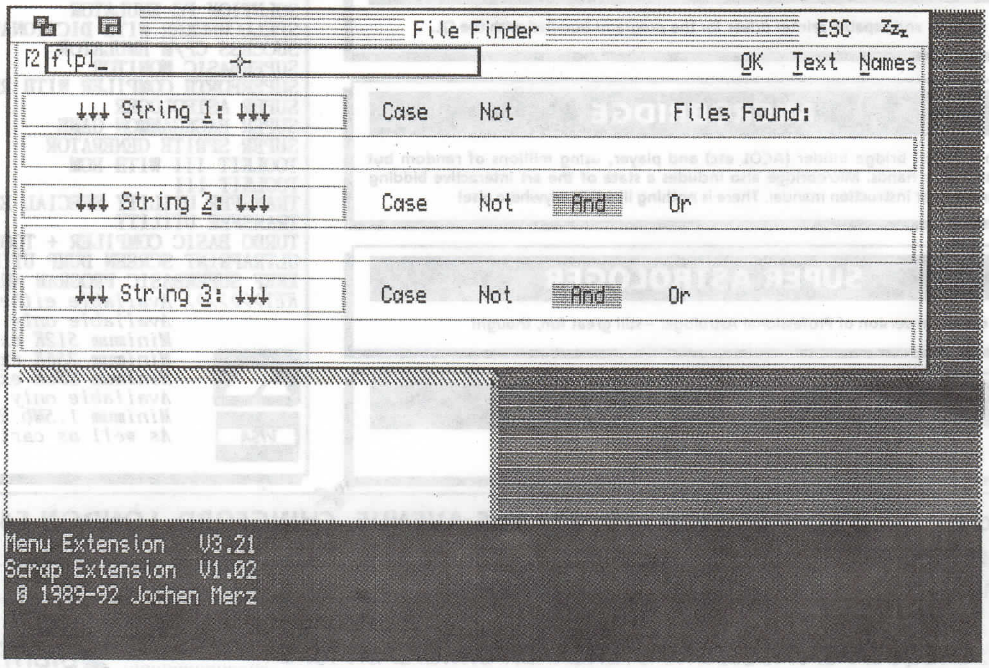
User instructions generally start with an exhortation to make a back-up copy of the program, but the subject does not get a mention in the FiFi instructions. The program writer, Wolfgang Lenerz, clearly feels he should not insult users by telling them what they already (should) know. He starts straight off with a short and adequate statement of what the program is for - "FiFi is a retriever... It will search for, and retrieve, all files containing one or several strings (up to three). These strings are combinable..." The program disk supplied for review contained duplicate sets of files, and had German, French and English versions. A working copy can be made in the usual fashion, with the COPY or WCOPY command, or similar.

The program is started with the EXEC (or equivalent) command, and can be activated from a Hotkey. Leastways, that is what the instructions state. In fact, there is a boot file which loads extensions and then tries to EXEC the file Fifi_exe. Unfortunately, there was no file of this name on the review disk. The impression given was that you are left to figure out that the _exe has been missed off the program file; the version English users will want has _English tagged on the end, not _exe. French users face the equivalent errors. It may be that the program disk on which the review was based differed from what will be supplied to users in the U.K., as there is no obvious reason to sell a multi-lingual version here. Once the boot file was sorted out, the program started quickly and displayed the screen shown in the first illustration.

In the usual PE fashion, the pointer - a cross in the case of FiFi - can be moved by cursor keys or mouse to the desired screen area. Once there, the function is marked or activated, or a text box made active, by pressing the Space Bar or Enter key, as appropriate. Where a letter is underlined, that letter can be keyed to mark/activate the corresponding function; this is the fastest way of working. Instructions are not really necessary when the opening screen is displayed; you specify the drive (and directory if applicable), the text string or strings to be searched for, and the conditions to be imposed on the search(es), then move the pointer to OK and press Space or Enter (as usual, it is acceptable in some circumstances to press either, but sometimes only one).

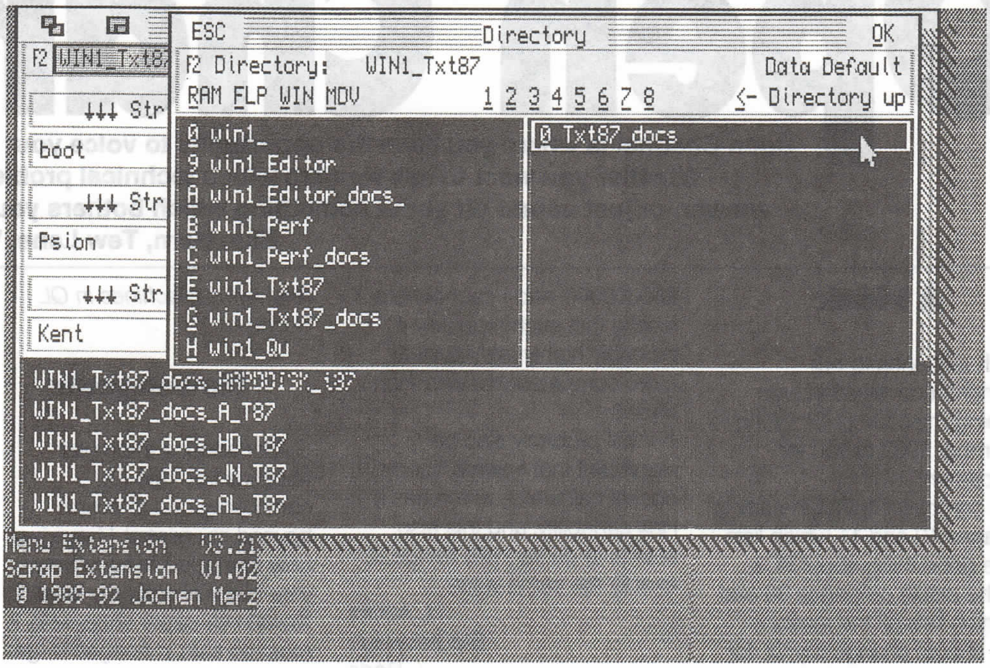
Speed search

Speed is naturally important in a search operation and there does not seem to be any problem here. On a Gold Card JS system, a search for files containing three individual words, one with case being significant, took just a few seconds on a double-density disk with about a dozen files on it, and all similar searches tried produced results within the same space of time. Making life more difficult for the program, a search for a file which was a page created with Professional Publisher (not basically a text file) containing three strings of alphabetic characters, that was one of several hundred on a Miracle 40 MB hard disk drive, resulted in nothing being found, even though the strings were



```
Menu Extension   V3.21
Scrap Extension  V1.02
 @ 1989-92 Jochen Merz
```

known to exist. Guessing at the reason for this, there might be a limit to how many files FiFi would search, and the root directory of the hard disk perhaps had far more files than this limit; alternatively, sub-directory names could be giving it trouble. The results of a search are displayed in a window which opens up below the main search window; the full directory and file name are shown for each file which matches the search criteria (see the second illustration).



```
  F2 WIN1_Txt87   ESC               Directory                  OK
                  F2 Directory:   WIN1_Txt87          Data Default
  ↓↓↓ Str         RAM FLP WIN MDV    1 2 3 4 5 6 7 8  <- Directory up
  boot            0 win1_                    0 Txt87_docs
                  9 win1_Editor
  ↓↓↓ Str         A win1_Editor_docs_
  Psion           B win1_Perf
                  C win1_Perf_docs
  ↓↓↓ Str         E win1_Txt87
  Kent            G win1_Txt87_docs
                  H win1_Qu
  WIN1_Txt87_docs_HARDDISK_i87
  WIN1_Txt87_docs_A_T87
  WIN1_Txt87_docs_HD_T87
  WIN1_Txt87_docs_JN_T87
  WIN1_Txt87_docs_AL_T87
Menu Extension    V3.21
Scrap Extension   V1.02
© 1989-92 Jochen Merz
```

One disconcerting thing was the behaviour when nothing matching the search criteria was found. There did not appear to be any problem with floppy disk searches, and the program displayed the word "nothing" quite quickly, but "Nothing" was what happened in some hard disk searches. As noted in the previous paragraph, the program appeared to stumble when confronted with a large number of files. It happily found files in sub-directories containing a few dozen files, and also found file names in the well-filled root directory, but it did not find any of the files in the root directory of the hard disk when asked to locate them by text strings. The screen showed no change from when the search was initiated and you were left wondering whether or not the large number of files was making the job a long one. The situation was not helped by the lack of an activity indicator on the Miracle hard drive. The pointer disappeared, but Ctrl-C was still active. It was sometimes possible to switch back to SuperBasic, but FiFi itself appeared to be dead. In fact, on some occasions, the FiFi screen disappeared, or the QL reset when Ctrl-C was keyed.

Discrimination

There would be nothing particularly interesting about a program which looked for specified strings in every file on a device without discriminating between one file type and another. FiFi provides two very useful options - Text and Names. Selecting Text before OK causes the program to exclude executable files from its search. Selecting Names first causes it to search only file names, not the files themselves. Both selections speed-up searches, in proportion to the ratio of executable to text files on the searched medium.

The search conditions are fairly obvious, Case, And, Not and Or being standard terms users are likely to have come across often before. However, the Or deserves comment. It is not an Exclusive Or; that is, specifying that a file should contain one string Or another means that it is acceptable for either string - or both of them - to appear in a file. The Not indicates that a string should not be found in a file. Strings and conditions are treated in sequence. You must insert String 1 if you want to insert String 2, and likewise with String 3. Returning to String 1 and modifying it, after setting Strings 2 and 3, will cause all strings to be deleted, the assumption being that the user wants to start the insertion process over again. The conditions apply individually to the three strings; you could select, for instance, one Not, one Or and one And, or three Nots.

Directories

Directories are not something which concern most users, but they are very important to the hard disk user, and can also matter to Gold Card users with HD or ED drives. Pressing Enter when the directory box (to the right of F2 in the first illustration) has been highlighted by the pointer brings up a dialogue box listing available devices and directories, as shown in the second illustration. Initially, the directories listed are ones which are unlikely to correspond to the user's own names, but they can be configured. Using the Directory function brought up the confusion of the supplied set-up again, as the text in the box was German!

It is desirable to be able to view a file which has been found in the search, and FiFi provides this facility. Moving the pointer to any of the files on the "found" list, and then pressing Space causes the file name to be inserted into the Hotkey buffer and also selected; pressing Enter causes the location of the last search string in the highlighted file to be displayed. The window displaying the chosen file has << and >> icons, and moving the pointer to these and pressing Enter causes the previous or next, selected file to be displayed (keying the underlined < or > did not have any visible effect, however). Files have to be selected by moving the pointer to them individually and pressing Space, before the forwards/backwards function is usable with them.

Configuration for one's own system set-up is accomplished with the QJump Config routine. Making changes to run the program from hard disk was straightforward, although it involved several files. The Directory window still came up with flp1_ as the default device afterwards, despite win1_ being inserted everywhere possible, and there was no evident way to alter this. Up to eight directories can be specified, as shown in the second illustration.

Conclusion

There were several detail faults in the review copy which needed fixing, but overall operation of FiFi was good, the speed (with Gold Card) being particularly appreciated. A file finder will be a desirable accessory for many users; this one is speedy, and has sensible options. For users already having the Pointer Environment, this is one more useful tool which can easily be added to others. It does take rather a lot of memory for loading extensions - over 50 KB - and this might preclude its use on systems with little ram to spare. About 30 KB of this total would be allocated for the same extensions in a system already running any other PE program, however.

Open Channel

Open Channel is where you have the opportunity to voice your opinions in Sinclair QL World. Whether you want to ask for help with a technical problem, provide somebody with an answer, or just sound off about something which bothers you, write to: Open Channel, The Blue Barn, Tew Lane, Wotton, Woodstock, OX7 1HA.

A blank

With reference to our telephone conversation I am returning my copy of QL World December 1992 to you for replacement.

My complaint is that the article on pages 10 and 11 which lists all the good thing about PCs, and the article on pages 38 and 39 which lists all the reasons why I should change my QL for a PC, have no place in my magazine.

I enjoy your magazine very much and wish all your staff a happy new year.

P R Denscombe
Redditch
Worcs

Here's a reader who knows how to get an editor reaching for the tranquillisers. But it seems that Mr Denscombe has misread his Contents ... the articles on pages 10/11 and 38/39 of his December QL World are about phoning up the printers and asking them why (oh why?) are there four blank pages in this magazine? When everyone else's copy has two pages of Open Channel and two on Desktop Publishing? The only answer is to send him one that's black and white all over, and hope that it will quickly be read.*

*(*Tea and biscuits.)*

Clocked

The quirky clock described in November's *Troubleshooter* is not confined to the Sinclair roms, as I find my Minerva-equipped QL seems to predict that time will reverse at around 7 seconds after 14 minutes past 3am on January 19th, in the year 2029. Perhaps we should all make a note to avoid benchmark testing around this time.

If my calculations are correct, this is the point at which the Hex representation of DATE should change from $7FFFFFFF to $80000000, and I cannot help feeling that such regularity if probably highly significant to more competent hackers than myself.

It will probably also be significant that I own a Tandy pocket calculator, which has a Hex capability, and this also seems to do very silly things at exactly the same value.

L Atkins
Biggleswade
Beds

On the other hand, maybe they know something that we don't. The message is: enjoy the early years of the next Millenium, as we may have to live through them all again backwards.

Defeated

I am writing to you because I would like you and some of your members to help me with information on the program *PC Conqueror*, as I cannot save or print via my QL serial output using a Centronics Parallel printer interface when running Wordstar.

I have been your reader now for over five years. I have DOS 3.2 and 3.3 and with Conqueror I can run all my programs of IBM type, but I can't save or print, and also can't print the screen.

I have tried to get in touch with Digital Precision but have had no luck. Also the program Xover will not load, and when it does the exchange between the IBM and the QL will not work as it should.

H Gataaura
Hounslow
Middlesex

Has anyone experienced this problem with this combination? If traders like Digital Precision are not available over holiday periods, incidentally, they generally become reachable soon afterwards, and nearly always have answering machines. See the section on Answering Machines in QL Scene.

Newsagents

After about a three-year absence from the QL community due to events beyond my control, I have just now managed to track down your magazine via my local Newsforce paper shop, who, it must be said, managed to get hold of the magazine very quickly within one week of my initial order, which is excellent. I now receive a copy on a monthly order basis, as opposed to taking out a subscription which, being as I am on state benefit, would make a large hole in my income.

The cover price in all the time since I bought my last QL World has only gone up by about 25p. Other magazines for certain Sinclair products that I get have gone up from £1.25 to £2.50, an increase of 100%, or five times the rate of QL World, and all you get is a duff covertape with programs no-one really wants.

The last issue I bought, from WH Smiths, was May 1989, the issue that was the last to be stocked on general magazine stalls, I believe. I have since tried to find out if the Microdrive Exchange is still in operation, but it isn't, is it? Pity, it would be great if I and other 'new' readers could have the chance to sample programs of years and issues gone by.

Is there a chance that you may introduce this or a similar feature (like a public domain selection list, etc.) again into your otherwise excellent magazine?

I know budgets may be on the tight side, whose itsn't? But £2 per disk would be acceptable to most users I think, and you'd be a supplier to be trusted. (The magazines referred to above do this to a limited degree, but the software is commercial, not public domain.)

This is the first letter I have ever sent to any magazine, so, if it doesn't make it, I'll understand and try again!

Keep up the valiant fight!

Chris Dalby
Bradford
W Yorkshire

And a very sensible letter it is too. I suppose being away for three years has provided a new perspective. There were certainly copies of QL World in 'selected newsagents' (as the trade likes to say) well after May 1989, but it has been hard to find 'off the shelf' copies for many years and moreso since the recession started. However, Arcwind deals directly with a lot of small outlets, so it is very viable to order copies from a helpful local newsagent. Back issues, unfortunately, are only available from May/June 1992.

Microdrive Exchange used to provide a royalty for programmers commensurate with the effort of producing a good program, but that arrangement broke down a long time ago with a falling market, rising costs, problems with distributors, shortage of mdvs — you name it. I kept it going for a while on a value-added basis for readers, but it emerged that sales were erratic and programmers were not receiving royalites (quite apart from those lost when Focus went down), so I called a halt. I discuss it with people from time to time, but I won't even try to take it up again until there is a sizeable clear space in other ongoing administrative challenges. The possibility of doing something combined with Public Domain offers is not a million miles from my mind, however.

Machine code

I wrote to *Open Channel* in August suggesting that at least one article on Machine Code for complete beginners would probably be welcome to many like myself, who do not have the least idea how to start.

I took Ms. Armstrong's suggestion I should have a go without necessarily understanding everything. I purchased a copy of the Metacomo *QL Assembler Development Kit* (second hand, of course) and decided to have a try.

My first attempt was made with Listing One from Mr Mellor's *Font Utilisation* article in the October *QL World*. The assembler came out with a list of aroud 40 errors! Careful checking revealed that four of them were typing errors, but I could discover no more mistakes of my own.

In sheer desperation I started guessing, and - what do you know? - eliminating $$ from after the operands in lines 7 and 10 removed the errors against those lines, while assuming that the many .1 entries that I had placed in the Opcode field should have been lower case I eliminated the rest!

But after all this I have achieved very little! When I try to use the routine with one of the Textfonts in the Quanta library version of *Page Designer*, nothing happens.

I had another attempt, this time using the Real Time Clock Display in Mr Dicken's book *QL Advanced User Guide*, only to come to grief once again. I have searched and searched, but can find no difference between what I have listed and what is printed.

Please try to persuade one of your contributors to provide a starter article which goes through a few codings, line by line, explaining what each opcode and each operand does, and an alphabetical list of the standard mnemonics, with an explanation of their function (or pointers thereto). Words of no more than two syllables, please.

L Atkins
Biddleswade
Beds

Starting this month!

History lesson

PLEASE REPEAT AFTER ME: THE FIRST CONSIDERATION IN PRINTING A SERIOUS MAGAZINE IS THE LEGIBILITY OF PRINT. The information on (the upaginated) page 31 is virtually unreadable without some hard work on the part of your faithful subscribers. You may recall that I questioned a (much less) difficult piece of printing in the issue carrying my Armada Index. Having read it once, I still can't recall what the SJPD/QL World competition is all about! Please keep your designers on a tight rein.

John Roberts
Littlehampton
E Sussex

Yes John Sir, right away. Don't you mean a tight lead? I must admit, it isn't one of the art department's triumphs. I think I might run it again. By the way, that is the most hideous typeface I've ever seen in my life, although (unlike many hard-up users), thanks for using a ribbon which isn't virtually transparent.

Washable

I am saving a stamp by enclosing my return for the readers' survey (with another letter). You will notice that it is somewhat battle-worn. When the mag arrived, its envelope contained as much water as magazine. One unconsidered benefit of the paper you use nowadays is that it is much easier to dry out. Although I am sometimes reminded of a lady I once knew who had been a Petty Officer (boats) in the WRNS during the war, and in her later years sailed a very lovely ketch around the West Coast. Her first encounter with a terylene genoa prompted the comment that the *!£@* thing would blow overboard. Modern magazines are similarly slippery. Very difficult to balance on one's knee and twiddle a forkful of spaghetti at the same time.

PH Tanner
Glasgow

But easy to wash up afterwards. Glossary: genoa: not a small rodent; a type of sail; ketch: chunky sailing boat with a large hold used for fish or weekenders; terylene: synthetic substance not used for making paper. The washable qualities of modern magazines are appreciated by those who own elderly cats. The code in the middle of this letter has been (you'll be glad to hear) accurately reproduced.

Urgent!

Whenever my files get corrupted; say, twice a year, the most disastrous and likely event is the destruction of the memory map at the start of the disk, because that involves the loss of not one, or two, files, but ten, fifty, a hundred or 250 files. Of my disks, several have gone that way over the past few years, and been consigned to a temporary graveyard, while I hope for something to resurrect them. Now, alas, something more serious has happened.

My excellent Extra High Density disk-drives supplied by Miracle Systems have proved ideal (apart from Ferret not working on them) until I found my Disk Adapter (also from Miracle) which permits me to link up the older Cumana disk drives. The Adapter worked, but its fit into the Gold Card was not particularly firm, and being just next to the F5 key, it was easily knocked, with unhappy results.

In my case, apart from the time that it simply turned off the two disk drives, on one occasion it cleared some of the memory map. Now I have a 3.2M disk, which is full of files, including, inevitably, some recent ones not backed up, but no Memory Map to them.

So, the question is: what recovery program will cope with a missing Memory Map, reconstruct the files listed on it, and then write a new map? Surely it can't be that difficult for the machine-code wizards who create programs for reading tracks and the headers that constitute the maps. How about it? PS It would be good to have Ferret working again to find urgently-sought strings, too.

John Roberts
Littlehampton
E Sussex

Oh dear. You seem to be having serious problems with readability everywhere this month. Try: Simon Goodwin's Revive (SuperBasic in Action, this month and last month); various public domain routines from Qubbesoft or CGH Services; Media Manager SE from Digital Precision (that's the high-tech expensive option); in any case, talk to Ron, Richard or Freddie (or all three) before you make a purchase, to see if their programs will deal with your situation. If you can find a copy of Disk Doctor, which used to be popular, try that.

The horrible truth is that you are storing up trouble for yourself by using a system of mixed type and age. Stuart at Miracle says: call him about the Adapter connector - they haven't had this problem before, and its a standard push-in connector widely used in industry (we have never had a complaint about a Disk Adapter falling out or getting knocked).

Editor's notebook

A long article this month: an introduction by Ian Bruntlett to the much-discussed but often mysterious QJump *Pointer Environment*. This is incorporated into several modern QL programs - the most well-known of which is *QPac2*, a bundle of QL accessories by Tony Tebby. It provides something like a graphic front-end, and makes multi-tasking easier by controlling program windows. It uses the keyboard or a mouse, and is not expensive. Some people have been put off by its apparent complexity, but it is well worth investigating. Ian will be writing more on Qpac2, and Dilwyn Jones can offer more information.

I'm running the Competition again - but don't forget the clue is in the December issue!

Don't forget - if you have an idea or program you would like to write for *QL World*, write and tell the Editor, who will supply comments and further information on formatting, etc.

One of the QL suppliers (who rarely uses an answering machine) has asked me to say a few words about telephone answering machines (non-use of, enquiriers and customers by). But I phoned him for advice and he was out ... maybe next month!

EEC MONITORS

A stop press last month announced that Bill Richardson had not been able to obtain top-up stocks of the popular Goldstar QL monitors.

The good news, however, is that he has a good stock of the Philips CM073 monitors, which are excellent for the QL The monitors are good-quality, tested second-user machines, at a competitive price with a 3-month warranty.

The Philips CM073 14-in EGA colour monitor has an excellent specification, is very controllable, has an 85-character capability, and externally accessible controls for width, height and centering. It has a green/amber text-mode switch, and a dedicated QL monitor lead is provided with the purchase.

The monitors cost £145, inclusive of VAT, plus £9 for packing and delivery charges.

Contact W N Williamson (EEC), 18-21 Misbourne House, Chiltern Hill, Chalfont St. Peter SL9 9UE. Tel. 0753 888866.

International QL Meeting

The Dutch QL User group Sin_QL_Air, and the German QL User Club SQLUC are proud to announce the 1993 International QL Meeting on Saturday 27 February. The meeting, the third of its kind, will be held as before at the St. Joris College, Roosterlaan 296, Eindhoven, in Southern Holland, from 10am to 5pm. Set-up is available for groups and suppliers on the previous evening (Friday), but people are asked to keep in touch with the organisers for an appointment. Entrance will cost 3.50 Dutch Florins (including a free drink), and a two-meter table for exhibitors costs 15 Dutch Florins.

The meeting is dedicated to the QL and Qdos (including Minerva and SMS2). People will be able to give and get information (although the only person we know who actually becomes information is Freddie Vachha when the spirit is on him), show off private and commercial projects, meet other QL users and have a good time. Now that EC border tariffs have been reduced, a little shopping might be in order, as well, for those who like Dutch food and drink. No information is available at the moment about which dealers will be there. (You can always phone anyone you wish to see, and ask them.)

Further information can be obtained from the organisers: In the Netherlands, JJ van der Molengraaf, Mullerweg 17, 5624 JC Eindhoven, Holland, tel. (from UK) 010 31 40 442309 or Marco Holmer, J P Coenstraat 51 bis, 3531 EN Utrecht, Holland, Tel. 010 31 30 948673; in Germany, Franz Herrmann, Talstrasse 21 W-5460 Ockenfels, Germany. Tel. 010 49 2644 1855, weekends only. Bulletin Board: QLAT-BBS, 010 31 30 962265 (v21,v23,v22(bis),v23) 2:283/508, 24 hours. A map and information about hotels and boarding houses is also available (although advice may be needed by those who don't know Dutch).

Canadian Club

The Toronto Timex Sinclair Users Club has been in touch with a copy of their bi-monthly bulletin *Sinc-Link*, Big Fall 1992 Issue. Named as the largest computer-club newsletter in North America by a survey in a US magazine, Sinc-Link covers all Sinclair computers (ZX81, 2068, etc.) but has a fair amount of QL coverage. The newsletter has 36 pages in US format, stapled at one corner.

Material clearly comes from many different authors and sources - some is a bit filled-in, but it is all clearly readable except a rather "dotty" section on the ZX81 which might give some people problems - I certainly found it hard going. Sinc-Link prides itself on not overdoing its listings section, but generally providing information. Several letters are reproduced, including the information from Carl Delhez about his Sinclair emulators (which *QL World* readers can find on page 32 of the December issue). Former Quanta editor Alan Pywell writes "SuperBasic Ramblings".

Membership of the club is $20 a year. The QL contact is Hugh Howie, 586 Oneida Drive, Burlington, Ontario L7T 3V3, Canada.

Star QL Drivers Moondust!

News has gone out from popular printer purveyors Star Micronics UK that 'Sinclair users' now have easy access to printer drivers for all Stars models.

What, we asked their publicity department, was a Sinclair User? They said they'd get back to us on that one.

Eventually we spoke to Star's Technical Hotline. Alas, it seems that there has been a mistake. "We haven't had anything like this," the Technical Hotline told us. Star's Hotline office - better informed than many computer dealers, their printer-driver adviser knew at once about the QL and was as surprised as we were at the news - will tell us if any such thing does indeed arrive. But he expressed his doubts. David Taylor in marketing checked up and told us: "It's a mistake. Thank you for getting in touch with us."

So it looks as though the QL and its Sinclair cousins are simply the victim of the eager media-world's tendency to mistake the august name of 'Sinclair' for just another brand of PC. Shame. But we were impressed by the Hotline and Marketing Department's quick response when we were on the line.

Leeds Quanta

Eleanor Kirkpatrick is co-ordinating a new Quanta group in Leeds/Brandford area. The next meeting should be Wednesday 31 March from 7 to 9 pm at John Parkin's "QL Classroom", St Edmund Capion School, Rhodesway, Bradford. Send Eleanor an SAE for details and a map at: 37 Gledhow Wood Avenue, Leeds LS38 1NY.

MC Erratum

As well as improved versions of Listings 5 and 6 from Systematic Machine Code part 10, we also have corrected versions of the KEYROW and SOUND routines from the same episode. Write to the Editor if you want copies. Please note that the same routines in the whole-program listings are from a different file and work as printed.

Dilwyn Dazzles

New from Dilwyn Jones is *Screen Dazzler*, a screen-saver program which produces some form of animation to save screen phosphor burn-in if no key is pressed for a given period of time. This requires at least 384K of memory on the QL. Users can write their own animation routines if they have a compiler, using the Screen Dazzler manual as a guide. Some sample animations are on the disk. The program is on 3.5in or 5.25 in disks, but not mdv, and costs £15.

For the puzzled: a Screen Saver program places a moving picture or pattern on the screen when the computer is running but not in use, to prevent accidental burn-out of monitor phosphors if it is left running for long periods (see *Open Channel*, Fishhhhh, last month), and is not to be confused with a Save Screen or Screen Save routine, which saves the contents of a screen to a file for printing or transfer.

Dilwyn writes: "If people who want fishtanks are prepared to produce the animation program themselves, Screen Dazzler could activate them as screen savers. What is needed is a Basic compiler, and a sprite package such as Digital Precision's Super Sprite Generator. Write the program, get it to work, and finally add it ot the list of programs which can be activated by a program such as Screen Dazzler."

Contact Dilwyn Jones Computing at 41 Bro Emrys, Tal-y-bont, Bangor, Gwynedd, LL57 3YT, UK. Tel. 0248 354023. Please avoid calling (very early or late, as the phone still rings even when the answering machine is on!

QL Packaging for the year 2000

Jurgen Falkenburg Computer Technik are offering a professional packaging job on the QL, known as the QL 2000.

Falkenburg also manufacture some expansion add-ons (see below), but the QL 2000 is essentially a re-housing of existing QL systems in a modern tower case, with room for expansions and connections for a Falkenburg's own, popular QL-Keyboard-90 outboard keyboard.

Users can send their existing QL rigs to Falkenburg to be built up, or buy a do-it-yourself package. The custom-built option costs from DM 699 (about £280) and the DIY option costs DM 399 (£160) inclusive of EC VAT. Fitting of the DIY option requires "only some screwing-and soldering-work" following the installation manual which is included.

The QL 2000 package has been developed with users of the Gold Card, Keyboard 90, and/or the Falkenburg QL-HDD-card hard disk interface, but other peripherals such as the QBoard, Trump Card and Miracle disk drives can be used; and, unlike many upgrades, the old QL microdrives can be assembled, two together, in one of the two 5.25-in disk slots provided.

The QL 2000 is compact, with a footprint of 5in wide by 17in deep, and a height of 13 inches. The front panel has room for two 3.5-in floppy or hard disk drives, two 5.25-in floppy or hard drives, one 3.5in hard disk internal socket, plus room for a further 5-in or 5.25-in internal hard disk socket.

The internal space is enough for the regular QL circuit board, plus up to five QL peripheral cards. Falkenburg's own expansion cards are designed to be mounted vertically or horizontally depending on what type of case they are being installed into; in the QL 2000 case, they mount vertically.

On the back of the case are 110/120-volt or 220-240-volt AC, monitor, serial port, joystick, network and keyboard connectors. The keyboard connector is, as stated, provided for the QL-Keyboard-90, but fitting a different keyboard normally only requires changing the socket type.

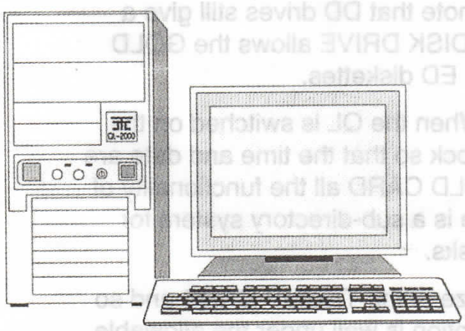
Also available is a "powerful 200-watt power supply with a noiseless blower".

The already-established QL-HDD-Card is a hard-disk expansion board suitable for expanded and unexpanded QLs, designed with considerable versatility for use with "any MFM-resp. RLL hard disk with up to 416 MB capacity". It supports the filing system 2 used by the Gold Card, or Level-2 updated Trump cards and QBoards with subdirectories. Two hard disks can be plugged into the interface.

Users interested in the QL-HDD-Card are advised to get in touch with Falkenburg with information about their systems, as prices etc. vary with the type of disk drive to be driven, case configuration, etc.

In the UK, WN Richardson & Co. (EEC) are acting as Falkenburg's agents. This is particularly useful for users who don't want to send hardware back and forth to the continent. Both companies are a source of information on the QL 2000 system and other Falkenburg upgrades.

Jurgen Falkenburg
Computer Technik, Thanweg
36, D-7539 Ersingen, Germany.
Tel. (from UK) 010 49 7231
81058 (voice or fax).



QBits and Pieces

QBits of Willingham, Cambridge, a small dealership that has been trading away quietly for the last three years, is selling a range of software under the QBits Budget Software label at £5 plus 50p post and packing per program.

Titles in this range include *Conundrum*, where you work against the clock to build up a word chosen by the computer, using either jumbled letters or clues. This currently includes free of charge a file editor called *WordGen* to enable you to generate your own word-lists for the game; *Early Learn*, two simple educational games (*SpellBear* and *Clocks & Tables*) to help children master the basics of spelling, tables and time-telling (analogue and digital); *Sam*, a two part game (*Storeman Sam* and *Warehouse Sam*) with an educational basis, in which you unload and load lorries in a depot, aiming to make profits from orders; *FTidy_128*, a simple file-handler displaying up to 64 file names, to copy, delete, print, rename and view files easily. This package currently includes a free game, *APM_Sweeper*, and finally, *QL-Engine*, a detailed animation of forty screens showing the internal combustion engine in a four-stroke, four-cylinder combustion cycle, written by an experienced engineer. *QL-Engine* needs a Gold Card to run because of the detailed graphics.

QBits also handles a useful list of second-user QL and disk-drive hardware, software, books and backissues of *QL World* and *Quanta*.

QBits are at 29 Silver Street, Willingham, Cambs. CB4 5LF. Tel. 0954-60288. Phone or write for a current list or information.

All Formats Fair Diary

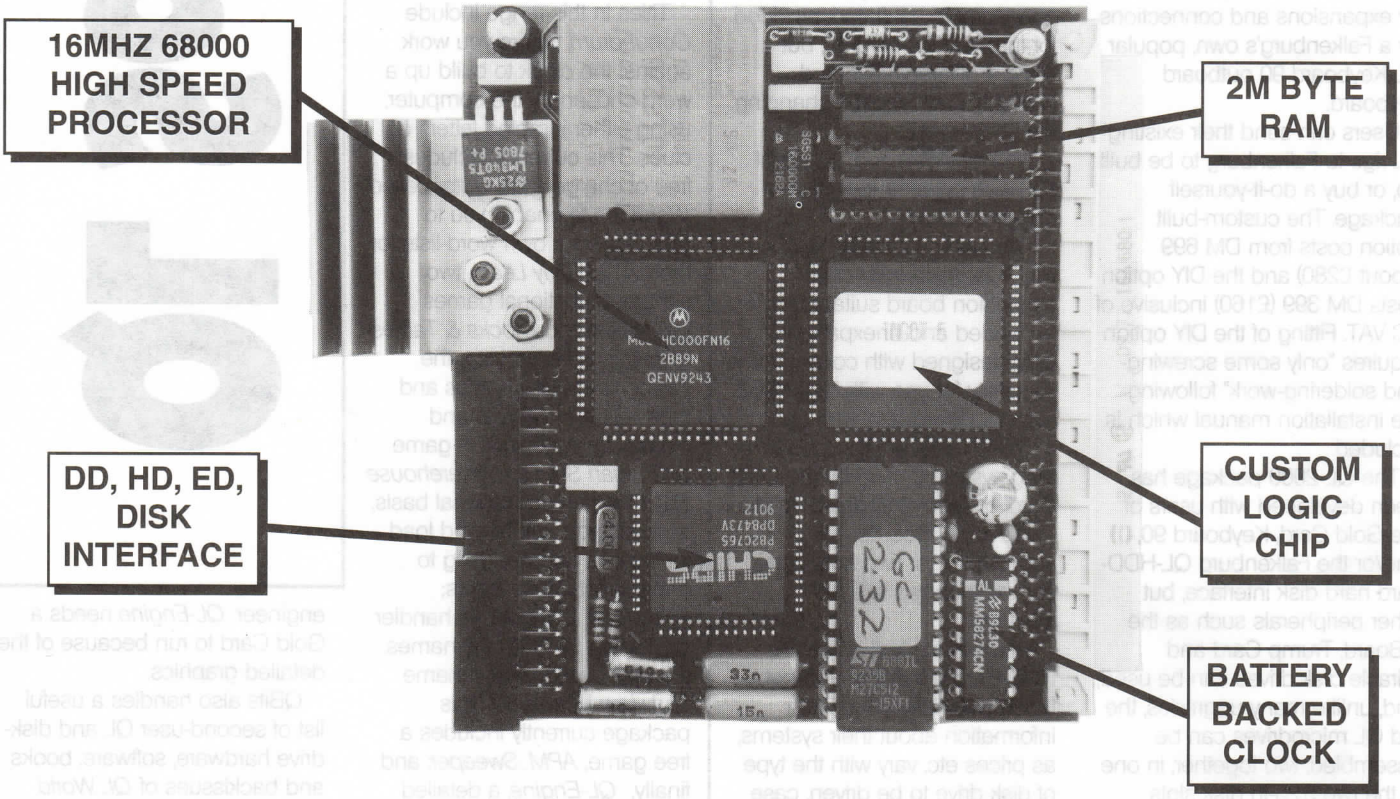
Coming dates for the All Formats Computer Fair are:

Feb 20: Haydock Park Racecourse, M6 junction 23; Feb 21 Scotland City Hall, Candleriggs, Glasgow; 27 Hemel Hempstead Dacorum Pavilion, The Marlowes Centre; 28 Feb western Brunel Centre, Templemeads Station, Bristol. Mar 7 North University Sports Centre, Calverley Rd., Leeds Mar 20 London Sandown Park Mar 21 West Midlands National Motorcyle Mar 27 North West Haydock Park Racecourse, junction 23 M6 Motorway 3 Apr Edinburgh Appleton Tower, George Square 4 Apr Glasgow City Hall, Candleriggs, Glasgow Apr Nottingham University, Jesse Boot Centre Apr 18 West Midland National Motocycle Museum, junction 6 M42 Apr 24 London Sandown Park, Esher, Surrey junctions 9/10 M25 Apr 25 Bristol Brunel Centre, Templemeads Station.

Check with any particular supplier that you are hoping to see whether they will be at a particular Fair. If you have far to travel phone All Formats 0608 663820 to check arrangements haven't changed. Many QL suppliers only attend the Glasgow and London fairs with any regularity. In London the Hammersmith venue is preferred.

Tickets are £4, but attendees can get up to 50 £1-off vouchers if they send an SAE to the organisers at: Maple Leaf, Stretton-on-Fosse, Moreton-in March, Gloucestershire GL56 9QX. (Only one voucher can be used per ticket, of course.)

MIRACLE



QL GOLD CARD

£225 inc. (£200 outside EC)

This is the expansion that has been revolutionising the QL. It is very easy to fit - it simply plugs into the expansion port at the left hand of the QL and once fitted it will instantly increase the execution speed of the QL by about 4 times due to the presence of a 16MHz 68000 on board. There is 2M of fast 16 bit RAM of which QDOS sees a contiguous 1920K. The remainder is used for shadowing the QL's ROM and display memory and for the GOLD CARD's own code.

There is a disk interface which can access 3 mechanisms (4 with the DISK ADAPTER) of 3 different densities, DD (double density, 720K), HD (high density, 1.44M) and ED (extra high density, 3.2M) in any mix. The disk interface connector is the same type that was fitted to the TRUMP CARD so most QL compatible disk drives can be used. Please note that DD drives still give a capacity of 720K per diskette. Our DUAL ED DISK DRIVE allows the GOLD CARD to access DD, HD and ED diskettes.

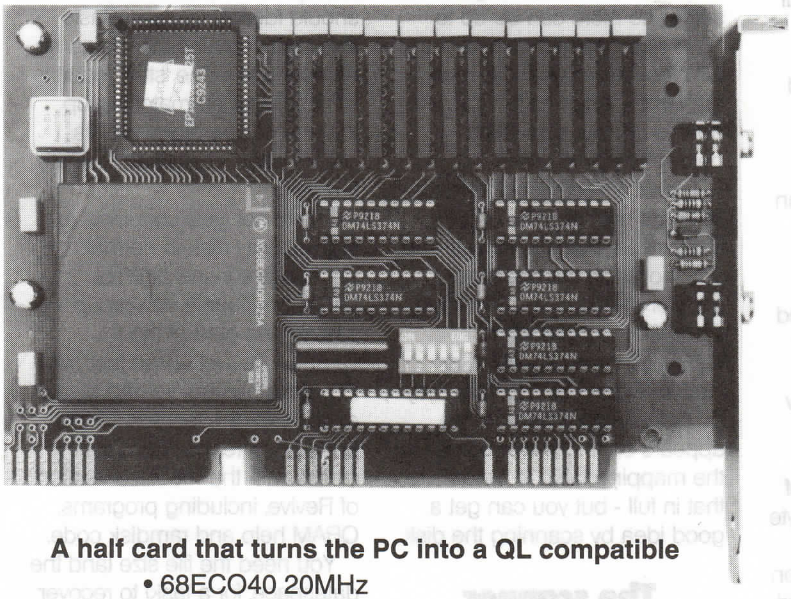
Another feature is the battery backed clock. When the QL is switched on the contents of the clock are copied into the QL's clock so that the time and date are correct. The firmware in the ROM gives the GOLD CARD all the functionality of the TRUMP CARD like TOOLKIT II and there is a sub-directory system for floppy and RAM disks.

Physically the GOLD CARD is about half the size of the TRUMP CARD and so fits almost all within the QL. Its current consumption is well under the allowable maximum so no special power supply is required. The GOLD CARD comes with a 14 day money back guarantee and a 2 year warranty.

SYSTEMS

Coming soon...

THE QXL



A half card that turns the PC into a QL compatible

- 68ECO40 20MHz
- 8M bytes of RAM
- QL compatible network port
- QDOS compatible operating system
- Uses PC as I/O system

Product philosophy

To maximise the performance of new and existing QL programs on readily available low cost hardware.

Intended user

Everyone who has access to a PC or anyone who requires the ultimate QL replacement.

Minimum PC configuration

XT or compatible, 512K, EGA, 1 spare slot.

See us, DP, TF Services and many others at St Joris College, Roostenlaan, Eindhoven on Saturday 27 February 1993.

VISA MASTERCARD

Tel: (0904) 423986

To place an order by phone please have your credit card ready. For overseas customers we charge the prices shown in brackets.

To order by post, please fill in the form opposite or write to us quoting your credit card number and expiry date, or enclosing a cheque payable to MIRACLE SYSTEMS Ltd.

To: MIRACLE SYSTEMS Ltd, 25 Broughton Way, Osbaldwick, York, YO1 3BG, U.K. Tel: (0904) 423986

Please send me ______________________________

I enclose a cheque to the value of: £ ______________

Or debit credit card □□□□□□□□□□□□□□□□

Expires □□ / □□

Name ______________ Signature ______________

Address ______________________________

SuperBasic in Action

In part two of Revive, Simon Goodwin addresses error trapping and high-density disk formats.

Revive is a SuperBasic program to recover files from damaged disks. Modern floppies are very reliable, but you should still keep backup copies of all your files. It is always easier to recover a program from a backup than from a damaged disk.

That said, most of us have ended up at some time with valuable data on a disk with an unreadable directory. That's when Revive comes to the rescue.

The program lines presented this month fit into gaps in last month's listing. They perform *Turbo* error trapping and allow Revive to scan disks and recover files in blocks. I also discuss the special features of the High Density 1.44 megabyte and Extra High Density 3.2 megabyte disk formats used on the Miracle Systems Gold Card.

By default Revive expects double density disks in drive 1, but you can change to another drive or density by typing the QFLP direct sector access name in full, for example: FLP2_*D2D, FLP1_*D2H or FLP3_*D4E (lucky you!) when the program asks for a good disk. Alternatively, edit line 420 to change the default.

The program is written for double density disks of 180 to 756K capacity, but it can easily be adapted to repair Gold Card high density disks. These have "QL5B" at the start of the first sector, rather than "QL5A", so alter line 650 to check for "QL5" in SECTOR$(1 TO 3).

Gold formats

1.44 megabyte HD disks use 512 byte sectors, like double density, but pack 18 onto each track and side. Files are allocated in 1.5K blocks, as before, but the disk map is twice the original size (2880 bytes, two blocks) and there are 960 blocks allocated on each disk.

3.2 megabyte ED disks use 2K sectors, so the value 512 in the program needs to be replaced with 2048, at line 420, 740, 800 and 1670. The directory and map take up eight sectors, and each block is a single sector, so there can be up to 1593 files on each disk, and a full block reports can be more than three times as long as the double-density equivalent - perhaps 200K or more.

The QL directory holds fixed-sized 64 byte records of the name, type, size, dataspace and dates of each the file. If the directory is intact you can read these 'file headers' with OPEN_DIR from Toolkit 2, or GetHEAD from DIY Toolkit. They do not tell you where the file appears on the disk - you need the mapping table to determine that in full - but you can get a good idea by scanning the disk.

The scanner

A vestigial 64 byte header is created at the start of the first block when a new file is opened; the size and dates are unset but the name does appear. Revive's SCAN_DISK option spots likely headers, indicating the start of a file and its original name. Beware: RENAME changes the directory entry, but not the name at the start of the file!

The Gold Card formats do not use or set this header - my investigations with version 2.31 of the Gold Card find it unset on HD disks. On ED disks it contains junk from the disk buffer - often old and arbitrary directory information. If an ED disk is full of 2K files, up to 100K might be occupied that way!

If you're working with the new Gold Card formats you can leave out lines 1240 to 1330, and 1500 to 1560, as these handle the double density header. You should add one line, to make it easier to spot the start of files:

```
1345 PRINT #4;"Later: ";
TIDY$(block$(65 to 114)
```

If this is the beginning of a new file, the 'start' report line will be junk. The new line shows the first bytes of the file, which should identify text and Basic files, and contains the job name of most tasks. The 'start' report is still useful when matching up the end of one block with its continuation in another - usually the next.

I have not (yet!) corrupted a high density disk in normal use, but I have a friend at West Midlands Quanta sub-group who will be glad of the re-vamped Revive! I have just two ED disks, but they've had plenty of use over the last few weeks, and I have 'recovered' a variety of files with the modified version of Revive, including programs, QRAM help and ramdisk code.

You need the file size (and the dataspace, for a task) to recover a code file successfully. Sometimes the end of the file is obvious when you examine the recovered blocks in an editor; ideally you'd have a WSTAT report taken before the disk was damaged.

FORMAT writes strings of zero bytes to each virgin sector, and Revive can guess the end of most files by spotting these. In some cases the file may be 'fragmented', which means allocated in several chunks. This may happen if the file is extended after an older file is deleted.

Big blocks

If a block is readable, Revive returns the entire contents, even if the last block was not completely used by the file. You need the exact file length to recover a code file, and the dataspace for a task, but Psion data, text files and program source still load with extra bytes from the end of the block.

If you find junk at the end of the file, you may need to chop the end off using a text editor, or the PUT and TRUNCATE commands of Toolkit 2 and most disk systems.

Often you can determine the end of a task by scanning the file with a binary editor like *Spy* or *The Editor.* QL task lengths are always even - if not, EXEC will crash! Dataspace must be even and positive; you might start at 8K and adjust this if you get an 'out of memory' report.

When a disk's directory is intact you can read the size, type and dataspace of a file quickly with GetHEAD, from DIY Toolkit volume H. You can change the type or dataspace with SetHEAD. If you've time and memory to burn, use LBYTES and SEXEC.

The program was written and compiled with Turbo, and uses a mixture of *Turbo Toolkit* and *Toolkit 2* extensions. Now I'll explain how you can get it working with other Toolkits or secret features of the Sinclair rom.

The IMPLICIT% and DATA_AREA directives are only needed for Turbo compilation, and can otherwise be omitted. The QLib equivalent of IMPLICIT% is DEF_INTEGER. Such directives show a compiler that a name is only used for integer values - whole numbers between -32768 and 32767 - allowing faster, more concise code to be generated.

Error trapping

RETRY_HERE statements and the WHEN_ERROR .. END_WHEN block at the start of the second listing implement Turbo error trapping. You should not enter these unless you intend to compile the program with Turbo.

Any error in the compiled task causes the code at WHEN_ERROR to be executed. RETRY sends execution back to the most recent RETRY_HERE point, discarding any local variables or return details generated since then.

RETRY_HERE statements appear before all the lines that may need to be repeated if an error is detected. A 'blanket' error trap covers other cases, so the task should never crash, regardless of the circumstances, unless error details fill all its memory, or something goes wrong in the WHEN_ERROR block!

The WHEN_ERROR block calls ERLIN% to check the number of the line where the error occured; if it is not one of the expected lines it uses ERNUM% to find the cause and reports suitably. If you have a JS or later rom you could use REPORT to print the corresponding Qdos message, and ERNUM and ERLIN to pinpoint the cause.

Minerva users could employ their interpreter's WHEN ERROR facility to trap errors. This may seem to work on JS or MG roms but in practice Sinclair bugs mean that some errors are ignored or lock the machine, even if Toolkit 2 is loaded.

If an unexpected error occurs, the program closes channels #3 and #4 and re-starts after complaining in window #0. If this happens hundreds of times you are likely to run out of dataspace eventually. RETRY is used whenever repeated errors are likely; it is preferable because it tidies unused local variables and RETurns, so the task loses no memory.

Line 640 and 1670 use GET to position the file pointer and INPUT$ to read the sector. If you do not plan to compile the program with Supercharge or Turbo, you can replace the INPUT$ with an extra parameter for GET:

640 GET #3\1,sector$

Beware: attempts to read a 512 byte sector from an ED disk lead to a total crash, requiring reset to recover. The opposite case is more forgiving: Gold Card 2.31 reports a drive error and returns control to the user, if you unplug the cable after accidentally trying to read a big sector from a DD or HD disk.

The Turbo Toolkit function DEVICE_STATUS(1,file$) is used at line 2080 to check the report file name. This returns an error code if the file does not exist, rather like FOPEN_IN, or the amount of free space on the drive otherwise. By default the report appears in a large SCR window, but you can send it to any device.

Line 2160 uses GET #3\1E7 to write to the end of an existing file, extending a report. The large parameter winds the file pointer to byte 10,000,001 of the file, or the end, which is usually sooner. Use Turbo Toolkit's SET_POSITION instead of GET if you want the program to work without Toolkit 2.

CURSOR_ON #0! is the Turbo Toolkit equivalent of Toolkit 2's CURSEN #0, or the useful but unexpected PAN #0,0,115 on Sinclair roms. Likewise CURSOR_OFF #0 corresponds to CURDIS #0, or PAN #0,0,116.

The STRING% function is a SuperBasic version of the eponymous Turbo Toolkit routine. If you have Turbo Toolkit you do not need to enter this function. The SELect statement avoids temporary results outside the range -32768 and 32767; this suits Supercharge's fast integer code.

More soon

I welcome suggestions for future articles and information in this series. The aim is to show how useful, adaptable programs can be written in SuperBasic. Write to Simon Goodwin, SuperBasic in Action, QL World, The Blue Barn, Wooton, Woodstock, Oxon OX7 1HA. I look forward to hearing from you.

```
100 REMark REVIVE - SuperBASIC in Action disk data recovery program - ©SNG
160 WHEN_ERROR
170   reason=ERLIN%
180   SELect ON reason
190     =600
200       COMPLAIN "This interface does not allow direct access to data."
210       STOP
220     =640 : COMPLAIN "Format description is unreadable."
230     =1090,2190 : COMPLAIN "Please enter first & last block numbers."
240     =1150,2100,2160 : COMPLAIN "Unable to open " & file$
250     =1670 : read_ok=0 : REMark Set flag for trapper
260       PRINT #4\"BLOCK " & block_no & " IS UNREADABLE."\\
270     =REMAINDER
280       reason=ERNUM%
290       SELect ON reason
300         =-11
310           COMPLAIN "Drive full."
320         =-1,-5,-6,-9,-10,-16,-20
330           COMPLAIN "Output failure (code " & reason & ")."
340         =REMAINDER
350           COMPLAIN "Unexpected error " & ERNUM% &" at "& ERLIN% & "."
360      END SELect
370      CLOSE #3 : CLOSE #4 : GO TO 480 : REMark POP?
380   END SELect
390   RETRY
400 END_WHEN
410 :
1030 DEFine PROCedure SCAN_DISK
1040 HEADING : INK green
1050 PRINT \"        Ready to scan disk - disk blocks are numbered from ";
1060 PRINT "0 to ";total_blocks-1;"."
1070 INK white
1080 RETRY_HERE
1090 INPUT \"Enter first & last block numbers to be scanned "\first\last
1100 IF last>=total_blocks : last=total_blocks-1
1110 IF first<0 : first=0
1120 RETRY_HERE
1130 INPUT \"Enter output device name for report ";file$
1140 IF LEN(file$)<3 : file$="SCR_448x200a32x16"
1150 OPEN_NEW #4,file$
1160 on_screen=file$(1 TO 3)=="scr" : IF on_screen
1170   CLS #4 : CURSOR_ON #0!
1180   PRINT #0;"  Press CTRL-F5 to pause, or ESC to quit."
1190 END IF
1200 file$="" : busy=0
1210 FOR index=first TO last
1220   GET_BLOCK index
1230   PRINT #4;"Block ";index
1240   IF CODE(block$(1))=0
1250     REMark It may be a file header: length is < 2^24
1260     name_length = STRING%(block$(15 TO 16))
1270     IF name_length>=0 AND name_length<=36
1280       REMark Name seems plausible
1290       IF CODE(block$(4))=64
1300         SHOW_HEADER
1310       END IF
1320     END IF
1330   END IF
1340   PRINT #4;"                Start: ";TIDY$(block$(1 TO 50))
1350   PRINT #4;"                End  ";
1360   IF busy AND block$(block_bytes-9 TO block_bytes)="0000000000"
1370     PRINT #4;"of file '";file$;"' inferred at block ";index
1380     busy=0
1390   ELSE
1400     PRINT #4;": ";TIDY$(block$(block_bytes-49 TO block_bytes))
1410   END IF
1420   IF on_screen
1430     IF INKEY$(#0,1)=CHR$(27) : EXIT index
1440   END IF
1450 END FOR index
1460 IF on_screen : CURSOR_OFF #0
1470 CLOSE #4 : CLOSE #3
1480 CLS #0 : COMPLAIN ""
1490 END DEFine
1500 :
1510 DEFine PROCedure SHOW_HEADER
1520 file$=TIDY$(block$(17 TO 16+name_length))
1530 PRINT #4;"                **** Start of file '";
1540 PRINT #4;file$;"' ****"
1550 busy=1
1560 END DEFine
1570 :
1910 DEFine FuNction TIDY$(text$)
1920 LOCal index,c
1930 FOR index=1 TO LEN(text$)
1940   c=CODE(text$(index))
1950   IF c<32 : text$(index)="?"
1960   IF c>top_limit : text$(index)="!"
1970 END FOR index
1980 RETurn text$
1990 END DEFine
2000 :
2010 DEFine PROCedure RECOVER_FILE
2020 HEADING : INK white
2030 REPeat recovery_loop
2040   RETRY_HERE
2050   PRINT \"Enter name of file for recovered data (or enter nothing";
2060   INPUT " to restart) "\file$
2070   IF file$="" : EXIT recovery_loop
2080   status=DEVICE_STATUS(1,file$)
2090   IF status=-7
2100     OPEN_NEW #4,file$
2110   ELSE
2120     IF status<1536
2130       COMPLAIN "Unable to open '"& file$ &"', error code: "&status
2140       NEXT recovery_loop
2150     END IF
2160     OPEN #4,file$ : GET #4\1E7
2170   END IF
2180   RETRY_HERE
2190   INPUT \"Enter first & last block numbers"\first,last
2200   INK green
2210   PRINT \"Writing to ";file$
2220   FOR index=first TO last
2230     GET_BLOCK index
2240     IF index=first
2250       IF status=-7
2260         PRINT #4;block$(65 TO); : NEXT index : EXIT index
2270       END IF
2280     END IF
2290     PRINT #4,block$;
2300   END FOR index
2310   CLOSE #4
2320   PRINT \file$;
2330   IF status=-7 : PRINT " created." : ELSE PRINT " extended."
2340   INK white
2360 END REPeat recovery_loop : CLOSE #3
2380 END DEFine
```

Easy with Easel Part 2

Henry Orlowski creates a custom graph - Load up Easel and follow him.

The first thing we need to get a feel for in *Easel* is the method of putting items into the graph. We touched on the subject of direct data entry last month in part one. With Easel newly loaded up, you will have the crosswires over 'Jan'. To enter the value 25, simply type 25, watch it appear in the input line at bottom left hand side of display, and then press Enter.

You will now notice that three things have happened. First, a large bar has appeared in the Jan cell. Secondly, the left hand scale has expanded from 0 to 10, to 0 to 25 automatically. This is of course to accommodate the new value of 25. Thirdly, the crosswires have moved one cell to the right and now occupy the Feb cell.

Now type in some text. This is different from entering values, but only because you have to remember to precede it by the double or single quotation marks, " or '. These tell Easel to expect text to be following. Type in your quotation mark and see how the input line changes to 'Text'. Now type in something like 'This is a piece of text' - don't type the quote marks again, these are only included to make you realise which bit it is you're supposed to be typing in. You don't need closing quote marks.

Changing text

Notice how the text you have just typed in appears not only in the input line at the cursor prompt but also at the crosswires position in the display area. Of course the chances are that the text will not be in the place that you want it. No problem - once you've pressed Enter to say you're happy with the text, you can then move the text with the cursor keys in any direction to the position you want. Try it. When you're ready, press Enter again and the text will be positioned in that spot.

Suppose the text you've just put in was a mistake or you changed your mind about where you wanted it. Just press F3 to get into command mode (like *Quill*). Select E for Edit from the command menu in the control area. Press T to select the Text option. Now move the crosswires until they are somewhere at or near the text you've just keyed in. Now press Enter. As if by magic, the text is now displayed on the input line for you to edit as required. This can be done with the standard Ctrl-left or -right and retyped as appropriate. However what we are going to do - is get rid of it altogether. Do this by pressing F4, and the text will disappear. Try to remember the use of F4 for deleting, as it will come up again and not just for the text option.

If necessary now press Esc until you are out of command mode and back into input mode. You will recall that we started off by putting in a value of 25 for Jan into our graph. What we need to do now is to get red of that value because we're going to create a new set of figures, and change all the other items you see on the display to make our customised graph or chart.

Get the crosswires back to Jan, by keying in Shift-tab until they are over Jan. To delete the value of 25 we can either press F4, or simply overwrite the value directly with a new value. Let's just press F4 on this occasion. We now have a blank graph again so we can start from scratch. Position the crosswires back on Jan. I hope you can remember how this is done!

Feeding plans!

Say you were an interested parent doing a small PTA project involving something like schoolchildren's meal preferences at school. You were researching both boys and girls, a certain number of each, and noting their choice of a particular meal type. Your table of results might look something like this:

	Boys'	Girls'
Cooked	5	6
Takeaway	3	4
Sandwich	4	3
Snacks	3	2

Some of you will have grasped that this type of table is typical of an *Abacus* spreadsheet in appearance and application, and this is indeed so. They both after all display the same values. It's merely the method of presentation that's different.

For now we will confine ourselves to Easel. Let's start creating our graph. Starting with the boys, key in 5, then 3, then 4, then 3, ie the values on the table above, not forgetting Enter after each one. You now have four bars on screen of varying heights representing those values. Notice how each new value caused the crosswires to move one cell to the right.

Title and text

We now need a title. Let's make it 'Meal Choices'. We'll leave out 'boys' as we will use this to distinguish from the second set of figures we will key in for the girls subsequently. As before, key in F3 then E for Edit then T for Text. Move the crosswires close to 'Title' and press Enter. 'Title' will be displayed on the input line at the cursor prompt. Remove this text (you should know how by now; if not, read back a few paragraphs). See how the text both disappears from the input line and the display at the same time. In its place enter 'Meal Choices'. You don't need the quote marks, but I won't go on telling you this. Press Enter, then move the text to a central position near the top of the graph. Again you should know how to do this now.

At the end of a command, Easel is inclined to stay in the command mode until you press Esc to return to input mode. Do it now. I won't remind you again after this.

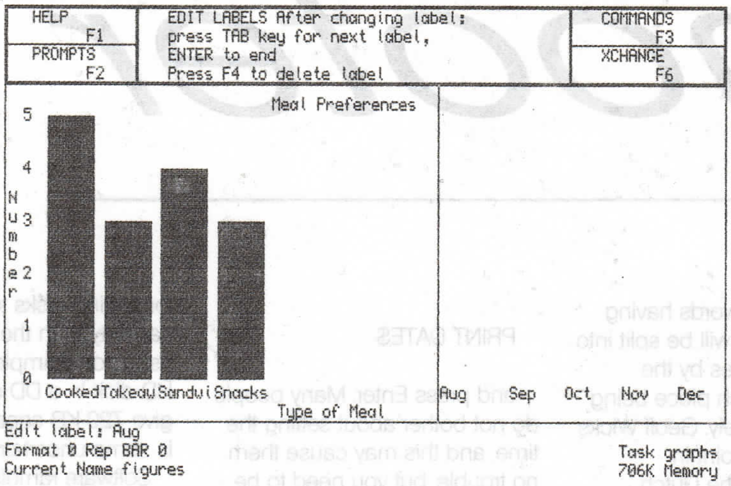
Now we will put in another item of text, 'Boys' to distinguish this graph from the one we will do for the girls. Get into text entry mode. Remember that you do this by pressing the single or double quotation marks. Key in 'BOYS' then Enter, and position it somewhere near the top left hand corner of the graph using the cursor keys. Key Enter when satisfied with the position.

Changing labels

Now we need to do something about those months of the year. These are known as 'labels' and the labels we need in this case are types of meal rather than months of the year. Get into command mode and E for Edit then L for Labels. The crosswires will attach themselves to a cell and the label of that cell will be displayed on the input line. Get them to Jan if not already there, and use the line editor to change Jan to 'Cooked', but don't press Enter.

You will now have 'Cooked' as your first cell label. You will still be in the edit label mode, so Tab to Feb. Change that to 'Takeaway' then change Mar to 'Sandwich' then finally Apr to 'Snacks'. Press Esc until you have exitted the command mode. Only part of some of your labels may be visible in the display but don't worry about that at this stage.

We now need to do something about Axis 1, the horizontal axis. Return to Edit mode, this time selecting A for Axis then H for horizontal. Change Axis 1 to 'Type of Meal'.



```
HELP        EDIT LABELS After changing label:      COMMANDS
      F1    press TAB key for next label,                F3
PROMPTS     ENTER to end                           XCHANGE
      F2    Press F4 to delete label                     F6
```

```
Edit label: Aug
Format 0 Rep BAR 0
Current Name figures
                                                  Task graphs
                                                  706K Memory
```

Move the text to a central position and press Enter.

Let me just point out at this stage again that I will no longer give you the exact keys to press every time because I consider you should have grasped it by now. So you just have to remember to do things like press Enter to finish, how to enter different modes, how to get out of different modes, which keypresses give specific results, etc.

The vertical axis also needs some editing. Use the E for Edit mode then V for vertical axis and change Axis 2 to 'Number' and position centrally.

Kill a cell

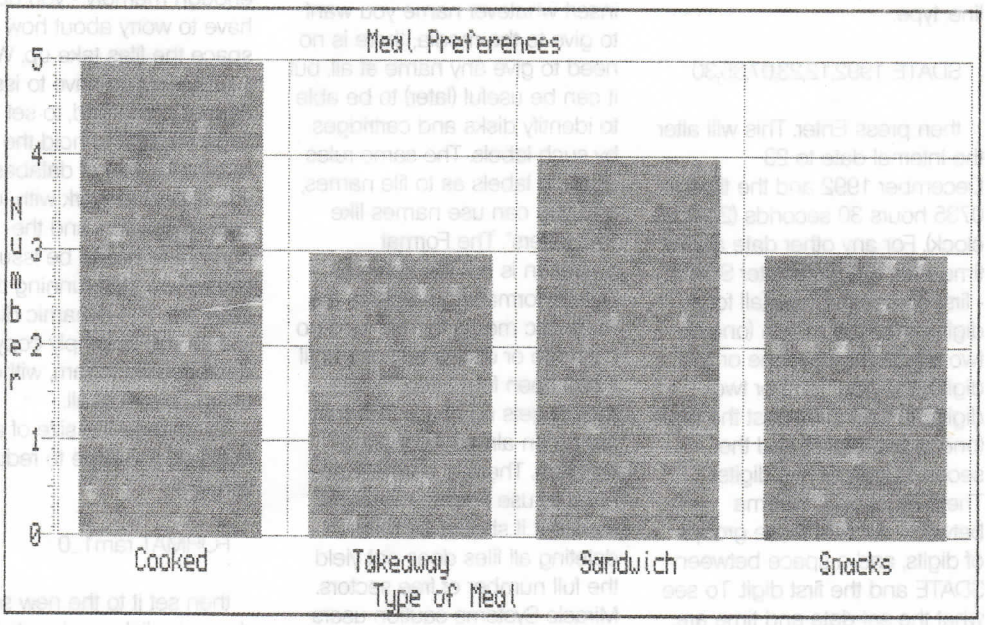
The graph is now starting to look the part. The one remaining aspect that still needs sorting is the fact that we still have the May to Dec labels intact. If you delete the labels, the cell may still be there albeit without a label or name. To delete a cell altogether, we must delete its value and its label. In this case these cells have not had any values entered into them so we are part of the way there. Remember how we deleted values before in case you ever need to do this again. Use the Edit mode with L for Label, and F4 to delete all the remaining labels.

Notice again how Easel helpfully keeps you in that mode and moves on to the next cell as you continue, speeding up the whole operation immensely. When you get to Dec, Easel opens up another range of cells for you, and presents you with an even greater crowding problem. This is Easel just trying to be helpful. It doesn't know that you don't want any more cells.

Don't worry. Instead just enter command mode again and this time select V for View. At the input line cursor prompt accept 'All figures' by Enter, and accept the 'format' in the same way. Don't worry about these at this stage. All will be revealed later on, it's just that it's easier now to allow Easel to get on with it.

As if by magic our graph is now redrawn on screen but without the unwanted cells and labels but with the wanted cells expanded to take up the available display width. Not only that, but there is now enough room for the whole text of the cell labels to be visible.

Just compare what you see now on screen to the table of values we started with. Which do you prefer? If you prefer the Easel version, and it's purely a subjective matter, then you may just have experienced what Easel is all about and what it can do to enliven a bland set of figures by giving them extra impact.



Naming sets

We're still not there, because we have still to think of the girls, whose results we will compare with the boys. Firstly however to avoid confusion between different sets of figures we need to name this set something instantly recognisable. In the status area you will see that the current name for the set of figures is 'figures'. This is the default that Easel gives when you first enter a set of figures. However it would make more sense to us to have it called something like 'boys'. To do this we use the 'Rename' command. Of course it is not always necessary to call your set of figures a special name, especially when you are only using one set at a time, but in this case we need another set for the girls.

The rename command is accessed by selecting R for Rename in the command mode. At the prompt type in or accept the old name of the set of figures, in this case 'figures', then Enter, then type in the new name, 'boys', then Enter again. Notice that 'boys' is now reported as the current set of figures in the status area.

Next month we will look at how we can efficiently create the girls' graph and compare the two in tandem. We will also look at various options to customise our graph exactly to our requirements.

Before you switch off, save the work we've done so that you can load it up again next month. To do this you select S for Save from the command menu. At the prompt, key in the filename, say 'meals' not forgetting the drive specifier, eg flp2_meals, unless your data defaults are automatically set up. This will save all figures currently in memory. Then press Q for quit to exit Easel.

Trouble shooter

Bryan Davies goes back to basics with some readers' letters

This month's article is concerned solely with readers' letters.

Roland Kaiser wrote about using *Perfection* for German correspondence. He had apparently misunderstood information given to him about using the Perfection spellchecker. While Perfection itself can handle the "special" German characters (eg those with umlauts), the spellchecker is primarily for use in checking English words, as would be expected from the fact that the word lists supplied are in English only. Word list files for other languages can be created by users, but they should not contain characters which are not in the standard English-language set. The spellchecker regards non-standard characters as being gaps between words, so that German words having these characters will be split into two or more pieces by the spellchecker, each piece being checked separately. Geoff Wicks mentioned this point in connection with the Dutch language, in his letter published in *Open Channel* in the December 1992 issue of *QL World*.

Following recent meetings of a local Quanta group, Mick Halpin asked that a few basic points be repeated, for new users like him. As he says, however well-known some operations on the QL are to experienced users, there are still users who do not understand how to do some fairly simple things, and cannot readily track down articles which dealt with them in the past. They may not know where to look in the *QL User Guide* for help, either, although they should certainly read that book through carefully, more than once, as it is generally helpful. Here are some of the items he asked for advice on.

Date and time

On the SuperBasic command line, type:

```
SDATE 1992,12,23,07,35,30
```

then press Enter. This will alter the internal date to 23 December 1992 and the time to 0735 hours 30 seconds (24-hour clock). For any other date and/or time, alter the digits after SDATE - first comes the year (all four digits), then the month (one or two digits), the day (one or two digits), the hour (one or two digits), the minutes past the hour (one or two digits), and the seconds (one or two digits). There must be a comma between each of these groups of digits, and a space between SDATE and the first digit. To see what the set date and time are, type:

```
PRINT DATE$
```

and press Enter. Many people do not bother about setting the time, and this may cause them no trouble, but you need to be aware that there are circumstances in which it can cause problems. For instance, if you run an accounting program and need to have back-up copies in chronological order, you will not be able to do this without making sure the clock is correctly set. When you come to check which is the latest version out of two copies of the same file, how can you tell (quickly), if the date stamp on both is incorrect?

Formatting disks

This job is basically the same for disks and microcartridges. Type:

```
FORMAT devn_label
```

then press Enter. For "devn", use the device and its number, that the medium to be formatted is in. That is, if you want to format a disk in the first floppy drive, insert "flp1". For "label", insert whatever name you want to give to the media; there is no need to give any name at all, but it can be useful (later) to be able to identify disks and cartridges by such labels. The same rules apply to labels as to file names, and you can use names like "wp_letters". The Format operation is not the same as Delete; formatting prepares the magnetic media for use, and no cartridge or disk is any use until it has been formatted. Delete merely gets rid of any files that are on an already-formatted medium. There is normally no need to use Format a second time, but it should be used if deleting all files does not yield the full number of free sectors. Miracle Systems caution users about possible problems when formatting disks to higher capacity than the drive being used; for example, formatting a HD disk in a DD drive should give 720 KB capacity, but there is some uncertainty about this.

Software ramdisks are essentially the same as hardware floppies or cartridges in use, but there are a couple of different types, called fixed and dynamic. The fixed ramdisk does not change size unless you issue a revised Format command for it. The size of the disk is specified in sectors - that is, the figure used is twice the figure for the kilobytes space required. Type:

```
FORMAT ram1_600
```

to get a 300 KB ramdisk. The dynamic version changes size according to what is going on in the QL, in the way of programs being run. Initially, all the available memory is combined and called Ram1_. If another program is then run, memory will be taken from Ram1_ to allow running of the program. A big advantage of this flexibility is that you can run Archive programs with the database files in ramdisk and - provided there is enough memory - you don't have to worry about how much space the files take up. With the fixed form, you have to issue a Format command, to set a disk large enough to hold the maximum size of database file you expect to work with in the current session, and the command has to be issued before you start running the program. The dynamic disk allows you to simply copy the database file to ram, without using Format at all.

To change the size of a fixed ramdisk, you have to reduce it to zero first:

```
FORMAT ram1_0
```

then set it to the new size. A dynamic disk can be changed in size by simply issuing the Format command again, with the new size. Both types need to be empty of files before the size is changed. As the dynamic disk takes all available free memory by default, the effect of reducing its size is to create another one to take up the freed space. For example, with 300 KB free, you get Ram1_600. If you then format Ram2_300, that automatically reduces the first disk to Ram1_300. Naturally, any files stored in existing disks place limitations on the space left to be allocated to new disks. As with other devices, there is a limit of eight, so Ram8_ is the last one you can have.

Switching on

All equipment manufacturers (so far as am aware) advise that high-power devices be switched on first. This is to try and avoid surges of electrical current passing from high-power devices into the computer itself and damaging it. In practice, this means switching on printer, display, drives, then last - QL. One chip in the QL - the 8301 - tends to be very touchy, because it is not protected by any form of buffering from surges of current emanating from other devices. In particular, the display can "kill" the 8301 if switched on after the QL. The order of switching off should be the reverse one - that is, switch the QL off first. As I have said before, my own practice is not to do what the manufacturers advise, simply because it is very inconvenient. Going round all the switches one-by-one is a nuisance. Instead of this procedure, all my system devices - including the QL - are connected to one multiple socket and are switched on and off together. As this procedure has not resulted in any failures over eight years, it would appear to be a safe one. To some extent, the safety or otherwise of the procedure must depend upon how well designed the equipment is. Clearly, the old QLs in particular leave something to be desired in this respect, but the Microvitec Cub display on my system would appear not to cause trouble for the 8301 chip. Be aware, though, that other displays may not be as obliging.

Disks in drives

This subject is related to switching on the system, in that the QL does not provide good protection for disks during the switching period. Files and/or directory information can get corrupted if disks are in drives at switch-on or switch-off time, so switch on first, then push the disks in. Likewise, pop disks out before switching off. This advice does not apply for resetting the QL unless something is wrong with the QL, resetting should not harm any disks which are inserted in drives. The same should apply to cartridges, although I have always tended to remove them before resetting; maybe they are less-well protected than disks. To ensure that the system boots automatically from a disk or cartridge, the latter has to be pushed in promptly after the system has been switched on, before the F1/F2 initial screen appears. The order of priority for booting is mdv, flp, win; that is, the system will not boot automatically from a hard drive if either a cartridge or a disk are already inserted in mdv1/flp1.

Write-protection

Cartridges and disks are similar to music cassettes in having small tabs which prevent existing data being overwritten or deleted. The cartridge has a tab on its right side (as viewed when pushed-into the drive) when it is supplied new. To protect the information on the cartridge, the tab can be cut off (using a pair of electrical side-cutters, for example). It is desirable to remove the tab from all your important master program cartridges. When the time comes to modify files on a cartridge, or add new ones, the protection can be removed by placing a piece of tape over the cutout that the tab covered. Unlike disks, the cartridge tab is for mechanical, not optical sensing; the tab actuates a microswitch via a lever, when the cartridge is pushed into the drive. It is sufficient to replace it by a piece of thin transparent sticky tape. Don't use a bigger piece than is necessary to bridge the gap, and avoid folding it underneath the cartridge, to prevent misalignment of the cartridge when in the drive.

Disks of both sizes have tabs which are detected by optical sensor; the presence of the tab prevents the disks being written to. 5.25-inch disks have a cutout on the left edge, and this will be uncovered when the disk is supplied new. Place a piece of tape over the cutout to protect files on master program disks. The tape must be opaque - transparent sticky tape is not suitable. If you don't have any of the black or silver tabs supplied with new disks, use black electrical insulating tape. Apply it neatly and firmly - it can get stuck inside the drive and cause you a lot of trouble, and maybe necessitate a repair. (Be aware that electrical tape can go sticky after a few months or years not good in the drives.) 3.5-inch disks have a much neater arrangement - a slider at the front left corner. On new disks, the slider will be pushed away from the front edge of the disk, and it will be covering the square hole that the optical sensor looks for. In this condition, disks can be written to. To protect them, push the slider to the outside edge, uncovering the hole. DD, HD and ED disks are all the same in this respect. The other square holes that HD and ED disks have are there for density-recognition purposes - a square hole on the right edge, located the same distance from the front edge as the write-protect hole, indicates an HD disk, whereas a similar hole located slightly further down the right edge indicates an ED disk.

Saving files

This is a very basic operation, which all users should be familiar with, but there are variations of it, depending upon what program is being run. *Quill* and the other Psion programs have a Save command which is essentially the same for all, but there is one slight difference in what the user has to do. When you select the Save command, the current (screen) file will be saved to cartridge or disk, under the name you give it at the time, or the name it already had when you loaded it. If a version of the file already exists on the medium, you will be asked whether or not you want it to be overwritten by the current version. Depending on which program you are using, the answer has to be pressing the Y (yes) or the Enter key, to cause the existing file to be overwritten. The Enter key is used as a way of saying "yes" in some programs - such as The Editor - but that convention is not followed in many other programs and you have to answer "y" with them. The standard QL Basic has a Save command, but it does not ask whether or not an existing file should be overwritten; when a file of the same name exists, the Save command simply does nothing, but you get the message "already exists". You have to Delete the existing file first, then Save the new version - a tiresome procedure, although it does help prevent overwriting a wanted file by accident. Many systems have some form of Toolkit installed and they usually have the modified command SAVE_O; this automatically overwrites an existing file of the same name. This command is very useful when you are modifying boot files.

Backing up files

There can be confusion over the use of the terms "back-up" and "make a copy". They are essentially the same. Quill has a Backup command, SuperBasic has a Copy command; both do the same job, which is to make a copy of a file on one medium onto another medium. You end up with two identical files, each having the same file name but being on different disks or cartridges. It does not matter what the file is - data or a program. You are making a spare copy for security reasons; that is, in case the original file gets deleted or corrupted. In Quill, there is no effective difference between using the Save and the Backup commands. If you Save to different media one after the other, that is the same as using Backup to make a copy from one medium to another. There is some difference between what the commands can be used for. The Backup command can be used to load a new PRINTER_DAT file, to enable printing to another printer or with a different set of

Translate settings without having to exit Quill. The Save command is strictly for _DOC files. Both commands have the confirmation request, which is made if a file of the same name as the current one already exists on the destination medium.

In Basic, Copy is like Save in that it will not overwrite an existing file of the same name, but you get the "already exists" message. Toolkit commands include WCOPY ("wild-card copy"), which is much more useful than the basic Copy. You can copy any or all files from one medium to another, by making use of the Y/N/A/Q query which appears after you have issued the command. The procedure is to type:

```
WCOPY flp1_,flp2_
```

then Enter. You will be asked what you want to do about the first file (in alphabetic order) on the source disk (flp1 in this case). Y to copy it, N to skip over it, A to copy this file and all others following it, Q to quit the command there and then. Choosing A or Q leaves you with no more questions to answer about file names (except as noted below) - all the files from there on are scheduled to be copied, or the command is cancelled right away. Y or N in answer to the first question bring up the next file name and the same question, and so on. You can be completely selective in what you want to copy. Y and A bring up a further question - overwrite or not? - with the same four options, if a file of the same name is eventually found on the destination medium. Hopefully, after this two-stage warning, you should have been alerted to the possibility of overwriting files which you do not actually want changed.

Quill loading

Quill cropped up again in a letter from G T Morris. He uses *TaskMaster* for switching programs and finds that Quill will load only about half of a 17,400-word file (the Gospel of St. John). The remainder of the file appears as "only two or three letters at the beginning of each line". By splitting the file into two halves, he has been able to load one half and merge the other half into it. If I understand what he says correctly, the second half was saved using another WP program. The Quill version he is using is 2.35, which is the best one available, and it should not give problems loading a file of this size. It was a long time since I had worked with TaskMaster, but it seemed likely that the memory allocation set up in that program, for Quill, was too small. Another possibility was that Quill could not overflow the large file onto disk because there was not enough space there. There is a file size limit beyond which Quill 2.35 gives trouble, but I found that it was around 600 KB and the Morris file should be much smaller than this. Apart from the usual problem of slowness, and occasional hiccups, there should be no real difficulty in handling, say, 250 KB files.

To check Quill's behaviour under TaskMaster with large files, I set TaskMaster up with the default allocation of 65000 bytes of memory for Quill, then tried to load a 117 KB, 17,000-word _DOC file. The first try produced a typical Quill "no can do" message, presumably because the data disk did not have enough free space to allow Quill to overflow the loaded file back onto that disk. When enough space was left free on the disk, the file started to load, but the QL locked up, with just the first line of the file displayed on the screen. The DEF_TMP file on the disk was exactly the same size as the main file, indicating that the load attempt was almost complete. The third try was successful. The next file tried was 145 KB and 22,000 words. This loaded remarkably quickly, thanks to the Gold Card, and gave no trouble. A 173 KB, 27,000-word file also loaded quickly and without trouble. At this point, it seemed time to "go for broke", and the next file wheeled in was 288 KB and 42,000 words. This was going to be a close thing, as there was about 576 KB of space on the disk, for the main and temporary files. Sure enough - "out of memory", with a word-count of -2212. Quill runs out of steam when counting words at around 25,000 to the best of my recollection.

Not obvious

The reason for not loading was not the obvious one, however. There was still 9 KB of free space on the disk, and main and temporary files on there were the correct size. Trying again with more free space on the disk gave the same result. Increasing the memory allocated to Quill by TaskMaster to 70,000 bytes did the trick - for a while. A few moves down the document with SHIFT+Down Arrow had the "out of memory" message flashing again. However, you have to accept that Quill operation will always be borderline with large files if only 60-70 KB space is allocated to it; 80-100 KB is much better. One point to bear in mind when disk space becomes scarce, and a document is overflowing, is that you can put the cartridge/disk with the document file on it into mdv1/flp1 and load it from there; if you put a blank, formatted cartridge/disk in mdv2/flp2 for the overflow, both main and overflow files can be roughly the size of the medium.

That is only a partial answer to G T Morris's letter. Yes, Quill is capable of loading files much larger than 17,000 words (assuming the file size is in proportion to the number of words and is not considerably inflated by formatting information). No, I have no idea why it should load a file with the last half of it being shown as just a few characters per line. On second thoughts, there is one thing I noted about Quill years ago, and that is that it sometimes loads the bulk of a file, then baulks at some character(s) - possibly not displayed on-screen - it comes across, with the result that the remainder of the file looks like nonsense. To deal with this, you can make a copy of a few pages around the suspected area of text, then delete that area and see if loading is now successful. If it is, the removed text can be merged back in, block by block, and loading checked, until it is established just where the problem lies; any remaining text can then be re-typed in.

Card query

Miguel Estarellas of Madrid wrote in September and November about a dispute he has with TK Computerware. Items that he ordered earlier in 1992 had not been received, but his credit card account was debited for purchase of them. He will presumably have since realised from comments in this column that TK has not been responding for six months or so to requests for information on several complaints. We can only advise that Estarellas make a claim to his credit card company for refund of any payment against which goods have not been received. He could also write for advice to the local Trading Standards Office in the area where TK is based (see INFORMATION).

INFORMATION
(The publication from which the following addresses were taken is out-of-date and the information may not be correct, but letters addressed as below should be forwarded to the proper place automatically)

B R Toone
Area Trading Standards
Officer
County Offices
Tufton Street
Ashford
Kent TN23 1BU

G E Edwards
Area Trading Standards
Officer
9/10 Crescent Road
Tunbridge Wells
Kent TN1 2LU

Trouble Shooter

SINCLAIR QL WORLD

THE NEW USER GUIDE

SECTION TWENTY THREE

KEYWORD INDEX

This month in the Keyword Index, Mike Lloyd covers PRINT and PROCEDURE. Only two? PRINT is the Big One.

PRINT #chan, exp1, exp2, exp3, ...

	OUTPUT COMMAND
#chan	(Optional) A valid channel number (default is #1)
exp1, exp2	(Optional) String or numeric constants or variables
separators	The commas separating the parameters can be replaced by any of the following:

!(exclamation mark)Intelligent spacing
\(backslash)Force line feed
;(semicolon)No space between print expressions
,(comma)Tab 8 spaces
TO x   Place next print expression at the Xth character position on the line

PRINT is one of those commands that has a superficial simplicity which leads programmers to forget about its greater powers and subtleties. The first Basic command many people learn is PRINT "Hello World" and for many that is as far as it goes. PRINT is the simplest method of putting text onto the screen. To print a column of numbers (left justified) it is necessary only to write a quick loop along the lines of:

```
100 FOR x = 10 TO 99: PRINT x
```

You can print constants such as "HELLO WORLD" and 98.4 as easily as you can variables such as text$ and this_number. PRINT will also force the calculation of numeric expressions such as 25+7 and can concatenate (add together) text using the ampersand symbol (&). PRINT "Hello " & "World" produces exactly the same output as PRINT "Hello "; "World". Note that spaces within inverted commas are significant, but that spaces in other places in a PRINT statement do not form part of the output. Due to SuperBasic's coercion principles, PRINT "2" + "2" is significantly different to PRINT "2" & "2". The former coerces the text into numeric values and adds them together to produce the result 4, whereas the latter treats the expressions as text and so prints "22". PRINT with the null string ("") or with no parameter or separator at all forces a linefeed to be printed. Any number with more than 6 significant digits will be converted to scientific notation before being printed. If you want to format numbers with trailing zeros, comma-separated thousands, leading currency symbols, etc., you need to write your own function or use the PRINT_USING command provided in *SuperToolkit II*.

Expressions can be mixed in the same PRINT statement by separating each expression with a "print separator". Print separators are punctuation marks with special significance for the PRINT command. Commas, for instance, tabulate output into columns each 8 characters wide. The TO separator also tabulates output, but it allows you to declare at which character column the next print item will appear. PRINT TO 30; "Hello World" places the "H" of "Hello" at the thirtieth character position on the current line. PRINT TO will not backtrack to an earlier point in a line and will not print beyond the width of the current device. Fractional parts (such as PRINT TO 20.4) are ignored.

Beware of the natural inclination to place a comma between the TO clause and the next print item: it will force a tabulation to the next 8-character column, which removes the advantage offered by TO.

The exact size of a single character position depends upon the current screen mode and the values used in any CSIZE command. The backslash character forces a linefeed to be printed, so that the command PRINT "Hello " \ "World" places "World" beneath "Hello". The semi-colon places the next print item immediately following the last item, so that PRINT "Hello" ; "World" prints the two words together without any intervening space. The most complicated print separator is the exclamation mark, which produces what Sinclair called the "intelligent space". In the middle of a line, the intelligent space is no different from any other space. PRINT "Hello" ! "World" produces the output "Hello World". At the end of a line, however, where there is insufficient room for the second item a linefeed will be generated. If the exclamation mark would produce a space at the beginning of a line it is suppressed.

Punctuation

Rather against the grain of normal SuperBasic syntax and the standard rules of English, PRINT statements can include an uninterrupted series of punctuation marks, such as PRINT ,,,,, "Hello".
One of the nice things about PRINT statements is that they can end with a print separator which determines what will happen when the next PRINT command is reached. Where there is no separator at the end of a command a linefeed is entered. Adding a separator adjusts the position of the next print item according to the rules just explained. The number-printing loop mentioned earlier could be rewritten to place the values into a number of columns, as follows:

```
100 FOR x = 10 TO 99: PRINT x ;
```

Note that this feature is device-dependant: a PRINT #2 command has no effect on the next PRINT #5 command, for instance. There is a quirk to the way the exclamation mark works when it is used at the beginning or end of a PRINT statement. It will only do its task properly if there are exclamation marks at the end of the previous PRINT statement and at the beginning of the current statement. The terminating exclamation mark can be replaced by a semicolon if required. A single exclamation mark in these circumstances is not enough.

The PRINT command is not limited to placing text onto the screen, but is also capable of printing to files, the network, and to the printer. This is one of the consequences of Qdos's principle of treating all devices as similarly as possible. However, for newcomers to the QL and to SuperBasic, it leads to an unexpected problem: how on earth do you make the printer work? All previous Basic dialects used the LPRINT command which is conspicuously absent in SuperBasic. The answer, of course, is to open a channel to the printer port (eg OPEN #5, ser1) and then to direct PRINT statements to that channel. PRINT is also invaluable when writing information to files, although there are now alternatives offered, such as Super Toolkit's BGET and BPUT, which can be more appropriate to particular circumstances.

The Qdos code that handles all PRINT requests is also used by the INPUT command. Prompt and answer formatting which is typical of INPUT activity can be undertaken using the same separators as are available to the PRINT statement. Incidentally, all of the low-lying Qdos printing code is substantially rewritten by the Minerva rom and by the Lightning and Speedscreen utilities. None of the changes affect the syntax of the language, but there are some side effects other than improved screen handling of which programmers need to be aware.

Despite the apparent flexibility of the PRINT command it has some serious drawbacks and limitations. The first is that because Qdos does nothing to optimise screen handling the PRINT command is extremely slow, even compared with the lowly Spectrum. The cure is to replace the Sinclair rom with Minerva, or to invest in Digital Precision's Lightning, or to purchase a Gold Card from Miracle. The PRINT syntax has an unusual limitation in that exact screen locations cannot be defined within the PRINT statement itself; this is instead the preserve of the AT command. This rather eccentric arrangement is probably a compromise imposed by the principle of treating all devices identically.

Qdos screen handling is always pixel-based, which means that there is no function to read what character has been placed at a particular character position within a window. The advantages that offset this difficulty are that text can be positioned pixel-perfect using the CURSOR command and that various character sizes can be used within a single window. Nevertheless, the loss of the Spectrum's useful SCREEN$ function has been particularly hard on SuperBasic arcade game writers.

The output from PRINT commands is always left-justified, which immediately imposes a problem for programmers anxious to right-justify numbers or to place all decimal points beneath each other in a column of numbers. The user-defined procedures R_JUST and D_JUST listed below go some way towards redressing these weaknesses, but Super Toolkit users can turn to the PRINT_USING procedure described elsewhere.

```
100 DEFine PROCedure R_JUST (text, charpos)
110 PRINT TO charpos - LEN(text); text
120 END DEFine R_JUST
200 DEFine PROCedure D_JUST (text, charpos)
210 PRINT TO charpos - (“.” INSTR text); text
220 END DEFine D_JUST
```

PRINT_USING #chan, mask, item1, item2, item3, ...

[Super Toolkit II]

OUTPUT COMMAND

#chan	(Optional) A valid output channel
mask	A sequence of strings, print separators and field definitions (see below)
item1, item2,..	A list of variables, one for each of the field definitions (see below)

The PRINT_USING command can appear quite daunting at first sight, but it overcomes a number of the weaknesses of the native SuperBasic PRINT command. Its role is to format numeric output according to the standard rules of displaying numeric values, although it can also mix text and numbers together if required. Perhaps its most useful attribute is that it can inhibit the QL's enthusiasm for scientific notation. The easy parts of the PRINT_USING syntax are the conventional channel number and the trailing list of output items. The difficult concept to master is that of the output mask, which is best thought of as providing a template into which the output items are placed. The mask can contain ordinary text strings and should have at least one field definition (although this is technically optional there is no point in using the command unless there is a field definition for it to work on).

At its very simplest, PRINT_USING might be asked to format a number so that it is right justified and has commas separating the thousands:

```
PRINT_USING “###,###,###.##”, 1234.5
```

The output will be 6 spaces followed by “1,234.50”. The leading spaces ensure that correctly-formatted columns of numbers can easily be printed. The hashes in the format mask each represent a numeral and together form a “field”. It is essential to realise that the total width of the field represents the total number of character positions that will be occupied by the associated output item. If the output item formats to more characters than are available in the field an error is produced.

Fields are separated in the mask by at least one space (or the backslash newline character). In order to print two numbers in adjacent columns, the following command might be used:

```
PRINT_USING “#,###.## #,###.##”, 3456, 245.89
```

The items following the format mask would, of course, normally be variables.If the values represented currencies the appropriate currency symbols could be added to the front of each value:

```
PRINT_USING “$##,###.## = $$#,###.##”, 450.25, 450.25 * dollar_rate
```

The dollar symbol is a special character which indicates that what next follows is the correct currency symbol. For dollars themselves this means that a second $ must be included. European currencies can be represented by “$DM”, “$Fr”, etc. The difference between a mask of “###,###.##” and “$###,###.##” lies in the location of the pound sign on the screen or page. With the former format the pound sign is always the first character in the column with any spaces placed between it and the first digit of the value. With the second format the pound sign is moved so that it is immediately to the left of the first digit of the value, pushing any leading spaces to the left of the pound sign. Generally, the latter format is to be preferred for neatness.

PRINT_USING format masks contain two more features of particular interest when printing currency values. The first is the asterisk special character which replaces all leading spaces in fields so that the output from PRINT_USING “**,***.**” , 980.40 would be “***980.40”. This format is of most use in financial applications that involve the completion of cheques. The second feature provides three methods of showing negative numbers. If your accountant prefers negative amounts in brackets then the PRINT_USING mask should show them, as in "(###.##)". The brackets are automatically replaced by spaces for positive numbers. Otherwise, negative numbers can either be preceded by or followed by a minus sign that, again, has to be shown in the format mask:

"-#,###.##" or "$##,###.##-".

If the minus sign is replaced by a plus sign in the mask then the correct sign will always be displayed, otherwise the minus sign will be replaced by a space for positive numbers.

The QL is always over-anxious to use scientific notation, a weakness that PRINT_USING can overcome. On the other hand, PRINT_USING includes special characters for formatting any number using scientific notation. The rules are quite complicated: the mask must begin with a single hash followed by a decimal point and it must end with four exclamation marks representing the mantissa. The number of decimal places in the exponent is up to you. To reserve a character space for the sign a minus sign can be used for optional signing and a plus sign for compulsory signing, so a typical scientific notation mask looks like "-#.####!!!!", for which the output might be "4.2340E+02".In any format mask it might be necessary to include one of the special characters without intending its special meaning. This is achieved either by prefacing the character with the @ symbol or by placing it inside single or double quotation marks (of the opposite variety to those used at either end of the mask). For example, part numbers on a product database might be prefaced by the hash symbol:

```
PRINT_USING "@# ######", part_num.
```

Another special character is the backslash, which can be used within a mask with the same effect as when it is used as a print separator. If the backslash is needed to represent itself it should be preceded by @.

A final advantage offered by PRINT_USING is that the format mask is simply a string like any other and can be replaced by a variable, such as PRINT_USING #5, mask$, 78.25. This allows you to create masks at runtime, or perhaps allow your users to declare an output format or choose one from a menu.

PROCEDURE

[Turbo Toolkit]

COMPILER DIRECTIVE

The PROCEDURE keyword and its companion, FUNCTION, are two of the cleverest workarounds in the Turbo package. They appear only in compiled code and then only in EXTERNAL and GLOBAL declarations. Technically, PROCEDURE is, according to the *Turbo* manual, a numeric function which returns the amount of free memory remaining (the alternative function FREE_MEMORY is recommended for this task, however). Their job is to identify what would otherwise appear to the Turbo compiler to be a variable as the name of a user-defined procedure or function.

Every EXTERNAL statement has a GLOBAL opposite in some other compiled task. The two declarations allow a program in one task to call routines and refer to variables and arrays that belong to the other task. This arrangement permits code and data to be shared between modules. It would be possible, for instance, to create a library of procedures and compile them just once with a GLOBAL declaration to identify that they can be shared by other programs. Client tasks can then include an EXTERNAL declaration that effectively adds the contents of the global task to their own routines. With arrays and standard variables the statement syntax is easy, but procedure and function names posed a problem: how could the compiler distinguish between the variable VARNAME and the simple function FUNNAME, or between an array ARRAY(0, 0) and the procedure PROCNAME(x, y)? The answer was to precede the function and procedure names with the keywords FUNCTION and PROCEDURE, but the SuperBasic interpreter checks some syntax on entry and it rejects more than one item between commas in a list. The workaround is that FUNCTION and PROCEDURE are dummy simple functions that take no arguments. The SuperBasic interpreter thinks nothing unusual of them, but the Turbo compiler recognises them as keys to the true identity of the item that follows them. Ideally, a directive such as EXTERNAL varname, array(12, 4), PROCEDURE procname would have been preferred. Instead, SuperBasic dictates that the format should be:

```
EXTERNAL varname, array(12, 4), PROCEDURE, procname
```

What's a comma between friends?

Beginners' Machine Code

Beginners start here. This month, Alan Bridewell explains machine code - from the bottom.

Everyone who has had a computer for any length of time will have heard about machine code. If they don't ever learn anything else about it, the one thing they will find out is that to program computers like the experts do, they will have to learn about machine code. (This is not actually true - but that's another story.) What is true is that the ability to introduce a bit of machine code into your SuperBasic programs can improve them out of all recognition, both in speed and in flexibility.

In this series, my aim will be to explain in simple terms what a beginner needs to know to get going. So I will be looking at what machine code actually is, at what assembler language is and what assemblers do, and at many of the instructions needed for programming. In each article there will be simple listings to illustrate what has been covered. In order to make even very short, simple listings do something obvious and visible, I shall make all of them do something on the screen. At some stage it will be necessary to explain exactly how the QL interprets data for the screen, but at the start this will not be needed.

What is it?

So what is machine code? A computer is an electrical device, which treats low voltages as number zero, and high voltages as number one. By putting these ones and zeros together, the computer can make any number you like (within reason!). In the computer, these numbers are stored in memory locations. Each memory location has a unique number, called its address, which has to be stated whenever that memory location is to be used.

Memory locations in a computer are of two types. There is rom (read only memory). With this type of memory, the number stored can be read, and used, but it cannot be changed. All the computer code which is present when you first switch on your computer is stored in its rom.

The other type of memory is ram ('random access memory', which is an obsolete name that ought to be changed, but I don't suppose it will be now. The best thing is to forget what the initials actually stand for, and just remember what ram actually is.) With this type of memory, the number stored can be read and used like rom, but, unlike the rom, it can also be altered. When people talk about how much memory a computer has, they normally mean how much ram it has available.

Ram and rom are each stored in separate memory chips within the body of the computer.

Different things

The numbers stored by the computer can be used to mean different things. They can be simply what they are, numbers, to be used in whatever way our program requires. They can also be used as the addresses of other numbers stored in ram and rom. They can be instructions which the computer uses to manipulate the numbers in its memory. For hardware items like monitors, printers, modems, etc, the numbers can represent characters to print or transmit. (If you have other things attatched to your computer, like robot arms, or sound systems, the numbers can represent other types of information you may need to give them, or receive from them.

The heart of your computer is the microprocessor chip (also called the central processing unit or cpu). This is a device which manipulates all these numbers, treating some as instructions to manipulate other numbers, some as addresses to store and retrieve numbers, and some directly (as actual numbers in an arithmetical calculation, for instance).

How do the numbers 'travel' from the keyboard, or the rom or ram chips, to the microprocessor and back again? As tiny pulses of electricity, originating (usually) either with keying on the keyboard, or as stored one/zero sequences on a magnetic medium such as a disk or microcassette. Put simply, a key operated at the keyboard is translated within the computer into a predetermined sequence of electrical pulses, which travel round the innards, and (if they are the correct combination) will trip further clumps of pulses (binary numbers) which initiate other sequences, and so on like an incredibly complex domino-pattern. Which is why computers need exact instructions before they can perform a given calculation - your computer cannot guess what you mean.

Hexadecimal

For reasons that we need not mention here, the numbering system used on assembler language is not written in the conventional way. We are used to numbers in 'base ten', or the 'decimal' system for everyday arithmetic. In this system we have ten digits to represent the numbers 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9. For bigger numbers, we use more than one digit in a group to represent tens, hundreds, etc. So, for example, we write '555' to mean five lots

LISTING 1

```
; ************************************************************
;         PUTTING NUMBERS DIRECTLY INTO SCREEN ADDRESSES
; ************************************************************
;
        MOVE.B    #$FF,$21000        ; BYTE TO THE SCREEN
        MOVE.W    #$FFFF,$22000      ; WORD TO THE SCREEN
        MOVE.L    #$FFFFFFFF,$23000  ; LONG WORD TO THE SCREEN
        RTS                          ; RETURN TO SUPERBASIC
;
; ************************************************************
```

of one hundred plus five lots of ten plus five, or five hundred and fifty five. Each digit is a number ten times bigger than the same digit on its right.

Our assembler language uses numbers in 'base sixteen', or the 'hexadecimal' system. It obeys the same rules as the decimal system, except that it uses 16 digits to represent the numbers 0 to fifteen. As we only have ten normal digits, to represent 0 to nine, the numbers ten to fifteen are represented by the letters A, B, C, D, E and F. For bigger numbers, we use a second digit to represent sixteens, a third to represent sixteens times sixteen, etc. So in this system, 555 would mean five times sixteen times sixteen, plus five times sixteen, plus five, which comes to one thousand three hundred and sixty five (I think!). If a hexadecimal number contains one or more of the letters as digits, then it's obviously hexadecimal. But if it doesn't, then we need a way to tell us. The normal convention is to put a dollar sign in front. So 555 is decimal, but $555 is hexadecimal.

If this is confusing - don't worry about it. Most of the time we don't need to know what a particular hexadecimal number is in decimal. We simply need to be aware of the fact that the numbers we are using are not decimal numbers.

Word lengths

The microprocessor can handle numbers in three different sizes. First, there are bytes. A byte occupies one address only, and has a size from 0 to 255 in decimal, or $0 to $FF in hexadecimal. Next, comes a word. A word occupies two consecutive addresses, the first one always being an even number. Its size can be from 0 to 65535 in decimal, or $0 to $FFFF in hexadecimal. Last, comes a long word. A long word occupies four consecutive addresses, with again the first one always being an even number. Its size in hexadecimal can be from $0 to $FFFFFFFF, which is 4 294 967 295 in decimal.

This language of numbers is very difficult for human minds to deal with. (Early workers in computers believed this meant that only a tiny number of people would ever be capable of programming effectively - what a shock they would get today to see primary school children doing it at school!) The answer was of course the invention of programming languages that human minds could deal with. However, conversely, the easier it is for human minds to handle the languages, the more difficult it is for the computer. Assembler language is a language that can be converted into machine code directly, and so is easy for the computer to use effectively. It is also possible for the human mind to make sense of it, which makes it the vehicle for human beings to write programs in machine code, without actually learning machine code.

Assemblers

Converting assembler language into machine code requires a special program, called an assembler. If you are interested in starting machine code programming, you really have to get yourself an assembler. Any QL assembler will do for work at a simple level, and there are a variety available with different levels of sophistication and price. Top of the range is probably the *GST Macro Assembler*, which has all the bells and whistles any professional programmer might want. Somewhere in the middle comes Assembler Workbench by Talent. I am not recommending or advertising here. With assemblers, as with everything else, you get what you pay for, and the choice is yours.

The one I use is the *Assembler Workbench*. The only reason to mention this is not to recommend it (although it suits me fine), but the fact that different assemblers use slightly different syntax (arrangement of commands), and if you simply copy the listings in these articles without taking this into account, you may find they will not work on your assembler. The differences are very slight, and easy to deal with, and I will try to point them out as we go along.

The listing

When we write an assembler listing, we write it in four columns, not all of which are needed for all lines in the listing.

The left column is for labels. These are simply names we give to certain places in the listing. They are very useful when we need to refer to one part of a program from another part. For the present, we will not be needing labels, so more about them later.

The next column contains the mnemonics for the actual instructions. These are things like 'BSR' for ' branch to subroutine', and 'RTS' for 'return from subroutine'.

The third column contains the data (called operands) on which the instruction is carried out. Not all instructions require operands. 'RTS', for instance, means return to wherever you got to in the program before you jumped to this subroutine, and it needs no operand. 'BSR', on the other hand, needs an operand in the form of an address of the subroutine. 'MOVE' instructions need two operands, one to tell you what to move, and the other to tell you where to move it, the two operands being separated by a comma.

The right hand column is for comments, like REMark statements in SuperBasic. The assembler ignores them, but leave them out at your peril! Good comments can turn an incomprehensible listing into something you can understand. Even if you understood what you were doing when you wrote a program, the chances are you will not understand what you have written when you return to it later unless you make good use of comments. Comments can also sometimes spread across all the columns.

Doing something

That's enough basic principles for the time being. Let's get on to the first listings. I said earlier that all examples will do something visible on the screen. The QL stores all the information for the screen at a consecutive set of ram addresses from $20000 to $27FFF (Remember, the dollar sign means the numbers are hexadecimal.). When these addresses all contain zeros, the screen is completely blank. What we shall do is use SuperBasic to produce a blank screen, and use a bit of machine code to put other numbers at a few addresses to see the effect.

At this point, we need to deal with an important point of syntax, which is: how does the assembler know when a number is just a number, and when it is an address? The answer is this. If the number is written as just a number, the assembler takes it to mean an address. If it is written with a hash (#) symbol in front, it is taken to mean a number. So, for example, $1C4D means address $1C4D to the assembler, but #$1C4D mean the actual number $1C4D to the assembler.

If you look at listing one, you will see that the first line of code reads:

```
MOVE.B #$FF,$21000
```

MOVE.B means 'move a byte sized number'. The number we are moving is #$FF, (the '#' means its just a number), and we are moving it into ram address $21000 (no '#' means it's an address).

The next line is very similar, except that this time we shall move a word into another address (well, two addresses to be precise) with the instruction

```
MOVE.W #$FFFF, $22000
```

and thirdly, we shall move a long word into a third address (that is, four consecutive addresses) with the command

```
MOVE.L #$FFFFFFFF,$23000
```

As these addresses are all in the part of the ram used to store screen information, the three instructions should all put something on the QL screen. Since our bit of machine code is going to be called from SuperBasic as a subroutine, we need to get back into SuperBasic with a fourth line, which simply says

```
RTS
```

You will notice that, in Listing one, all the comments begin with a semicolon (;). In the Assembler Workbench, this is an essential piece of sytax to tell the assembler that what is to follow is a comment, and must be ignored.

Syntaxes

This sytax is not followed by all assemblers. For instance, if you look at Simon Goodwin's listings in his *DIY Toolkit* articles, the comments need nothing in front, as long as they are restricted to the fourth coloumn. But if they go across other columns, they must be preceeded by an asterisk (*). You must look carefully through the manual for your assembler to see exactly what syntax it uses. If you can't find it, you may be forced to try a little trial and error.

Our next problem is to get the assembler to process the listing. You may find that you can type the listing straight into the assembler. This is very convenient for very short listings like this, but becomes impossibly complicted with longer listings, especially when you find errors, as you almost certainly will. The normal way is to store your listing as a disk or microdrive file, which can then be accessed by the assembler by giving the program the name of the file. How this is done varies greatly from one assembler to another, and you will have to consult the manual to find out.

Having said that, since the name of an assembler code file is the most common information you feed into an assembler, the manual should make obvious how to do it! The other information the assembler will need is what address to start putting the machine code it is generating, and also the name of a disc or microdrive file to save the code. Again, consult the manual.

With SuperBasic

Once we have generated our machine code file, we need to be able to use it in a SuperBasic program as a subroutine. The steps for this are quite simple, and always the same. First, we must allocate some ram to hold the code, using the SuperBasic RESPR command. Next, we LBYTES the file into that ram. Finally, when we reach the position in the program where we wish to use the subroutine, we simply CALL the address obtained by RESPR, and where we loaded the code.

Listing two is a SuperBasic program to do all this. It also opens a window covering the entire screen, and blanks it out before calling the code, so you can see exactly what it does on the screen. The program assumes that the machine code is in a file called LISTING1_code on a floppy disc in disk drive

```
LISTING 2

100 z=RESPR(28)
110 LBYTES flp1_LISTING1_code,z
120 OPEN#3,scr_512x256a0x0
130 CLS#3
140 CALL z
```

flp1_. You may have to alter this to suit your needs.

If you have not yet acquired an assembler, then Listing three is a SuperBasic program which will generate the same file that the assembler would have produced. It does this because I have copied the assembler output into DATA statements, to be saved as a file. This is actually the HEX_LOADER routine that Simon Goodwin uses in DIY Toolkit articles for those who don't have assemblers. (Why should I re-invent the wheel - and an inferior one - when someone else has already produced the definitive one? Thanks to Marcus and Simon.) The difference is that I have added REMark statements, to show which assembler code produced which numbers.)

```
100 REMark Sinclair QL World HEX LOADER v 3
110 REMark by Marcus Jeffery & Simon N Goodwin
120 :
130 CLS: RESTORE :READ space:start=RESPR(space)
140 PRINT "Loading Hex...":HEX_LOAD start
150 INPUT "Save to file...";f$
160 SBYTES f$,start,byte:STOP
170 :
180 DEFine FuNction DECIMAL(x)
190 RETurn CODE(h$(x))-48-7*(h$(x)>"9")
200 END DEFine DECIMAL
210 :
220 DEFine PROCedure HEX_LOAD(start)
230 byte=Ø:checksum=Ø
240 REPeat load_hex_digits
250    READ h$
260    IF h$="*":EXIT load_hex_digits
270    IF LEN(h$) MOD 2
280       PRINT "Odd number of hex digits in: ";h$
290       STOP
300    END IF
310    FOR b=1 TO LEN(h$) STEP 2
320       hb=DECIMAL(b):lb=DECIMAL(b+1)
330       IF hb<Ø OR hb>15 OR lb<Ø OR lb>15
340          PRINT "Illegal hex digit in: ";h$:STOP
350       END IF
360       POKE start+byte,16*hb+lb
370       checksum=checksum+16*hb+lb
380       byte=byte+1
390    END FOR b
400 END REPeat load_hex_digits
410 READ check
420 IF check<>checksum
430    PRINT "Checksum incorrect. Recheck data. ":STOP
440 END IF
450 PRINT "Checksum correct. Data entered at: ";start
460 END DEFine HEX_LOAD
470 :
480 REMark Space requirements for the machine code
490 DATA 28
500 :
510 DATA "13FCØØFFØØØ21ØØØ": REMark          MOVE.B     #$FF,$21ØØØ
520 DATA "33FCFFFFØØØ22ØØØ": REMark          MOVE.W     #$FFFF,22ØØØ
530 DATA "23FCFFFFFFFFØØØ23ØØØ": REMark      MOVE.L     #$FFFFFFFF,23ØØØ
540 DATA "4E75": REMark                      RTS
550 DATA "*",2943
```

Small example

When you run the SuperBasic program in Listing two, if you have done everything properly, the screen will display a short green line, a short white line, and a longer white line. Unfortunately, it will also produce a 'Bad parameter' error, and the program will stop. There is no mistake here. To avoid this, we need an extra line of code, but this requires a bit more explaining, and I am going to leave this for later.

The effects the program produce on the screen are not much to look at, and may seem quite meaningless. It is very difficult to get much to see using such a small amount of code, but at least here we can see something happen. When we learn a bit more, we can start to do more interesting and meaningful things on the screen. Why we get these particular effects needs an understanding of how the QL interprets the data in the screen ram, and this will be explained later. Suffice to say that, with different numbers, the effects are slightly different when the program is run in Mode4 and Mode8. If you're curious about this, perhaps you might like to experiment by trying different numbers at different addresses.

Happy coding!

DIY TOOL*kit*

Simon Goodwin describes the second part of FLEXYNET, the fast network link for QLs and other machines.

Flexynet is a new fast network protocol. Last month I presented the first part of the code. Now I can reveal the rest of the assembler source for QL systems, and the nitty-gritty details of the protocol.

Flexynet is machine and operating-system independent, and requires the minimum of hardware - a single bit for input and output - so it can be used to link all sorts of computer, from QL and Thor to ST, Amiga, SAM and even PCs.

The speed is completely configurable at both ends of the link. The transmission and reception values count the number of times Flexynet goes round an internal waiting loop, so larger values mean longer delays.

Transmission delay values are greater than reception delays, as the receiving process is fundamentally slower. Thus a slow QL can talk to a fast one more quickly than the fast one can answer back intelligibly. This should come as little surprise.

Best results

Naturally you get best results with a Gold Card at each end of the link, but even Sinclair's humble 7.5 MHz 68008 can transmit at 6K per second if the receiving machine is a 16MHz Gold Card. This is three to six times faster than RS-232 or the Sinclair network protocol. I used NETRATE 2 to send and NETRATE 7 at the receiving Gold Card. Going the other way, use NETRATE 19 on the Gold Card and NETRATE 2 (faster than between QLs) on the 8 bit receiver.

Details of the Flexynet commands, and a program to create the code, appeared in QL World last month. If you missed the issue, or would like the complete source, code and documentation on QL disk, plus extra files, write to *DIY Toolkit* at Cwm Gwen Hall, Pencader Dyfed, Cymru SA39 9HA, or call 0559 384574.

The Flexynet collection makes up Volume Y, one of a score of volumes based on the long-running DIY Toolkit series. Volumes cost three pounds each, with a processing charge of four pounds per order, so it pays to order several at once. Consult the December Quanta newsletter or send a stamped addressed envelope if you need more details of the volumes available.

New thing

This is the first DIY Toolkit project to make use of the 'thing' list, a late addition to the Qdos system variables. The Flexynet thing records the current network transmission and reception rates. You can keep track of it, and other such things, with Dario Leslie's forthcoming 'Qdos Interrogator' or the 'Thing Manager' from Jochen Merz.

QJump's QPTR environment includes special TRAPs to create and access things, but you cannot use these unless QPTR is loaded. The DIY Toolkit

```
* DIY TOOLKIT FLEXYNET PART 2 - Copyright 1993 Simon N Goodwin.
*
* NETREAD address,length
*
net_read    bsr       find_thing2      Look for Thing address
            bne.s     release          Not found - give up!
            lea.l     net_input,a0     Point at the hardware
            move.w    4(a4),d2         Find slowest valid time
            move.w    d2,d3
            sub.w     2(a4),d3         Pick up 1 bit RX time
            bls.s     timeout          Too short!
*
start       movem.l   d5/d7,-(a7)      Preserve length and TX_MODE
            lea.l     key_spec,a3      Point at KEYROW(1) command
            moveq     #17,d0           Fetch MT.IPCOM trap key
            trap      #1               This uses D0, D1, D5, D7, A3
            movem.l   (a7)+,d5/d7
            and.b     #8+64,d1         ESC or SPACE bit set?
            beq.s     no_key
timeout     moveq     #-1,d0           Report not complete
            bra.s     release
no_key      btst      #net_inbit,(a0) Wait for block start bit
            beq.s     start            Loop till ESC or net busy
start_poll  btst      #net_inbit,(a0)
            bne.s     start_poll       Wait for a LOW level
*
* Receive D5 bytes from the port at (A0) into memory at (A2)
*
read_byte   down_bit timeout
            up_bit   timeout
            down_bit timeout
            up_bit   timeout
            down_bit timeout2
            up_bit   timeout2
            down_bit timeout2
            up_bit   timeout2
            down_bit timeout2
            move.b    d4,(a2)+         Store byte
            read_high                  Stop bit, length ignored
            subq.l    #1,d5
            bne       read_byte
finished    bra       it_worked
timeout2    bra       timeout          Extra exit; timing error
*
* NETSEND - prepare to transmit bytes
*
net_send    move.b    d0,d6            Save ZX-8302 TX mode
            bsr       find_thing2      Look for Thing address
            bne       release
            move.b    d6,d0            Restore ZX-8302 TX mode
            move.w    (a4),d6          Pick up TX delay
            swap      d6
            move.w    (a4),d6          Even mark/space ratio
            move.b    d0,d1            D0 is NET OFF pattern
            bset      #net_outbit,d1   D1 is NET ON pattern
            move.b    d1,tx_control    Activate net hardware
            move.w    #6000,d2         Wait a while for sync
get_ready   dbra      d2,get_ready
byte_loop   move.b    (a2)+,d4         Pick up data byte
            move.b    d1,tx_control    Send start pulse
            move.l    d6,d3
on_loop     dbra      d3,on_loop
            send_low
            send_high
            send_low
            send_high
            send_low
            send_high
            send_low
            send_high
            swap      d3
            move.b    d0,tx_control    Send stop pulse
off_loop    dbra      d3,off_loop
            subq.l    #1,d5            Count down bytes
            bne       byte_loop
            move.b    d1,tx_control    Terminal bit
last_bit    dbra      d6,last_bit
            move.b    d0,tx_control    De-activate network
            bra       it_worked
*
* NETRATE tx_time% , rx_time% , timeout%    (0 means no change)
*
set_rate    movea.w   $112.w,a2        Pick up CA.GTINT vector
            jsr       (a2)
            bne.s     no_good          Continue only if D0=0
            subq.w    #3,d3            Three parameters are needed
            bne.s     bad_param2
            trap      #0               Supervisor mode, fix A6
            movea.l   a1,a2            Save the RI stack offset
            bsr.s     find_thing
            bne.s     tidy_up
            move.w    0(a2,a6.l),d1    Check first parameter
            beq.s     default_tx
            move.w    d1,(a4)          Store TX speed
default_tx  move.w    2(a2,a6.l),d1
            beq.s     default_rx
            move.w    d1,2(a4)         Store RX speed
default_rx  move.w    4(a2,a6.l),d1
            beq.s     tidy_up
            move.w    d1,4(a4)         Store Timeout
tidy_up     bra       cheer_up         Revert to USER mode
bad_param2  moveq     #-15,d0
no_good     rts
*
* value%=NETVAR%(index%) - read NET time constant 1, 2 etc.
*
```

implementation emulates the standard format, using standard system calls that work on all QL roms, with or without QPTR. If you plan to put any new code in rom, where position-relative variables are impossible, these DIY thing routines will be right up your street.

If you re-assemble the Flexynet code you will find one difference between the Listing and last month's hex code. The name of the Flexynet thing is now held in lower case, to suit existing thing managers which fail to recognise capitals. I did not discover this until after last column was submitted. Flexynet itself uses a case-insensitive search, so it can recognise names in capitals or small letters.

The change from version 1.08 to 1.09 means that the thing can be removed by a THNG_REMV command. If you would like this change, but do not wish to re-assemble the source, you can patch the name from "FlexyNet" to "flexynet" with a utility like *Spy* or *The Editor*.

The listing

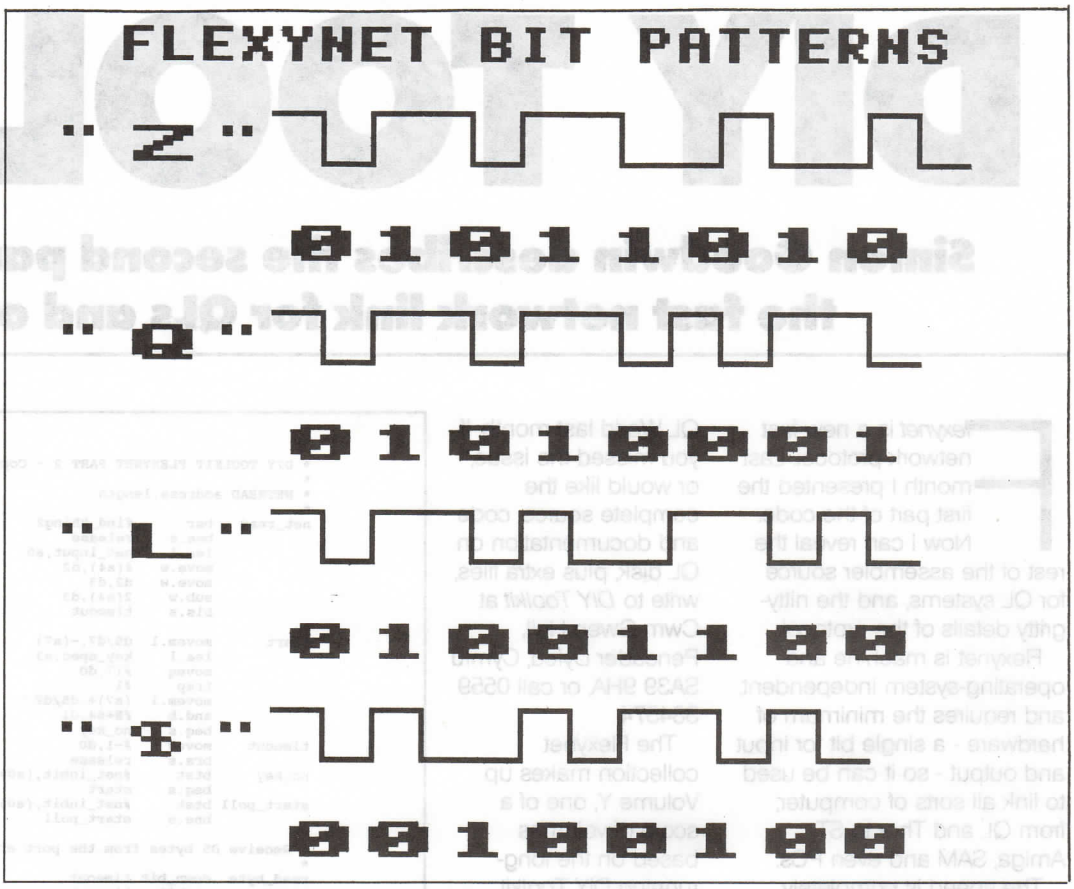
The accompanying listing completes the source for Flexynet version 1.09. It includes the important BYTE_LOOP and READ_BYTE routines that send and receive characters over the network.

NETREAD waits for the network level to become high, polling the keyboard in case Esc or Space are pressed in the meantime. The MT.IPCOM trap and the list of parameters at KEYSPEC mimic the SuperBasic KEYROW command, returning a value in D1.

NETSEND starts by sending a relatively long pulse on the network. The constant #6000 just before the GET_READY loop ensures that even a Gold Card will wait long enough to allow the original Sinclair rom and IPC to poll the keyboard for Esc without missing the start of the message. It corresponds to a delay of about 20 milliseconds on a slow QL, which may take 5 milliseconds to detect a key-press.

A factor of four is enough to keep Gold Card and 68008 in step. It is hard to imagine that future systems will handle their keys any more slowly. QL KEYROW is a thousand times slower than the equivalent on machines with memory-mapped keys, like the ZX Spectrum.

You can speed up the transfer of short messages by reducing this delay if you use Hermes and a recent version of Minerva, or all the processors on your network run at about the same speed. You may need to increase it if your processors differ in speed by a large factor, but Flexynet will need changes in any case if it is to run on a machine with a 68020 or later processor with cache memory.



Timing

The time-critical work is done by macros introduced last month. UP_BIT and DOWN_BIT wait for an appropriate change in the level at the net port, counting as they wait. A timeout error is reported if the count runs out before a change is sensed. Otherwise the loop count is compared with the limit in D3, and a one or zero bit is shifted into the least significant bit of D4.

The first bit time is ignored, to ensure that the receive and transmit loops are synchronised. After the eighth data bit is received the byte is stored, then the READ_HIGH macro waits for a final 'stop' pulse. If the count of bytes remaining in D5 is not yet zero, Flexynet loops back to READ_BYTE to pick up the next.

Notice that some of the bit input macros take the TIMEOUT label as a parameter, while others use TIMEOUT2. In each case the effect is the same - Qdos reports ERR.NC - but both labels are used so that all the macros can reach one or other label with a short (8 bit) branch offset. There would be no need for the second label if long (16 bit) branches were used, but they would reduce the maximum input rate.

SEND_LOW and SEND_HIGH macros in BYTE_LOOP generate timed pulses for NETSEND. The delay count for a bit value of one is twice that for a zero bit. In fact the difference in length is slightly less, after accounting for the other instructions in the macro and the way DBRA counts down to -1, but one and zero bits are still easily distinguishable once the correct reception threshold is set.

The diagram shows the pattern of pulses corresponding to two byte values. Notice that it is the distance between the changes of level, rather than the level itself, that determines the value of each bit.

Easy Bit

The code for NETRATE and NETVAR% is very straight-forward. They start by calling the CAGTINT vector to fetch integer parameters, obtaining a parameter count in D3. Once this has been checked they call FIND_THING to locate the Flexynet thing variables, and exit via TIDY_UP if an error occurs. Supervisor mode is selected with TRAP #0, so other jobs cannot manipulate the thing list while Flexynet is using it.

NETRATE copies its three parameters into the thing variables addressed by A4; if any parameter is zero the corresponding old value is left unchanged. NETVAR% doubles its parameter and uses it as an index into the table at A4. The offset -2 ensures that references to NETVAR%(1) refer to the first word in the table. There is no check on the value of D7, so NETVAR% can also read words from before or after the space currently used.

The result from NETVAR% is moved to the space previously occupied by its parameter, so there is no need to check that extra stack space is available.

Real things

The FIND_THING subroutine looks through the thing list for the Flexynet thing. If it cannot be found it attempts to allocate it on the common heap, returning ERR.OM from MT.ALCHP if there is no spare memory.

The Flexynet thing contains only six bytes of data, yet a further sixty bytes are needed to declare the thing in standard form. Many of these bytes can be left unset, as MT.ALCHP returns them, but others are set from the table labelled THING_SPEC. The default timing values are held in three words at THING_DATA, and could be patched to suit your hardware.

I have started by developing a simple driver for the rather peculiar hardware of the standard QL I plan to develop that prototype into a fully fledged Qdos device driver, with variants for other machines. The aim is to get Flexynet working on all pairs of machines with fast enough memory, then add bells and whistles.

Flexynet can expand into a Qdos device driver that lets standard commands read and write over the net in a device-independent manner. For the time being there are plenty of new ideas in this month's installment, and more to come in future columns.

The ZX-8302 version of Flexynet works well, but it is just a start. Besides support for other ports and machines, potential extensions include device-independence, checksums, Handshaking and collision checks, faster data formats (eg RLL), automatic speed sensing and remote file serving. Please let me know how your systems perform, and what you would like to see next, care of *QL World*.

```
net_value  movea.w   $112.w,a2         Pick up CA.GTINT vector
           jsr       (a2)
           bne.s     no_good           Continue only if D0=0
           subq.w    #1,d3             One parameter is needed
           bne.s     bad_param2
           move.w    0(a1,a6.l),d7     Pick up parameter
           ble.s     bad_param2        It must be greater than 0
           add.w     d7,d7             Form word index
           trap      #0                Supervisor mode, fix A6
           bsr.s     find_thing
           bne.s     tidy_up           Oops; return D0 error code
           movea.l   $58(a6),a1        Recover RI stack offset
           move.w    -2(a4,d7.w),0(a1,a6.l)
           moveq     #3,d4             Return an integer
           bra.s     tidy_up
*
* Locate the FlexyNet Thing; create it if necessary
*
find_thing moveq     #0,d0             MT.INF trap key
           trap      #1
find_thing2 movea.l  a0,a4             A4 -> System variables
           lea.l     sv_thingl(a4),a5

           lea.l     thing_name,a0     A0 -> required name
           suba.l    a6,a0             Make it relative
           movea.w   $e6.w,a3          Preset UT.CSTR vector
next_thing move.l    (a5),d0           Check list pointer
           beq.s     make_thing        End of list found?
           movea.l   d0,a5             A5 -> Thing base
           lea.l     42(a5),a1         Point at its text
           suba.l    a6,a1             All is relative...
           moveq     #1,d0             Ignore letter case
           jsr       (a3)              Call UT.CSTR
           tst.l     d0                Do names match?
           bne.s     next_thing
got_thing  lea.l     60(a5),a4         A4 -> Thing values
           rts                         Return OK, D0=0, Z set
*
make_thing moveq     #24,d0            MT.ALCHP trap key
           moveq     #60+6,d1          Total space required
           moveq     #0,d2             Owner is SuperBASIC
           movea.l   a2,a5             Save A2 (SEND/READ base)
           trap      #1
           movea.l   a5,a2             Restore A2
           tst.l     d0                Did ALCHP work?
           bne.s     done_thing        If not, return NZ error code
           lea.l     52(a0),a5         Point at Thing data values
           move.l    a5,16(a0)         Set pointer to Thing values
           lea.l     thing_spec,a5     Point at ASCII header
           lea.l     38(a0),a3         Point at destination
           moveq     #(thing_end-thing_spec)/2-1,d1
set_header move.w    (a5)+,(a3)+       Copy one header word
           dbra      d1,set_header     Initialise the rest
           lea.l     sv_thingl(a4),a5
           move.l    (a5),(a0)         Extend the linked list
           move.l    a0,(a5)           Update the list start
           lea.l     60(a0),a4         Make A4 -> Thing's values
           moveq     #0,d0             Set Z flag; ERR.OK
done_thing rts
*
thing_spec dc.b      '1.09'            Version number
thing_name dc.w      8                 Name length
           dc.b      'flexynet'        Thing name
           dc.b      'THG%'
           dc.l      2                 Shared data follows
thing_data dc.w      5,3,127           TX, RX, & Limit times
thing_end
*
key_spec   dc.b      9,1,0,0,0,0,1,2 KEYROW(1) IPC message
*
define     dc.w      5                 Number of procedures
           dc.w      send_bits-*
           dc.b      7,'NETSEND'
           dc.w      netbeep-*
           dc.b      7,'NETBEEP'
           dc.w      netpoll-*
           dc.b      7,'NETPOLL'
           dc.w      read_bits-*
           dc.b      7,'NETREAD'
           dc.w      set_rate-*
           dc.b      7,'NETRATE'
           dc.w      0,1               End procedures, one FN
           dc.w      net_value-*
           dc.b      7,'NETVAR%'
           dc.w      0                 End of functions
           end
```

Ian Bruntlett explains a significant upgrade to QL usability.

The POINTER ENVIRONMENT

This article describes, explains and demonstrates the Pointer Environment while assuming only a little QL knowledge.

In the future I will cover Qpac2 and other things there are no room for here. Some explanations are in the Appendices at the end of this article. Inexperienced users may wish to refer to the appendices while they are still gaining their "QL-legs".

If you know next to nothing about the QL, I suggest you browse through the *QL User Guide* (don't worry if you don't understand it) and gossip with some people who know about computers - you may pick something up. See Appendix "Commands you'll find useful" for the SuperBasic commands to use when taming your QL in the future.

Meetings

A good source of QL people is a local Quanta meeting. Non-members are often welcome, and you will find out more about Quanta and your QL.

The novice QL user is likely to get more from attending Quanta meetings than most people. Most of a novice user's problems will have been experienced and solved by the people there, so don't be afraid to ask questions. No-one in computing knows everything - be wary of people who reckon they do.

The QL uses the Qdos operating system and its SuperBasic interpreter and (when new) comes bundled with the four Psion programs - *Quill, Archive, Abacus, Easel* and (sometimes) a games microdrive cartridge. No disk drives, printers or extra software were ever supplied with the standard set-up by Sinclair.

This presents some interesting problems. You can do plenty of useful work on an unexpanded QL with just the Psion programs, but this does not do the machine justice.

Disk drives

Disk drives are faster than microdrive cartridges, more reliable and, when storing large amounts of information, much cheaper. A floppy disk costing less than 50p holds 720 kilobytes (KB) of information while a microdrive cartridge holds just over 100KB and costs about £2. In real terms, when you fill a single disk with information it is at least £10 cheaper than storing the same amount of information on microdrive cartridges. It is worth having a pair of reliable disk drives - cheap, unreliable disk drives are not worth the trouble they cause.

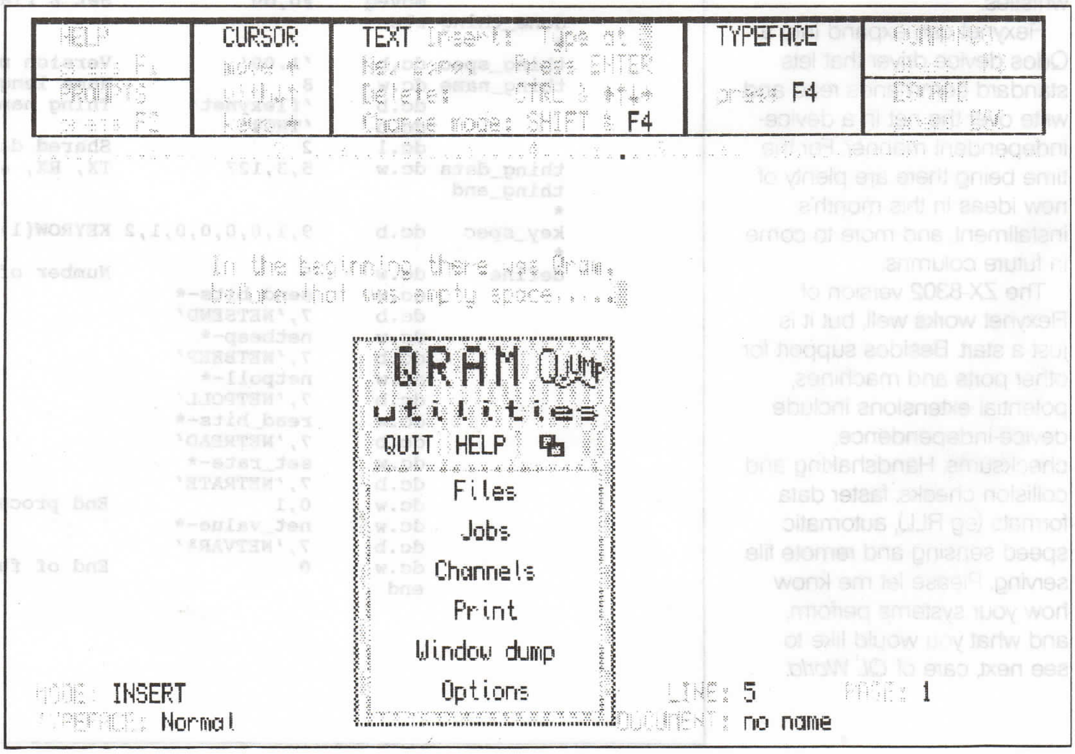
Additional memory allows you to edit longer documents with greater ease and have more than one program in memory. A unique feature of Qdos is that it uses "free/spare memory" to speed up file access, remembering the contents of files it has recently handled.

On my Gold Card QL with plenty of free memory (1252KB at the moment), file handling can almost be too fast. This happened when I was converting some C68 manuals into *DataDesign* files. The manuals were converted from Quill documents to text files and saved on my slow 5.25-in drive. A small SuperBasic program read the work files and produced a DataDesign database. The text files had to be edited and re-saved occasionally because the SuperBasic program sometimes became confused. So a loop of "write text file", "process text file", "find a problem", "fix it" and go back to "write text file" started. This was faster on a Gold Card because when "write text file" was done, Qdos had enough free/spare memory to keep a copy of the text file in memory. When the "process text file" started, Qdos checked that the disk had not been changed and just used the memory-based copy of the text file. This process of keeping copies of files in free memory without any involvement of the user is called "slaving" in the *QL Technical Guide*.

This tactic of the QL may seem like common sense but no other operating system does this as well as Qdos - other operating systems insist on having a fixed amount of memory for buffering. I use an improvement of Qdos, Minerva, which has given me no problems with Gold Card - it has been said that the Sinclair versions of Qdos don't take as much advantage of the Gold Card as Minerva. I have no intention of messing about with a Trump Card and a Sinclair version of Qdos to verify this.

Hardware

Back to the Pointer Environment. This needs a QL with at least 384KB of memory, a monitor, power supply and a



TEXT Insert: Type it
New paragraph: Press ENTER
Delete: CTRL & ←/→
Change mode: SHIFT & F4

In the beginning there was Qram, and the void was empty space....

MODE: INSERT
TYPEFACE: Normal
LINE: 5
PAGE: 1
DOCUMENT: no name

copy of a program called 'ptr_gen'. What an anti-climax! After the last paragraph you were beginning to think you'd need a second mortgage to afford all the kit.

Multi-tasking.

This is where a computer can have more than one program in memory and split its working time between the different programs (jobs). A bit like a secretary answering a phone, writing a memo using a word-processor, drinking coffee and chatting with a friend.

The QL's multi-tasking does not have "virtual machine" protection so if a program crashes, it can affect all the programs in your computer - the equivalent of the secretary swearing at a telephone caller instead of the photocopier and dropping the coffee on the word processor. "Virtual machine" protection on a micro-computer is very unusual - OS/2 on a 486/386 PC has it but Qdos has yet to take advantage of the 68030 or 68040 where virtual machines may be possible.

This lack of protection really is as drastic as it seems. When working with a computer you should save data regularly, make sure you have at least one back-up and gently introduce new software into your work.

It is no good to take an isolationist point of view with software, buying some software, getting that working and sticking with that permanently. Ignoring the rest of the world put the Chinese Empire at a disadvantage, so don't make the same mistake. Try to be aware of what is on offer and get friends to show you new programs they have bought.

Window-wipe

Before the Pointer Environment was introduced, the QL had "destructive windows". When multi-tasking, the user would swap from one program to another and would usually press a key (F4 or Shift-F5) to get the program to re-draw its screen. Unfortunately this would demolish the displays of the other programs. The result was a mess, but it worked and was economical with memory. The QL would multi-task and the user would just have to wait for the screen to be re-drawn. This mutually assured screen destruction worked as long as each program could re-draw its windows and had an active cursor. Many programs couldn't re-draw their windows, and many didn't have an active cursor. Those programs that didn't have an active cursor could sometimes "lock out" the user when involved in multi-tasking.

This state of affairs could not go on forever. It is easy enough to re-draw the screen of a word-processor but very inconvenient to re-draw that of a Fractal Generation program just because something else has corrupted its display.

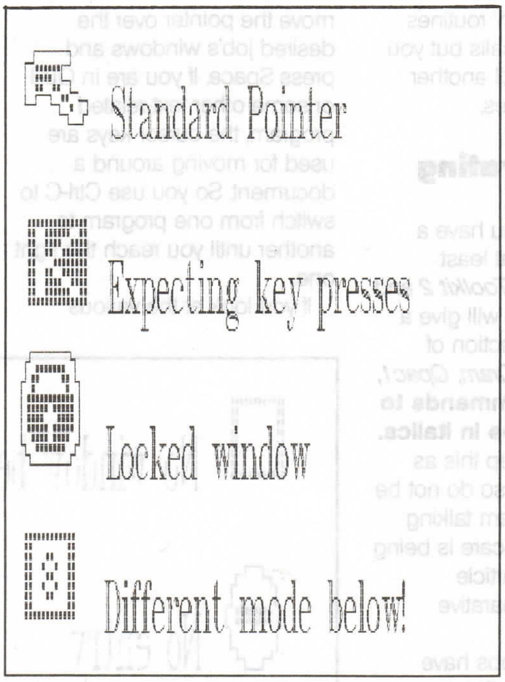
The Pointer Environment improves the QL's handling of the screen so that windows are preserved automatically. This was not done when the QL was first brought out because there wasn't enough memory inside an unexpanded QL and expansion memory was expensive (Roughly £1 per KB - at those prices Trump Card would be £900-plus, instead of £95 plus P&P from Qubbesoft PD).

The Pointer

The Pointer Environment was introduced in 1987 with Qjump's Qram **(Figure one),** a software package intended to take advantage of expanded QLs. The Pointer Environment stopped programs from over-writing each other's windows and added the use of a pointer device (usually a mouse) at a time when mice were becoming very popular on other machines.

Because most the mouse supply to QLs was still limited, the Pointer Environment was written to be used with either a mouse, or the keyboard, or both. Try using Microsoft Windows on a PC or Open-Windows on a Sun work-station without a mouse and you will appreciates that this "mouse independence" is no mean feat. *(The man speaks true, say Mac and PC users.)* On some systems, if your mouse breaks down the whole system is virtually useless.

The Pointer Environment had some other facilities for Qram - it introduced standard routines for handling images. Little symbols could be defined as sprites and



put on the screen without the programmer having to think about how the data should be created (sprite editors are available) or how to display the image. The "Qjump" logo on the Qram menu is a sprite. This is different from other computers which usually have sprites flying all over the screen, changing colours and blasting each other to pieces. The Pointer Environment sprites (apart from the cursor sprite) don't move.

The "pointer" in pointer environment refers to the cursor sprite which the user can move about the screen using a mouse or the keyboard. The shape of the pointer sprite/picture actually changes according to the state of your QL (Figures two and three). If I move my QIMI mouse while typing this article into text87, a picture of a "K" appears, telling me that the program underneath the pointer is waiting for keyboard input. If the pointer sprite was a red "no-entry" sign then it would mean that the program under the pointer does not want any keyboard input.

Compatibility

If a piece of software is described as "incompatible with Qram", you can be sure that it is conflicting with 'ptr_gen' from the Pointer Environment. The conflict is sometimes blamed on the *Window Manager*, 'wman'. This is nonsense.

In the Pointer Environment, 'ptr_gen' is the active partner, extending the system. The 'Window manager' is a set of routines for handling a consistent window/pointer-based interface that uses 'ptr_gen' to handle the mouse pointer, keyboard, preserving windows and drawing sprites. The Window Manager only manages individual windows, telling the programmer what the user is doing - all the nitty-gritty is done by 'ptr_gen'.

Technical

Qjump used the Window Manager to give a mouse-driven interface for Qram. So that other programmers could use the Pointer Interface and Window Manager routines, Qjump produced *Qptr*. As a reference guide to 'ptr_gen' and 'wman', Qptr is good. If you know about what you want and where you will find it, the Qptr manual will be very easy to use. For someone who is learning something new about Pointer programs, Qptr is a complicated manual that pushes you back on your own resources. Qptr provided some complete programs as examples (a sprite editor, a paint program, and a 'demo').

A Utopian Qptr would also have had many small examples with explanations. Each demo could have shown an individual aspect of either the Pointer Interface or the Window Manager. Qmon would have been updated to help when debugging Pointer programs.

For years I laboured under the idea that the Window Manager made life hard for the programmer to make life easier for the user. Now, I am being told that inexperienced users are finding pointer-driven applications hard to get started on. The Window Manager is responsible for drawing the mouse-driven windows of most Pointer Environment applications. It is possible to bypass the Window Manager and call the 'ptr_gen' routines directly with TRAP calls but you will have to wait until another episode for examples.

Demonstrating

Assuming that you have a disk drive, QL with at least 384KB of memory, *Toolkit 2* and a pointer program, I will give a brief demo of a selection of pointer programs (*Qram, Qpac1,* Quanta library). **Commands to be typed in will be in italics.**

I have tried to keep this as simple as possible so do not be upset if you think I am talking down to you. Great care is being taken to keep this article accessible to comparative novices.

In the QL, most jobs have windows. When the Pointer Environment is installed, it is possible to move from one job's windows to another by pressing Ctrl-C.

When using a QIMI mouse (the QIMI interface and mouse is now available from Quanta), it is possible to select a job by moving the pointer (via the mouse) from the current window to another job's windows and pressing the Left mouse button. QIMI is the best interface for this as it can move the pointer even if *Quill* is being used.

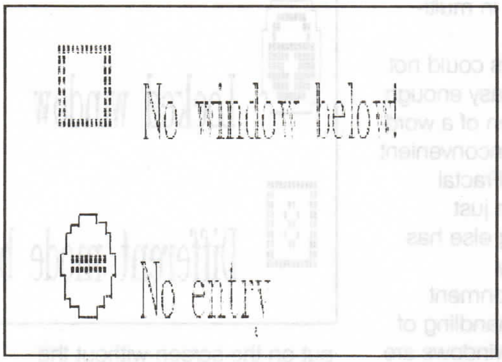
There is a serial mouse interface available which supports the use of the third (Middle) button. This usually acts as Quit/Escape. When pressed simultaneously with the Left button, it emulates "Wake". When pressed simultaneously with the Right button, it emulates "Sleep".

Key and Mouse

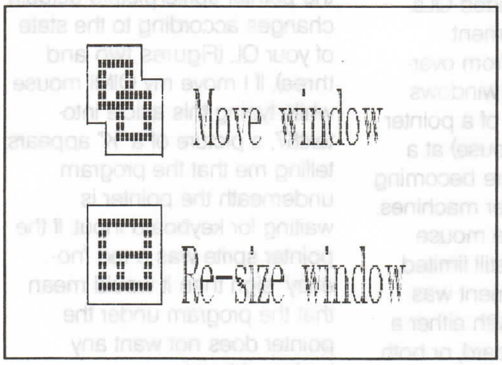
The QL's keyboard can emulate its mouse quite well - the cursor keys will move the pointer arrow and the mouse buttons are emulated by the Space bar (Left button) and Enter (Right button). The pointer environment, from version 1.23 of ptr_gen onwards, has commands to allow cursor key movement of the pointer to be disabled (CKEYOFF) and re-enabled (CKEYON).

If you are using the Pointer Environment without a mouse, then the pointer can be moved to another window with the cursor keys. To select a job under these conditions, just move the pointer over the desired job's windows and press Space. If you are in Quill or some other text-related program, the cursor keys are used for moving around a document. So you use Ctrl-C to switch from one program to another until you reach the right one.

If you look at the various screen dumps of pointer programs you should notice similarities. They have a title, a scattering of options at the top of the window and (usually) a display specific to the task in hand - a list of files, a list of jobs, etc.



Referring to **Figure one,** the "Quit", "Help" and "picture of two little squares" options may be selected by moving the pointer over them. When the pointer moves over an option, that option becomes the "current option". The "current option", if any, will be outlined in either black or white. Selected options that affect the status of a program (eg Sound On in a game) appear in a different colour, according to their status On/Off, etc. If an option is unavailable then this will usually appear in a subdued colour and you will not be able to select it.



Subtle

Pointer programs handle their options in a very subtle way. The Qram manual gives good guidance on these points.

To Do something from a menu:

A "do" is achieved by hitting the Enter key or Right mouse button. A "do" will select an option, and usually invoke any associated function, such as file copy. A "do" on an option may also be put into action by typing a single character, usually the first letter (such as J for JOBS), or an under-scored letter (such as N for "Save with New name"). QUIT breaks this rule by using ESCape. Some options may not have this feature - filenames and the Format option are examples. (From the Qram Files menu.

To change the status of an option:

A "hit" on an option is achieved by moving the pointer to the option using the mouse or the cursor keys, and pressing Space or the Left mouse button. A hit will toggle an option's status between available and selected.

The above definitions are best used when playing around with a pointer program. The difference between a Hit and a Do is important, but don't try to learn it by rote. Just get used to the Pointer Environment and you will "learn" it naturally.

The Sleep and Wake/Update options are not supported by all Pointer programs. They were introduced to users when *Qpac2* was launched. See **Figure six** for the Sleep and Wake symbols.

To put a job to sleep

Press Ctrl-F1 or Hit the "Zzz" symbol **(see Figure six).** This hides the job's windows and it is stopped. A button appears in the button frame with the program's name. The button frame is specific to Qpac2 and will be explained, with Qpac2, later in the series.

To update a window (Wake)

Press Ctrl-F2 or Hit the "jagged lightning strike" symbol **(see Figure six).** The windows will be brought up to date. This would be used in a jobs menu, for instance, to update its jobs list if the user had quit some of the programs and returned to the jobs menu.

To Resize a job's windows

This option has been supported since Qpac1 came out **(See Figure four,** "Re-size window"). *Press Ctrl-F3.* Programs handle re-sizing of windows in three different ways. (Most QL programs ignore requests to change the window size.)

Many Pointer programs will change the pointer arrow to a Bizarre Looking Blob (BLB) when changing the window size. The pointer moves to a new part of the screen and the new window size follows the pointer position (a process widely known as "dragging"). If you move the pointer to a new position above the old position, for instance, the window will be taller. If the pointer is moves to the left of the old position, the window is wider. If you move inwards or down, the window will be smaller.

Other Pointer programs, in particular the *Qpac1 Calendar,* have a distinct set of window formats. Telling this kind of program to re-size its windows will result in the next window format in the cycle being displayed. The Qpac1 Calendar goes from a medium sized Calendar (displaying Clock and Month) to a large sized calendar (displaying Clock, Month and extra displacement info) to a tiny clock, not showing the days of the month at all.

To Move a job's windows

This option has nearly always been supported (see Figure four, "Move Window"). *Press Ctrl-F4.* The pointer will change into a "double block" symbol. *Moving the pointer and pressing Hit, Do or Escape will move the window to the pointer's current position.* It is possible that Escape will one day abort the move window option, so don't get into the habit of using it to complete a "move window" operation.

Freezing the display

This is available without the Pointer Environment. *Press Ctrl-F5.* It delays any attempt to print anything on the screen until Ctrl-F5 or another key is pressed.

If the Pointer arrow is visible when Ctrl-F5 is pressed then the pointer will turn into a padlock (See **Figure two,** "locked window").

Preparing for the demo

Switch on your QL,

Press F1/F2 according to your display type

If you pressed F2 type in the command MODE 4 and press Enter

Put the appropriate disk in disk drive one.

DEMO 1: With Qram

TK2_EXT
DATA_USE FLP1_
PROG_USE FLP1_
LRESPR FLP1_Boot_REXT

This loads PTR_GEN, WMAN and Qram itself.

Press the Alt key and / simultaneously, and you should get the Qram menu, a little rectangle, appearing on the screen (Figure one). There should be a little arrow on screen (the Pointer) which you can move around with a mouse or the cursor keys. When the pointer moves over an option that can be selected, the option is highlighted with a little white box. Qjump's manual calls this the "current item" (page 9).

Move the pointer around the screen using the cursor keys. When the pointer is out of the Qram window, it changes to a padlock, which means that the window below the pointer is "locked" because it is buried by another window, in our case Qram's window. A locked window cannot be updated until it is unlocked, usually by "un-burying" (exhuming?) its windows. To do this, *move the pointer to the locked window and Hit it by pressing Space or the Left mouse button.* The selected job will be brought to the top of the pile of windows and any keypresses will be sent to it.

Psion

To multi-task Psion programs with Qram, you need to process your copy of Quill, etc., with the program *Grabber.* Simply put your Qram disk in drive one, enter the command EX FLP1_GRABBER, then put your Quill disk in drive two. Follow the instructions on Page 19 of the Qram manual. They worked for my *Psion XChange* a long time ago. Grabber will modify Quill so it will behave itself properly. If you save the new Quill to FLP1_GRQUILL then EXEC, FLP1_GRQUILL will run your new Quill. This process can be applied to Abacus, Archive and Easel. The "Grabbed" copies of the Psion programs cannot be configured with the Psion CONFIG program but there is a program by Rich Mellor, "CONFIGURE", that will - it is on the Quanta library disk PSION_1. I am unable to verify that "CONFIGURE" works as my copy of PSION_1 seems to have disappeared!

The Qram program HOT_MAKE will set up a file of programs that can be resident in memory and called up with a Hotkey. Simply put the Qram disk in FLP1_ and follow the instructions. It cannot put normal SuperBasic programs on a hotkey but it can put most QL jobs (started with an EXEC or EX command) on a hotkey.

Having the Psion programs behaving themselves and running from a hotkey is fine, but why not have them automatically LOAD or have them in memory all the time? It is possible to do all this with the Hotkey System Two (hot_rext). This was developed as the successor to Qram's HOT_MAKE and is much more flexible.

Qram is now obsolete - when Qram was replaced by Qpac2, old Qram users were allowed a discount when upgrading to Qpac2. But Qram had a few facilities that Qpac2 doesn't - it had a screen dump menu and the ability to automatically configure itself. Qpac2 has many things that Qram does not have and there are ways for a Qpac2 user to get round the missing facilities. See the Qpac2 demo in a subsequent article for more details.

DEMO 2: With QPac1

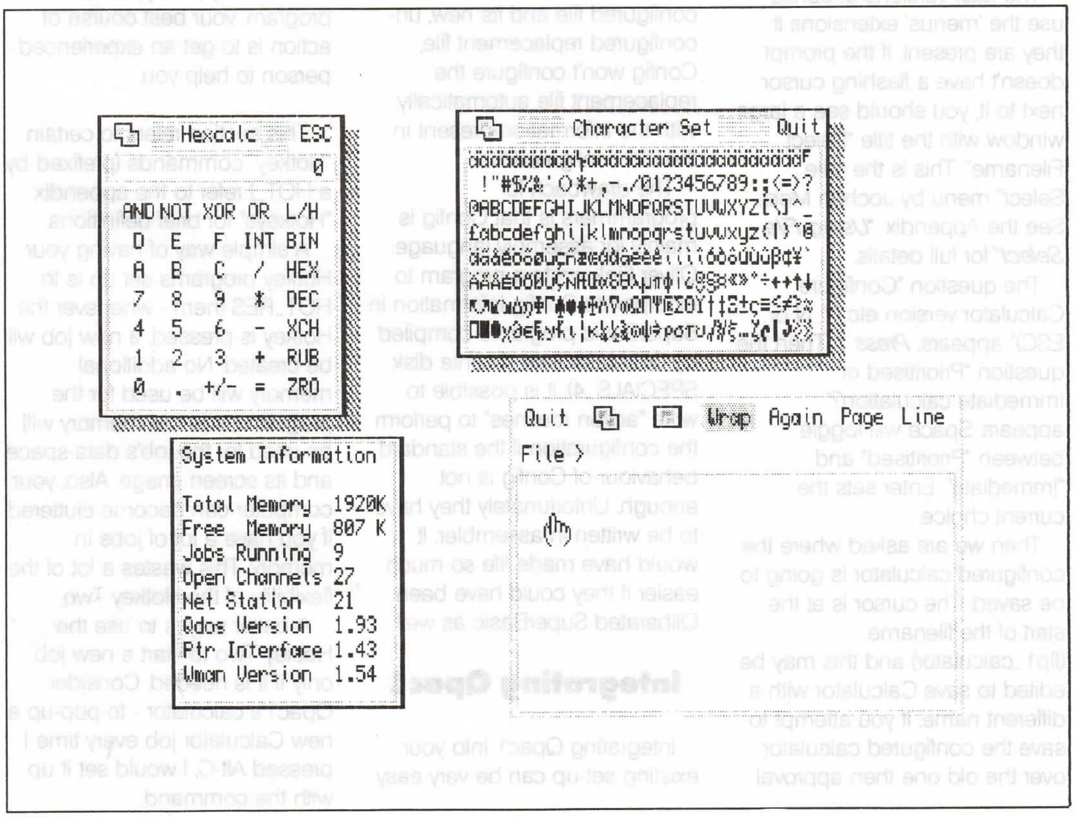
Qpac1 introduced the Hotkey System Two, a new CONFIG program and a set of six utilities (Calendar, Clock, Alarm Clock, Calculator and System Monitor). Because Qpac1 offers more flexibility, it is easy to be overwhelmed by the choices when trying to create a convenient boot disk for the first time.

Using the QPAC1 utilities

TK2_EXT
Put a copy of the QPAC1 disk in disk drive one.
LRUN flp1_Boot

After booting from a copy of the Qpac1 disk, *press the keys Alt and c together.* You should get the Qpac1 calculator to "pop-up". *Press Alt-k.* The calendar should pop up. There should be a little arrow on screen (the Pointer), which you can move with a mouse or the cursor keys. When the pointer moves over an option that can be selected, the option is highlighted with (usually) box this is the "current item".

Move the pointer over the screen using the cursor keys. When the pointer becomes a padlock, the window below is "locked". To un-bury it, move the pointer to the locked window and Hit it by pressing Space or the Left mouse button. If the Calendar or the Calculator overlap each other's windows, you will need to Hit the overlapped program (either Calendar or Calculator) before you can use it. You can shift the windows. Moving the pointer arrow over the "double box" symbol (See *Figure four,* "move window") and Hitting it changes the pointer arrow to a "double box". Move the pointer to where you want the window and press



```
Hexcalc              ESC
                       0
AND NOT XOR  OR  L/H
 D   E   F  INT  BIN
 A   B   C   /   HEX
 7   8   9   *   DEC
 4   5   6   -   XCH
 1   2   3   +   RUB
 0   .  +/-  =   ZRO
```

```
Character Set                    Quit
 !"#$%&'()*+,-./0123456789:;<=>?
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
£abcdefghijklmnopqrstuvwxyz{|}~©
```

```
System Information

Total Memory    1920K
Free  Memory    807 K
Jobs Running    9
Open Channels   27
Net Station     21
Qdos Version    1.93
Ptr Interface   1.43
Wman Version    1.54
```

Space or Left mouse. Now you will be able to go from the Calculator to the Calendar and back again, without having to "un-bury" windows.

Be thorough

The Qpac1 manual covers the use of the Qpac1 utilities well. Do not be fooled by its brevity, go through each section, or you will miss something useful.

Config

The *Config* program is a "universal" configuration program. Most Pointer-driven programs use this program to CONFIGure themselves. The advantage is that only one program is required to CONFIG many programs. Programmers can incorporate "config" blocks in their programs instead of writing a config program for each program they write, saving time. Users only need to handle one config program, so they have fewer programs to learn and keep on disk.

Type EX flp1_CONFIG to run Config. The prompt "Give the name of the next file to be configured or press ESC to quit the program" will appear. If Config has a flashing cursor next to the prompt then it is waiting for keyboard entry of the filename - type in flp1_calculator and press Enter.

The later versions of config use the 'menus' extensions if they are present. If the prompt doesn't have a flashing cursor next to it, you should see a large window with the title "Select Filename". This is the "File Select" menu by Jochen Merz. See the Appendix *"Using File Select"* for full details.

The question "Configure Calculator version etc (Y, N or ESC)" appears. *Press Y.* Then the question "Prioritised or immediate calculation?" appears. Space will toggle between "Prioritised" and "Immediate", Enter sets the current choice.

Then we are asked where the configured calculator is going to be saved. The cursor is at the start of the filename (flp1_calculator) and this may be edited to save Calculator with a different name. If you attempt to save the configured calculator over the old one then approval is asked "File exists, OK to overwrite?". If you change your mind about the choices you made during the configuration then pressing Escape will abandon the configuration changes.

Rebooting

Common sense would suggest that once a file has been configured and saved, the changes made with Config will take effect immediately. If a resident program (a toolkit, system extension or job that has been HOT_RESed or HOT_CHPed) has been configured and saved then nothing will take effect until the QL is re-booted.

The only way to configure a resident program without the need for re-booting is to configure it, remove the old version from memory and load the new version into memory. This is not usually practical so most people tend to reset their QL instead.

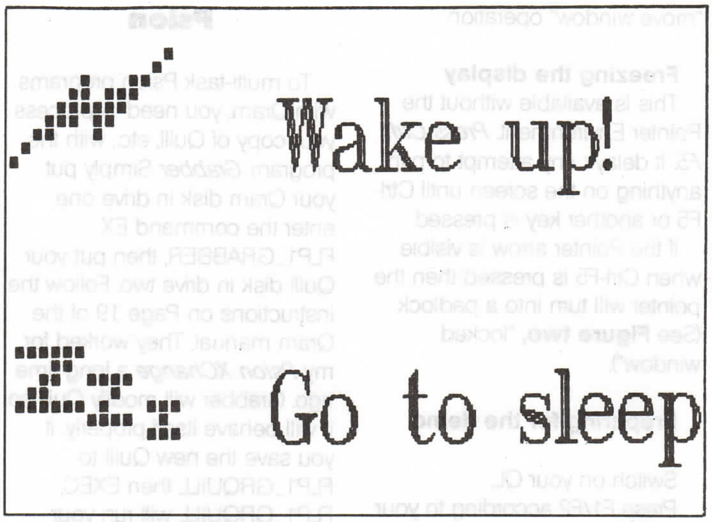
As usual, if you find a newer version of Config when you get a new Pointer program, use that instead of your old version. It may be a good idea to put the newest Config on your Boot disk. The very early versions of Config had bugs, and the later versions have extra features.

Config's biggest disadvantage for the user is that there is no "update" option - given an old configured file and its new, un-configured replacement file, Config won't configure the replacement file automatically with the information present in the old configured file.

The drawback for programmers is that Config is mainly for assembly language. Oliver Fink wrote a program to incorporate Config information in SuperBasic programs compiled by Qliberator (see Quanta disk SPECIALS_4). It is possible to write "action routines" to perform the configuration if the standard behaviour of Config is not enough. Unfortunately they have to be written in assembler. It would have made life so much easier if they could have been Qliberated SuperBasic as well.

Integrating Qpac1

Integrating Qpac1 into your existing set-up can be very easy



or very hard. If you already use the Pointer Environment and Hotkey Two then integrating Qpac1 is merely a matter of deciding how you want the Qpac1 jobs to be invoked. You need to decide whether they should be loaded from disk when needed, or made resident when Booting. Making the programs resident uses more memory but is convenient, especially if you don't have a hard disk.

The people who will find integrating Qpac1 hard are those who don't use the Pointer Environment and don't know much about SuperBasic. Some of their programs will clash with the Pointer Environment and they will be faced with an alarming case of culture-shock. If you use some old QL software with a "swapper" type of program, your best course of action is to get an experienced person to help you.

This section refers to certain "Hotkey" commands (prefixed by a HOT_) refer to the appendix "Hotkeys" for brief definitions.

A simple way of having your Hotkey programs set up is to HOT_RES them - whenever the Hotkey is pressed, a new job will be created. No additional memory will be used for the program code but memory will be used for the job's data-space and its screen image. Also, your computer can become cluttered if you have a lot of jobs in memory. This wastes a lot of the flexibility of the Hotkey Two.

A better way is to use the Hotkey Two to start a new job only if it is needed. Consider Qpac1's calculator - to pop-up a new Calculator job every time I pressed Alt-C, I would set it up with the command:

```
70 ERT HOT_RES('C',
'flp1_calculator')
```

Normally I would only ever need one calculator running at a time so I could set it up on a hotkey with:

```
70 ERT HOT_RES1('C',
'flp1_calculator')
```

When this statement is executed (usually) in the Boot program, it loads the calculator into memory and links it in as a "Thing". It also tells the Hotkey system that when Alt-C is pressed, it is to look for a calculator job - if it already exists then it is to display the calculator's windows (Pick it) instead of creating a new calculator job.

None of the above methods are good enough for me. I prefer to have my cake and eat it. So I define two hotkeys to deal with the same program.

```
70 ERT HOT_RES('C',
'flp1_calculator') :REMark
make resident & set key
75 ERT HOT_WAKE('c',
'calculator') :REMark set
another key for calculator
```

I press Alt-c to run a calculator and Alt-Shift-c to run a new copy of calculator. I manage to remember which key does what by using the Alt-Shift-key combination to perform various things (HOT_LOAD, HOT_RES) and the normal Alt-key combination to run the program "normally". The above sequence of commands will work if I want the program to be resident, but what if I wanted the program to be loaded from disk? I would use these commands:

```
70 ERT HOT_LOAD('C',
'flp1_calculator')
75 ERT HOT_LOAD1('c',
'calculator')
```

The calculator would be loaded from disk when needed - each time Alt-c or Alt-C is loaded, a new calculator would be loaded from disk. If it is likely that many calculator jobs will be running at any one time then a HOT_RESed calculator would be more economical in its use of memory. The HOT_LOAD routine is ideal for making hotkeys for programs that are only run occasionally.

People who are determined to build a Qpac1 boot should refer to the Appendix *"Building a QL boot program"* and *"Hotkeys"*.

DEMO 3: Pointer Software

This demo needs two disks from the Quanta library, 'SPECIALS_0' and 'UTILS_GEN4'. Put SPECIALS_0 in flp1_.

```
TK2_EXT
LRESPR flp1_PTR_GEN
LRESPR flp1_WMAN
```

This loads an old copy of the Pointer Environment. Some Pointer programs may refuse to run with it saying "Pointer Interface too old".

The disk UTILS_GEN4 has a selection of pointer programs. I'll deal with HEXC, VIEW, CHAR and INFO. **Figure five** is a screen dump of HEXC, VIEW, CHAR and INFO in action. Put UTILS_GEN4 in flp1_. Type:

```
EX flp1_HEXC_exe:EX
flp1_INFO_exe
```

You should get a display giving system information - the amount of free memory, network station number, various version numbers (be careful, my copy gets the version numbers of the Pointer Interface and the Window Manager mixed up, its easy to miss - Dick Copland spotted this one).

There should be a little arrow on screen (the Pointer). When the pointer moves over an option that can be selected, the option is highlighted with a white box - this is the "current item". Move the pointer over the screen using the cursor keys. When it is a padlock, the window below it "locked" as before.

The INFO program has buried the windows of HEXC, the hex/decimal/binary calculator (Hexcalc). Press Space and the pointer arrow will turn into a "double box" symbol to move a window. Move the pointer away to the far side of the screen and press Space or Left mouse button. This behaviour of INFO is not standard. The key-press for move-window is usually Ctrl-F4. You should be able to see the Hexcalc window and the *"System Information"* window **(Figure five).**

Having moved the windows so they don't overlap, move the Pointer to the Hexcalc window and press Ctrl-F4 or Hit the "double box" option for "move window". Move Hexcalc so its windows are only just overlapping System Information's windows. Move the Pointer over System Information's windows - it should change to a padlock. Press Space or Left button and Hexcalc's windows will be "unburied" covering System Information's windows. If System Information or the Hex Calculator jobs overlap each other's windows, you will need to Hit the overlapped program before you can use it.

Press Ctrl-C until you reach SuperBasic and type in the command:

```
EX flp1_VIEW
```

So far, in all these demos, the pointer is normally an arrow. The View program provides a custom pointer, a "hand" symbol. This is shown in Figure five. When the Pointer is over the View window, it turns into a hand. Press Ctrl-C to get another program on screen. Move the pointer over its windows. You should find that the pointer is again an arrow in the other program. View is a simple program - simply give it a file name and it will display the file in its windows, waiting for you to ask for more. As View does not read the whole file into memory (it opens a channel to the file and reads information when it is required), you should not remove a disk that has a file you are VIEWing. Any attempts to read a bit more of the file by the View program will fail. Any attempt by other programs to use other disks in the same drive will fail with the error message "Files still open" and "not found". Wrap is the only obscure option in View - see the Appendix *"Viewing files on your QL"* for Wrap On and Wrap Off.

Play around

Play around with the pointer programs. A Pointer Environment manual isn't provided with these Quanta disks - experiment and use this article for information. You won't be able to define hotkeys, as 'hot_rext' isn't in the Quanta library, but the other Pointer programs will be available.

THE APPENDICES

Viewing files on your QL

If you have Qram then pop-up the Qram menu and select/Hit Files. Then select/Hit any files you want to view and Do View (press V). Pressing "N" for "Next" will move to the next file if you selected more than one file to view.

If you have Qpac2 then pop-up the Files menu. Then select/Hit any files you want to view, press F4 and Do on the name of a file you want to view. Pressing '<' or '>' will move to "previous" and "next" files if you selected more than one file to view.

If you have *Toolkit Two* then the View command will show files in a SuperBasic window but long lines will be not be shown in full - only the part that fits on a single row will be displayed, the extra characters ignored by View. This is equivalent to having Wrap Off when viewing files in Qram or menu_rext's VIEW_FILE. This explains why VIEWing a Quill document does not show the full file on screen.

Toolkit Two can View files, with the equivalent of Wrap On, if you use the SPL command. The SPL command sets up a job to SPooL data from one place to another. The command *SPL "filename",SCR* will copy files to the screen. Because of the "non-destructive" windows of the Pointer Interface, the window will disappear when the whole file has been displayed. This is a nuisance if you don't press Ctrl-F5 quickly enough to freeze the screen to stop this from happening. An alternative - *SPL "filename",#1* - will send the file to SuperBasic channel 1. You can view many files easily, if you use a combination of the SPL_USE and WCOPY commands.

On a TV use:

```
SPL_USE con_x200
WCOPY flp1_
```

On a Monitor use:

```
SPL_USE con_512x206a0x0
WCOPY flp1_
```

When more than one file is displayed, one after the other, the display can be messy. But the use of WCOPY allows wild-cards to be used and allows simple browsing of a disk or sub-directory.

Note on SPL_USE: The effects of the SPL_USE command are permanent until a Reset or a SPL_USE/DEST_USE command is executed. Putting a SPL_USE command in your Boot file could save you time - you could view files with a simple WCOPY command.

Copying files on your QL

This is not the definitive guide to copying files but it will "get you started".

If you have two disk drives then it is very easy, just put the disk you want to copy from in drive two and the disk you want to copy to in drive one and type:

```
WCOPY flp2_ TO flp1_
```

This Toolkit Two command will give you a filename and ask ".Y/N/A/Q?". For example:

```
flp2_Boot TO
flp1_Boot.Y/N/A/Q?
```

The action the QL takes depends on your next key-press:

Y or y - to copy the file, ask me about the following files

A or A - copy this file and ALL the files after it

N or n - don't copy the file, ask me about the following files

Q or q - don't copy the file, don't ask me about any more files

If you don't have a double disk drive, and you have a ramdisk set up, follow these instructions:

If you type *DIR RAM1_* and you get the following two lines:

```
RAM1
200/400 sectors
```

Then you have a ramdisk (ramdisks are standard with Trump Cards and Gold Cards and don't have to be formatted). The first number refers to the amount of free space left on the ram disk, in units of "half a KB" (the 200 shows that 100KB is free, this is approximately the amount of space on a microdrive cartridge). The second number refers to the total amount of space on the ramdisk (simply divide by two to get the capacity in KB - in this case 200KB).

If you don't get the line 'RAM1' followed by a sector count then your ramdisk may need to be FORMATted before it is used. This was necessary with the ramdisk on the Sandy SuperQboard. Enter the command FORMAT RAM1_200. If you get "not found" or "out of memory" then please contact an experienced friend to help you further.

With a ramdisk ready to go, put the disk you want to copy FROM and type WCOPY flp1_, ram1_. Copy the files you want into the ramdisk. Then put the disk you want to copy to into the disk drive and type *WCOPY ram1_,flp1_*. Press A to put a copy of every file present on the ramdisk onto the floppy disk. The files are still on the ramdisk; to get rid of them type *WDEL ram1_*. It is a pity there is no WMOVE command.

Creating a Boot disk for your QL.

These instructions will help you set up a Boot disk for the first time and teach simple fault-finding.

If you are using the Pointer Environment it is a good idea to have a single disk to boot from (but keep backup copies of it). This is because the Pointer Environment (ptr_gen, wman) and related files (hot_rext, menu_rext etc) have been improving over time. If you get a new Pointer program that has a newer version of ptr_gen in it, put the newest version on your Boot disk. It is possible that your new Pointer program needs facilities that are only present in the newer files.

Finding out the version of most Pointer Environment files is easy - all you have to do is View them (see "VIEWing files" appendix).

The usual way to create a Boot file is to type "lines" into SuperBasic and then SAVE them. A "line" in SuperBasic is different from a "command" because it has a line number in front of it. When a "line" is typed into SuperBasic, it stores that line in its memory. Type NEW to clear any old "lines" from SuperBasic's memory. Enter these lines into SuperBasic:

```
5 REMark QPAC1 boot file
10 TK2_EXT
20 LRESPR FLP1_ptr_gen
30 LRESPR FLP1_wman
40 LRESPR FLP1_hot_rext
```

Line 5 is a comment line, used to keep notes on the Boot program. Line 10 activates toolkit two. Lines 20 to 40 load the Pointer Environment (ptr_gen), the Window Manager (wman) and the Hotkey System Two (hot_rext).

To show all the "lines" that SuperBasic has in memory, type the command LIST. A group of SuperBasic "lines" is called a SuperBasic program and they may be stored on disk by the commands LOAD and SAVE. Try LISTing the lines you have typed in so far. If you have made a mistake then type the command EDIT followed by the number of the line you made the mistake in.

The "lines" entered so far are instructions to load the Pointer Environment. As we have Qpac1, we would like to load those utilities as well. If we were using Qram, we would use the programs Boot_MAKE and HOT_MAKE. Being lucky Qpac1 users we get more flexibility and the thrills of typing in our own Boot programs. To complete the boot program, we need to add the commands for using the Qpac1 utilities.

```
50 REMark Load QPAC1 utilities
60 ERT HOT_RES('A', 'flp1_alarm')
65 ERT HOT_WAKE('a', 'alarm')
70 ERT HOT_RES('C', 'flp1_calculator')
75 ERT HOT_WAKE('c', 'calculator')
80 ERT HOT_RES('K', 'flp1_calendar')
85 ERT HOT_WAKE('k', 'flp1_calendar')
90 ERT HOT_RES('D', 'flp1_clock')
95 ERT HOT_WAKE('d', 'clock')
100 ERT HOT_RES('M', 'flp1_sysmon')
105 ERT HOT_WAKE('m', 'sysmon')
110 ERT HOT_RES('W', 'flp1_typer')
115 ERT HOT_WAKE('w', 'typer')
120 ERT HOT_PICK('b',")
130 ERT HOT_LOAD('s', 'flp1_config')
140 HOT_GO
150 HOT_Do 'm'
160 PAUSE 50
170 HOT_Do 'b'
180 HOT_LIST
```

When you have typed all these "lines" in, put a blank formatted disk in drive one and type SAVE FLP1_Boot. The SAVE and LOAD commands must always be given a file name to tell SuperBasic which file it is to save/load. The Boot file we have saved may be read into SuperBasic again with the command LOAD FLP1_Boot. When a file is LOADed, the previous SuperBasic program is removed from memory - it is NEWed.

Note - If your QL says "FLP1_Boot exists, OK to overwrite..Y or N?" *press N* and use a disk that really is blank.

When your disk drive light switches off, your QL will have finished saving the Boot file. Now we need to copy the other files onto your disk. Following the instructions "Copying files" in the appendix, copy the files ptr_gen, wman, hot_rext, alarm, calculator, calendar, clock, sysmon, typer and config to the disk you saved our "boot" file to. When the disk drives have stopped whirring, press Reset and F1/F2 when the prompt appears.

Simple fault finding in Boot programs

Some resident programs insist on being loaded before others. The manual with *Lightning* (most editions) insists that it should be loaded before Toolkit Two is activated. By the time my QL gets around to looking for Lightning, *Toolkit Two* is already activated. Nothing happens so I seem to be getting away with it.

When loading the Pointer Environment the Pointer Interface "ptr_gen" has to be loaded before "wman". You can load "hot_rext" whenever you want as far as ptr_gen and wman are concerned.

When your QL stops in the middle of Booting and prints:

```
5 REMark QPAC1 boot file
10 TK2_EXT
20 LRESPR FLP1_ptr_gen
30 LRESPR FLP1_wman
40 LRESPR FLP1_hot_rext
50 REMark Load QPAC1 utilities
60 ERT HOT_RES('A', 'flp1_alarm')
65 ERT HOT_WAKE('a', 'alarm')
70 programmer_LIST
At line 70;1
bad name
```

An error has occurred. The line number will vary and so will the message. (Note - Minerva error messages give a command number as well as a line number, a non-Minerva QL would have given the message "At line 70" instead of "At line 70;1"). The first thing to do is to see the line that the QL has stopped at. The command LIST 70 will list the offending line, in our case:

```
70 HIT_LIST
```

This is a spelling mistake - when writing the line, I was thinking of people who don't pay their invoices and so quite understandably typed "HIT_LIST" instead of "HOT_LIST". Most "bad name" error messages are caused by bad spelling of command names.

Attempting to use toolkit commands before they have been loaded will also give a "bad name" error:

```
20 LRESPR flp1_PTR_GEN
30 LRESPR flp1_WMAN
35 TK2_EXT
38 HOT_LIST
40 LRESPR flp1_HOT_REXT
```

The above Boot program attempts to use a Toolkit Two command (LRESPR) before Toolkit Two is active (Line 35 should be before line 20). It also tries a Hotkey command before Hotkey Two is loaded (Line 38 should be after Line 40).

```
10 TK2_EXT
20 LRESPR flp1_PTR_GEN
30 LRESPR flp1_WMAN
```

```
40 LRESPR flp1_HOT_REXT
50 HOT_LIST
```

Another common error is "not found" - this usually occurs when a file to be loaded isn't on the disk. It could be that you have forgotten to put the file on the disk or you have given your QL the wrong filename.

```
10 TK2_EXT
20 LRESPR flp1_POINTER_GEN
30 LRESPR
flp1_WINDoW_MANAGER
40 LRESPR flp1_HOTKEY_TWO
```

The above program would be understood by a person but a computer likes a bit more precision. A QL would just give up with an error message. The QL-friendly version of the above is:

```
10 TK2_EXT
20 LRESPR flp1_PTR_GEN
30 LRESPR flp1_WMAN
40 LRESPR flp1_HOT_REXT
```

When you are sorting out a Boot program, when an error occurs, you may want to tell the QL to carry on going, ignoring the current statement.

```
10 TK2_EXT
20 LRESPR flp1_PTR_GEN
30 LRESPR flp1_WMAN
38 HOT_LIST
40 LRESPR flp1_HOT_REXT
```

When the above Boot runs it will give an "At line 38;1 bad name" error message. LIST displays the whole Boot and we see that line 38 is a Hotkey command being used before the Hotkey Two is loaded. To tell the QL to skip this command and carry on running the Boot we type CONTINUE. The QL finishes the Boot.

To edit the Boot type ED. Move the cursor down to line 38 and delete the 38 by pressing Ctrl-Right arrow, press 5 then 0 to replace the 38 with 50 and press Down arrow. The screen should show:

```
10 TK2_EXT
20 LRESPR flp1_PTR_GEN
30 LRESPR flp1_WMAN
38 HOT_LIST
40 LRESPR flp1_HOT_REXT
50 HOT_LIST
```

Delete the unwanted line, 38, by moving the cursor over line 38 and pressing Ctrl-Alt-Left arrow. Press ESCape to leave ED and save the new Boot program with the command SAVE flp1_Boot. Your QL should ask "flp1_boot already exists, OK to overwrite..Y or N?". *Press Y* and wait for the disk drive light to go out before removing the disk or resetting your QL. Full instructions for the ED command are in the Toolkit Two manual (this may be found in the user guide of your expansion card, such as the SuperQboard manual, Trump Card manual, etc.).

Sometimes you will want to try a command in a boot program again. Consider the following 'Boot' file being run on a disk with the files PTR_GEN and HOT_REXT on it.

```
10 TK2_EXT
20 LRESPR flp1_PTR_GEN
30 LRESPR flp1_WMAN
40 LRESPR flp1_HOT_REXT
50 HOT_LIST
```

The Boot will stop with the error message "At line 30;1 Not found". I would type DIR flp1_ to get the list of files present on the disk. Hopefully I would realise that the file 'WMAN' isn't on the disk and put a disk that has 'WMAN' on it into drive two. and use the command COPY flp2_WMAN to flp1_WMAN. Then, with the problem fixed, I could type RETRY to tell the QL to try the line it stopped at (line 30) and carry on. As the file 'WMAN' has been copied, the boot will finish with no more errors.

File lengths and RESPR

When the QL first came out, the only way of loading toolkits was to allocate some resident procedure space (RESPR), load the toolkit into that space (LBYTES) and initialise it (CALL). This was inconvenient because when a new version of a toolkit was produced, all the boot programs had to be updated to accommodate the new size of the toolkit.

To load PTR_GEN 1.54 (file size 14534 bytes) using the old method:

```
address=RESPR(14534)
LBYTES flp1_PTR_GEN,
address, CALL address
```

The Toolkit Two equivalent is:

```
LRESPR flp1_PTR_GEN
```

It is only safe to replace the old method of RESPR-LBYTES-CALL with LRESPR if the code actually CALLs "address". If, in the example above, I had to use CALL address+54, the LRESPR command would not be suitable. Fortunately most programmers make sure their toolkits can be LRESPRed.

Coping with unexpected "freezes"

This can be caused by some toolkits clashing or by replacing a CALL-LBYTES-RESPR sequence by an inappropriate LRESPR.

Start tracking down the problem until you know exactly where in the Boot your QL is stopping/crashing. One way to do this is to insert lines into the program which display a message. For example, if 'WMAN' had become corrupted (without my knowledge) and this Boot program was crashing my QL for no apparent reason:

```
10 TK2_EXT
20 LRESPR flp1_PTR_GEN
30 LRESPR flp1_WMAN
40 LRESPR flp1_HOT_REXT
50 HOT_LIST
```

I would start putting PRINT lines in to indicate the QL's progress.

```
10 TK2_EXT
20 LRESPR flp1_PTR_GEN
25 PRINT 'ptr_gen loaded'
30 LRESPR flp1_WMAN : PRINT
'wman loaded'
40 LRESPR flp1_HOT_REXT :
PRINT 'hot_rext loaded'
50 HOT_LIST
```

The above listing shows line 25 has been added and that lines 30 and 40 have had extra commands added to the end of them. The commands are separated by a colon : and the only reason that they are on the same line is to give an example of this. Putting the PRINT commands on a new line or at the end of other lines does not affect the behaviour of this Boot program.

Hotkeys

The Hotkey System Two uses the Thing System to keep track of its resident programs. The Thing system is an extension to Qdos found in 'hot_rext'. It acts as a caretaker, keeping track of Things - where they are, what they are, who is using them, etc. QL users don't need to know much about them. QL programmers should use them more often.

Hotkeys to make resident jobs

When the Hotkey is defined, the program referred to will be loaded into memory and linked into the system as an executable thing.

The HOT_RES, HOT_RES1 functions will attempt to load the program into the resident procedure area (RESPR space) - if this is not possible (because some jobs are already running), it automatically loads the programs into the common heap. The HOT_CHP, HOT_CHP1 functions will load the program into the common heap.

If you weren't happy with the "Hotkey System 2" manual when you first looked at it, go back to it, look at the example boot programs (page 11) and refer to the table above. It may seem a bit clearer.

If that fails, have a look at the listing that accompanied my Qpac1 article, *(QL World,* March 1991). The listing has some Hotkey definitions and the article gave some explanations.

FILE SELECT - A SIMPLE GUIDE

The File Select menu, written by Jochen Merz, is part of "menus" extension from the file "menu_rext".

Installing menu_rext

When modifying your boot to load menu_rext, you will need to make sure hot_rext (the Thing & Hotkey system) is loaded before menu_rext. This is because menu_rext links itself into Qdos using the Thing system.

Have a look in your Boot file. If there is a command that loads hot_rext (usually LRESPR flp1_hot_rext) then place the command LRESPR flp1_menu_rext after that. If hot_rext isn't being loaded by your Boot file, change it so it does - menu_rext can't work without it.

A breakdown of the commands with Qpac1/Qpac2 (EE) page numbers.

Page		Function	Action performed when hotkey is pressed
Hotkeys to type in text			
6	EE32	HOT_KEY	Type in text
7	EE33	HOT_CMD	PICK SuperBasic & type in text
Hotkeys to load in jobs			
5	EE35	HOT_LOAD	Load & execute a job otherwise Load & execute it
Hotkeys for executable Things			
5	EE38	HOT_THING	Execute a thing (creates a job) otherwise execute a thing (creates a job)
Hotkeys to make resident jobs			
2	EE34	HOT_RES	Execute a job from memory otherwise execute the thing (creates job)

Using File_Select

Move the pointer over the window - notice how items are highlighted as the pointer passes over them. This highlighting shows the user different options that they can trigger.

Enquire about Hotkeys

Page		Name	Type	Action
7	EE42	HOT_NAME$	Function	Return text linked with a key
7	EE42	HOT_TYPE	Function	Return a key's action type
7	EE42	HOT_LIST	Procedure	List hotkey definitions
Hotkey system commands				
6	EE39	HOT_PICK	Function	Ctrl-C a named job to the top of the window pile
9	EE44	HOT_DO	Procedure	Execute a hotkey from a program
7	EE44	HOT_STUFF	Procedure	Place text in the stuffer buffer
7	EE41	HOT_STOP	Procedure	Stop the 'HOTKEY' job
7	EE41	HOT_GO	Procedure	Start the 'HOTKEY' job

The file select menu has three main areas - "file name", "device name" and "file list".

The "file name" area, just below the title "Select File Name", will allow you to enter the file name in full. There are two other options - Current and Previous. This refers to the Hotkey buffer (Alt-Space) which is set by some programs, usually to the name of a file being worked on. This "stuffing" of file names saves the user having to remember and type file names.

The "device/directory name" area, at the bottom left of the "file area", allows you to change the drive whose files are listed in the "file list". The current device can be changed to (MDV, FLP etc) and FILE_SELECT is intelligent. It searches the system tables for device names so even if you added a laser disk to your QL (LAS1_? OPT1_? CD1_?), FILE_SELECT would be able to handle it. The device/directory name may be Hit to edit it via the keyboard or it may be Done to pull down a Select Directory menu. Some of the facilities it offers are duplicated by FILE_SELECT - its unique options are "Data default" and a "common directory list". The "Data default" option reads the current Toolkit Two data default so that it can be edited or approved by "OK". The "common directory list" may be tailored to the individual user's needs by using config on menu_rext.

The "file list" area, at the bottom right of the "file area", is a menu of file names. Doing a file name quits the FILE_SELECT menu and marks that file name as "chosen". Hitting that filename puts it in the file name buffer where it can be edited or chosen by Hitting/Doing "OK". The "file list" has some controls to make life easier.

The "EXT" control will allow the user to set a file extension so that only files ending with a particular file extension will be listed. Hitting EXT (or pressing E) will allow keyboard entry of a new file extension. Doing EXT will pull down a menu of common file extensions. This menu of file extensions may be tailored to the individual user's needs by using config on menu_rext.

The "View" control, a recent improvement to File Select, is normally switched off. When View is switched on (by Hitting it) then the handling of the file menu is subtly changed. Doing a file in the file menu will still chose that particular file. But Hitting a file will now ask your QL to display the file. The View screen has a few simple commands - ESCape (Quit viewing), Wake (go to start) and Wrap. Wrap is usually switched off. With Wrap switched off, only the first part of long text lines will be displayed. With WRAP switched on, long lines will wrap around onto the next display line. It is best to view Quill documents, program files, any binary data with Wrap switched on.

The Directory Up and Tree controls deal with "level-2 device drivers" where "hard sub-directories" are being used - mainly of interest to Atari QL or Gold Card QL users. The Tree control, when switched on, will look for directories below the current directory (sub-directories). Then the names of files found in these sub-directories will be placed in the file menu. Switching Tree off will cause all the files from sub-directories to be removed from the file list. The Directory Up control is only of use in a sub-directory. It tells the file menu to get its file names from the "parent" directory of the current directory.

Sources of QL Expansion/related items

Miracle Systems Ltd.
25 Broughton Way
Osbaldwick
York YO1 3BG
0904 423986

Gold Card system expansion £225: all Trump Card facilities or better, 1920KB memory, QL 5x faster, plus DEV device, Hard sub-directories

Dual ED disk drives £175. Can handle 720KB DD, 1.44MB HD and 3200KB ED 3.5in disks. Intended mainly for Gold Card

Qubbesoft PD
38, Brunwin Road
Braintree
Essex CM7 5BU
0376 347852

Trump Card, £95.00 plus p&p. 768KB memory for a total of 896KB, Toolkit Two, screen dump, printer buffer, dynamic ram disk, disk interface.

Expanderam 512K, £45 plus p&p. Expands QL's memory to 512+128 giving 640KB. The drawback is that another interface is needed if disk drives are to be used.

Dual 720KB 3.5-in disk drives, £100 plus p&p.

See their regular adverts in QL World for more details.

EEC (W N Richardson & Co.)
18-21 Misbourne House
Chiltern Hill
Chalfont St. Peter
SL9 9UE
0753 888866

Various types of *Universal disk drives* (suitable for use on many computers). £25-£130.

QL serial mouse (three buttons) £45. In this almost complete package you get some driver software (a file called E_SERMOUSE), a mouse, a cable, instructions and a mouse mat. The only missing component is a copy of PTR_GEN. PTR_GEN is included with all commercial Pointer programs these days.

Mouse suitable for QIMI £10

Quanta
c/o Bill Newell
213 Manor Rd.
Benfleet
Essex SS7 4JD

QIMI mouse interface £27
Motorola 68000 reference £6
Definitive Guide to SuperBasic by Jan Jones £10

Membership to Quanta is by annual subscription.

Various second hand QL systems - Caveat emptor and vendor!

Dilwyn Jones Computing
41 Bro Emrys
Tal-y-Bont
Bangor
Gwynedd LL57 3YT
0248 354023

Main UK dealer for *Qjump software:*

Qpac1 £19.95, *Qpac2* £29.50

Other pointer programs sold include *Disa Disassembler* (intelligent machine code tool) £29.00 and *Data Design* (database) £50

See adverts in QL World.

Pointer Products
68 King Down Rd
Blandford
Dorset DT11 8BN
Sells many different Pointer Programs
0258 455117

Jochen Merz Software
Im Stillen Winkel 12
D-4100 Duisberg 11
Germany
0203 591706

A bulletin board "Qdos-Mailbox" service is planned to allow registered users to update easily. Main dealer for many German programs - English manuals provided, and English versions of German programs. Can accept UK cheques. Also sells some QL hardware and QL emulators for the Atari ST. See adverts in QL World. Recent releases include:

EasyPtr 3 - a Pointer program development tool. The new version provides support for C programmers.

QSpread spread sheet. This has been revised and has a new manual.

QVME QL-emulator for the Atari STE.

THE POINTER ENVIRONMENT

CLUB

FRANCE

QL Contact France. Contact: Jean-Louis Dianoux, 38-40 Rue Stephenson, 75018 Paris, France.

BELGIUM

Club Sinclair BruQsL (Belgium) Contact: Jaques Tasset, Aarlenstraat 104, 1040 Brussels, Belgium

SWEDEN

International QL Conference bulletin board system (Swedish and English). Contact: Michael Cronsten, System Operator, Jamten-TCL, S Soere 1073, 83030 Lit, Sweden.

USA

New England Sinclair QL User Group (USA) Membership Secretary: Sherm Waterman, PO Box 8763, Boston, MA 02114 8763, USA. Magazine: *NESQLUG News*. Editor: Peter Hale, 195 Central Ave., Chelsea, MA 02150, USA.

NORWAY

Norwegian All Sinclair Association (NASA) Contact: P Monstad, NASA, N-5580 Oelen, Norway. Magazine: *Sinclair Magazine*.

ITALY

Qitaly Club Chairman: Roberto Orlandi, Via Brescia 26, 25039 Travagliato (BS), Italy. Tel. (local) +39 30 6863311. Magazine: *Qitaly Magazine*. Editor: Dr Eros Forenzi, Via Valeriana 44, 23010 Berbenno (SO), Italy. Tel. (local) +39 342 492323.

TURKEY

QL Qlub (Turkey). Contact: Bulent Artuz, Prof. Sitesi B/1 D/5, Etiler 80600, Istanbul, Turkey.

AUSTRIA

DER Computer Club. Contact: Peter Postl, Stiegerg 5, 1150 Wien, Austria

ENGLAND

Quanta (UK) Membership Secretary, Bill Newell, 213 Manor Road, Benfleet, Essex SS7 4JD. Magazine: *Quanta*. Editor: Bill Fuggle, 20 Widnes Avenue, Selly Oak, Birmingham B29 6QE.

Bristol sub-group: Roy Brereton, 94 Teignmouth Rd, Clevedon, Avon BS21 6DR.

Essex sub-group: Dave Walker, 22 Kempton's Mead, Potters Bar, Herts EM6 3HZ.

London sub-group: Jeremy Davis, 6 Elmcroft Crescent, Harrow, Middlesex HA2 6HN.

Northern Ireland: Billy Turkington, Fairyhill, Rostrevor, Newry, Co., Down BT34 3BB.

Mid-Southern sub-group: Geoff Fish, 44 Billing Avenue, Wokingham, Berks RG11 4JE.

QL User Group (West Midlands): Mike Bedford-White, 16 Westfield Road, Acocks Green, Birmingham B27 7TL.

QL MUG: 3 Barnard Road, Birkenhead, L43 1TT.

SCOTLAND

Scottish QL Users Group Contact: Alan Pemberton, 65 Lingerwood Rd., Newtongrange, Midlothian EH22 4QQ. Newsletter.

HOLLAND

Sin_QL_Air (Netherlands) Membership Secretary: Bob Visser, Snelrewaard 6, 2904 SN Capelle, a/d Ijssel, Netherlands. Magazine: *Quasar*. Editor: C H M Biemans, Elzenstraat 5, 5461 CL Veghel, Netherlands.

GERMANY

Sinclair QL User Club eV (Germany) Foreign Contact: Franz Herrmann, Talstrasse 21, d-W5460 Ochenfels, West Germany. Magazine: *Quasar*.

SPAIN

Qliper Editor: Marcos Cruz, Acacias 44, (Monteclaro), E-28023 Pozuelo de Alarcon, Spain. Magazine: *Qliper*.

SINCLAIR QL WORLD

££££

SUBSCRIBE TO SINCLAIR QL WORLD AT LAST YEARS PRICES AND SAVE ££££££'s

UK
£23.40
(12 MONTHS)

EUROPE
£32.90
(12MONTHS)

REST OF WORLD
£40.90
(12 MONTHS)

PLEASE SEND YOUR
SUBSCRIPTION FORM TO
SINCLAIR QL WORLD,
THE BLUE BARN, TEW LANE,
WOOTTON, WOODSTOCK,
OXON. OX71HA.
TEL:0993 811181
FAX:0993 811481

I WISH TO SUBSCRIBE TO SINCLAIR QL WORLD
WE HAVE HAD TO RAISE THE COVER PRICE OF QL WORLD BUT WILL BE HOLDING LAST YEARS SUBSCRIPTION PRICE FOR THE IMMEDIATE FUTURE ONLY.

Existing subsrcibers will not be affected.

PLEASE DEBIT MY CREDIT CARD ACCESS/VISA

CARD NUMBER...

EXPIRY DATE

PLEASE SEND ME A 12/24 MONTH SUBSCRIPTION TO SINCLAIR QL WORLD.

NAME ....

ADDRESS ....

....

TELEPHONE NUMBER...

SIGNED...

DATE ....

STARTING WITH ISSUE...

ALL OVERSEAS READERS WE ARE NO LONGER ABLE TO SUPPLY SINCLAIR QL WORLD TO OVERSEAS NEWSAGENTS. TO ENSURE YOU RECEIVE THE NEXT EDITION (MARCH 1993) SUBSCRIBE NOW!

WARNING

I wish to subscribe to SINCLAIR QL WORLD

SINCLAIR QL WORLD

PLEASE DEBIT MY CREDIT CARD ACCESS/VISA (not American Express)

CARD NUMBER..

EXPIRY DATE..

PLEASE SEND ME A 12 MONTH SUBSCRIPTION TO SINCLAIR QL WORLD.

NAME..

ADDRESS..

....POST CODE..........................

TELEPHONE NUMBER..

SIGNED..

DATE..

Please send your subscription form to: Arcwind Ltd.
The Blue Barn,
Wootton,
Woodstock, Oxon.
OX7 1HA

EUROPE
£32.90 (12MONTHS)
REST OF WORLD
£40.90 (12 MONTHS)

DIGITAL PRECISION

THE FANTASTIC SPECIAL EDITION OF LIGHTNING

The Program for Everyone

If you are using a QL in **any** shape or form or with **any** accessory (thus including the new QXL 68040, Gold Card, Trump Card, ST/QL, Thor, PC CONQUEROR, Minerva, TURBO, and even the humble unexpanded microdrive-only QL), you **really should** be using **LIGHTNING SPECIAL EDITION**. If not, you are very severely and **unnecessarily** (our program is quite inexpensive) slugging your system's performance. The superb **LIGHTNING SPECIAL EDITION** will both automatically and very significantly accelerate almost every aspect of QL operation - whatever it is you use the QL for. ***"More than 10x is achievable and 2x-4x is typical"*** *(quote from page 24 of review in April '90 QL World).* The speedup ratio is independent of the system. However fast or slow your hardware, **LIGHTNING SPECIAL EDITION** will accelerate it much further. All recent versions of our software are carefully optimised for 16/32 bit processors, without compromising 8 bit working. The program has not got any adverse side effects at all, and it fixes QL anomalies. Installing it is a fast, once-only operation that takes two or three minutes and which assumes & requires absolutely no knowledge of programming or of anything even remotely technical about the QL: you are simply asked whether you wish to speed up text, maths and graphics individually, or **everything**. Unless you have a very good reason, opt for everything! Then **LIGHTNING SPECIAL EDITION** copies itself onto your boot-up disks, instantly modifying their **BOOT** files. Now every time you start up, full throughput acceleration is automatically invoked and everything goes much smoother and faster. In case you think that this is too good to be true, we quote verbatim the concluding para of the Sinclair QL World review: ***"I could not fault LIGHTNING SPECIAL EDITION on anything. It is a clear winner and a best buy at £49.95"***. The program includes a bundle of accessories (change fonts etc. in Quill etc., smooth scrolling and much more) and tweaks (vary maths and/or graphics precision, a null device and much more). Stop reading the manual where we tell you to - at around page four - if simple use is all that you want. The program also includes 84 excellent small fonts for use with both **PERFECTION SPECIAL EDITION** and **PROFESSIONAL PUBLISHER**: a real bonus!

LIGHTNING SPECIAL EDITION includes both a ROM (for plugging in at the back of your QL - no screwdriver needed) and a disk. As some QL hardware (QXL; Gold Card for speed reasons) is not ROM-friendly, or you might have something already plugged in (ICE, TK2 if not already on your disk interface), you can get a version of the program minus the ROM for just **£39.95**: this is the **GOLD CARD VERSION**. If you have two QLs, say one of them a QXL / Gold Card and one "ordinary", you should go for the full **LIGHTNING SPECIAL EDITION**, as you can use the ROM on the second machine. Extra ROMs cost £10 if ordered at the same time as the program, else £15.

Q1) What programs benefit from **LIGHTNING SPECIAL EDITION** (LNGSE)? **A)** <u>All</u>, including emulators. Perhaps PERFECTION SE benefits most. **Q2)** Why didn't you build it into all your programs? **A)** It would be very inefficient to do so because of multitasking. Also, LNGSE benefits all programs (even Quill etc.), not just our ones. **Q3)** Does the QL "know" it is running LNGSE? **A)** No. And it isn't "running" LNGSE either. In its first and only second of life LNGSE pages out, using a door deliberately left open by the QL's forward-thinking designer, large chunks of QDOS (AH, JM, JS, MG and all Minerva operating system variants) and replaces them with our fine-tuned supercode. **Q4)** Is it a compiler? **A)** No - TURBO is. LNGSE greatly improves the performance of TURBO'd programs too! **Q5)** Why is LNGSE so cheap? (happy users ask this) **A)** The truth is, we know that once you have experienced **LIGHTNING SPECIAL EDITION**, you won't abandon your QL. As **THE** QL software publisher, that is rather good news for us. **Q6)** Give me one more reason for buying it. **A)** Look at our **SPECIAL DEALS**, and <u>think</u>. Even before any seasonal discount, LNGSE Gold Card would add a mere £30 to the price of PERFECTION PLUS SPECIAL EDITION, for example. **SPECIAL DEALS** allow you to get programs for free, even to get us to pay you to buy them...

DJC

Dilwyn Jones Computing

41 Bro Emrys, Tal-y-Bont,
Bangor, Gwynedd LL57 3YT U.K.
Tel: Bangor (0248) 354023

QL SOFTWARE

NB PLEASE NOTE THE MEANING OF CODES IN SQUARE BRACKETS BELOW:
[R] RAMDISK REQUIRED
[F] AVAILABLE ON FLOPPY DISK
[M] AVAILABLE ON MICRODRIVE
[128K/512K] MINIMUM MEMORY REQUIRED.
[PC] FOR IBM PC AND COMPATIBLES.
[TK2] TOOLKIT 2 REQUIRED.

CATALOGUE

THE CATALOGUE CONTAINS MORE DETAILS OF THE PROGRAMS. CALL FOR A COPY, OR ASK FOR A COPY WITH YOUR ORDER.

DEMONSTRATION VERSIONS

Demo versions are available for some of our programs. They cost £2.00 each and this cost is refundable against the full price of the full version, see lists below.

FLOPPY DISKS & LABELS

3.5" DSDD UNBRANDED DISKS	£0.40
3.5" DSHD FLOPPY DISKS	£0.70
MICRODRIVE CARTRIDGES	£2.50
100 DISK LABELS	£2.00
100 DISK LABELS ON PRINTER ROLL	£2.50
100 ADDRESS LABELS ON ROLL	£2.00
100 MICRODRIVE LABELS ON ROLL	£2.00

Please add £0.50 postage per roll of labels (max postage £2.50 if only ordering labels. Add £2.50 per order for floppy disks. No extra postage on microdrive cartridges.

FILE TRANSFER

DISCOVER	£20.00
[F 256K] Copy files QDOS to PC disks and vice versa.	
MULTI DISCOVER	£30.00
[F 256k Enhanced versions, extra formats.	
TEXTIDY	£15.00
[F M 256k] Strip Quill files etc down to plain text	
OPD INTERCHANGE	£15.00
ICL/OPD file transfer utility. Unsuitable Gold Card.	
QL-PC FILESERVER	£24.50
[F 128k] QL/PC software for serial cable link, access PC's drives, printer, transfer files, etc	

LEISURE

SOLITAIRE	£15.00
[F M 128k] A very addictive patience card game.	
CRICKET SECRETARY	£12.00
[F M 128k] For cricket buffs everywhere!	
QUESTION MASTER	£10.00
[F M 128k] For revision or entertainment.	
QUIZ SETS each	£5.00
FLEET TACTICAL COMMAND II QL	£39.95
[F 512k] Naval strategy game, play on 2 networked QLs.	
FLEET TACTICAL COMMAND II PC	£49.95
[PC] Licenced PC version of this superb game.	
COMBINED QL & PC VERSIONS	£69.95
Package deal for both QL and PC versions together.	
FTC2 DATA PRINT UTILITY	£9.95
[F 128k] Print out data from FTCII described above.	
THE FUGITIVE	£9.95
[F 512k] Text adventure game.	
COCKTAILS WAITER	£10.00
[F 256k] Database of 100's alcoholic drinks recipes.	
RECIPE SETS each	£5.00

PROGRAMMING

S EDIT editor	£20.00
[F 384k] DEMO VERSION	£2.00
EASYPTR III	£60.50
[F 256k] Pointer environment programming aid.	

EASYPTR III BUDGET VERSION	£35.00
DISA DISASSEMBLER	£29.00
[F 256k] Interactive pointer driven disassembler.	
BASIC REPORTER	£10.00
[F M 128k] BASIC programming aid, list variables, etc.	
BUDGET QLIBERATOR COMPILER	£25.00
[F M 128k] BASIC compiler for unexpanded QL.	
QLIBERATOR BASIC COMPILER	£50.00
[F 256k] Excellent BASIC compiler for the QL.	
QLOAD & QREF UTILITY	£15.00
[F M 128k 128k] Fast loader and cross reference utility	
MEGA TOOLKIT on disk	£25.00
on EPROM cartridge and disk	£40.00
Comprehensive toolkit of over 200 BASIC extensions	

FILE HANDLING

LOCKSMITHE	£14.95
[M 128k] Microdrive backup program, byte for byte!	
4MATTER & LOCKSMITHE	£23.50
[F M 384k] Complete software backup system.	
TOOLCHEST	£14.95
[M F 256k] Create customised microdrive doctor	
FILES 2	£12.00
[F M 128k] Simple to use popup file handling utility.	
FILEMASTER	£12.00
[F M 512k R] For bulk copying of files, & disk labelling.	
THE GOPHER	£12.00
[F M 128k] File finder, for disk, mdv, or hard disk.	
[F M 128k] DEMO VERSION	£2.00
WINBACK	£25.00
[F 256k] Hard disk and ED disk backup utility	
[F256k] DEMO VERSION	£2.00

DISPLAY SOFTWARE

BANTER	£25.00
[F 512k] Banner printing program for Epson etc printers.	
VISION MIXER 1	£10.00
[F 512k] Screen display system.	
VISION MIXER PLUS	£22.50
[F 384k] Enhanced, menu driven version	
PICTUREMASTER	£15.00
[F 256k] Screen drawing utility for use with V.Mixer, etc.	
PICTUREMASTER PLUS	£20.00
[F 384k] Enhanced version, extra features.	
UPGRADE TO "PM PLUS"	£5.00

PRINTER DUMPS

SIDEWINDER PLUS	£24.95
[F M 512k] Screen and DTP page printer dump utility. Suitable for 9 and 24 pin printers	

GRAPHICS

THE PAINTER V4.04	£25.00
[F 512k] 100% m/code pointer driven graphics program.	
THE CLIPART	£12.00
[F 128k] 3 disks packed full of screen pictures.	
PICTORIAL INDEX FOR THE CLIPART	£4.00
New printed index for the Clipart, with screen dumps	
QRACTAL	£20.00
[F 512k] Machine code fractals program.	
QRACTAL SCREENS	£2.00
[F 128k] Sample screen pictures made with QRactal.	
IMAGE PROCESSOR 2	£15.00
[F 512k] Graphics and picture editing, etc utility.	
[F 512k] DEMO VERSION	£2.00
PD2 CLIPART	£10.00
[F 128K] 2 disks of clipart screens.	
SCREEN SNATCHER	£10.00
[F M 128k] Grab screen displays from other programs.	
TEXT 'N' GRAPHIX	£20.00
[F 256k] Print text files containing screen dumps.	
[F 256k] DEMO VERSION £2.00	
TRANS24	£10.00
[FM 128k] Translate 9 pin graphics for 24 pin printer.	
SCREEN COMPRESSION	£10.00
[F M 128k] NEW! Save space by compressing pictures.	

TEXT

QTYP 2	£29.95
[F 512k] Spelling checker, uses pointer environment.	
BIBLE TEXT DISKS, PLAIN TEXT FORMAT	£20.00
BIBLE TEXT DISKS, QUILL FILE FORMAT	£20.00
[F 512k] Text of the King James Bible on disks.	
SPELLBOUND	£30.00
[F M 384k] Spelling checker, non-pointer environment.	
SPELLBOUND SPECIAL EDITION	£50.00
[F 512k] Improved version with larger dictionary, etc	
UPGRADE TO SPECIAL EDITION	£30.00
Please return master disk/cartridge when upgrading.	
QUICK POSTERS	£10.00
[F M 128k] Text poster maker for use with Star printers.	

DATABASES

ADDRESS BOOK & LABEL PRINTER.	£15.00
[F M R 384k] As used by DJC, database + label printer.	
[F M R 384k] DEMO VERSION	£2.00
DATA DESIGN	£50.00
[F 512k] Programmable pointer environment database	
FLASHBACK	£25.00
[F M 256k] Fast database written in machine code.	
FLASHBACK SPECIAL EDITION	£40.00
[F 256k] Enhanced version of Flashback	
QL GENEALOGIST SECOND EDITION	£30.00
[F 384k] Family trees and family history program.	
UPGRADE TO SECOND EDITION	£12.00
Please return master disk with upgrade order.	
BUDGET 128K GENEALOGIST	£12.00
[F M 128k] Cut down version for unexpanded machines.	
DISK INDEXER	£12.00
[F M 256k] Make a database of contents of your disks.	
DBEASY	£15.00
[F 512k] Database front end for Archive etc.	
DBPROGS	£15.00
[F 512k] A collection of Archive utilities and text tiles	

QL HARDWARE

MINI PROCESS CONTROLLER	£59.95
Relay switched outputs, controlled via QL serial ports.	
SOFTWARE TOOLKIT FOR MPC	£9.95
[F 128k] Extensions to simplify control the MPC.	
NETWORK PROVER	£3.50
A small box which plugs between networked QLs.	
QPOWER REGULATOR	£24.95

OTHER QL SOFTWARE

QPAC1	£19.95
[F 512k] Pointer environment utilities.	
QPAC2	£39.95
[F 512k] Files maintenance menu, hotkeys, etc.	
QTOP	£29.50
[F 256k] Desktop system, files, job control, etc.	
PRINTERMASTER	£20.00
[F M 128k] Printer control utility.	
HOME BUDGET	£20.00
[F M 128k] Domestic bills and accounts program.	
REMIND-ME	£12.00
[F M 128k] Dates reminder program. Very easy to use.	
REMIND-ME PLUS	£20.00
[F M 128k] Enhanced version.	
SCREEN ECONOMISER	£10.00
[F M 128k] Screen saver blanker.	
SCREEN DAZZLER	£15.00
[F M 384k] Like screen economiser, but instead of blanking out the screen activates moving objects or screen effects to save screen burn-ins, like some PC programs!	
SLOWGOLD	£5.00
[F 128k] Slowdown routine for Gold Card orany QL	
TASKMASTER	£25.00
[F M 384k] Task switching utility.	
DISK LABELLER	£10.00
[F 256k] Prints neat labels for your floppy disks	
THE CAT	£5.00
[F M 128k] List filenames in columns.	
ROB ROY BARGAIN PACK	£10.00
[F M 128k] Reviewed in Q L World August 1991	

THE SMALL PRINT! POSTAGE AND PACKING CHARGES: Software is sent post-free to UK addresses. To other countries, please add £1.00 per program for postage and packing (sent by airmail where possible). PRICES: All prices are shown in UK Pounds Sterling. PAYMENT: We can accept payment by cheque (in UK Pounds Sterling currency only, please) drawn on UK branch of a bank or building society, by Eurocheque with card number written on the back, Postal Order, or by these credit cards: VISA, ACCESS, MASTERCARD, EUROCARD or by CONNECT card. Please state the card type, number, expiry date, your address, and sign orders sent by post. We can also accept orders paid by credit card over the telephone. There is an answering machine for when I am unable to answer in person so that I can call you back later. Goods remain the property of DJC until paid for in full. PLEASE STAGE IF YOU REQUIRE SOFTWARE ON 3.5 OR 5.25 INCH DISKS OR MICRODRIVE CARTRIDGE.