QL Today

Volume 4
Issue 1
May/June
1999

ISSN 1432-5454

The Magazine about QL, QDOS, Sinclair Computers, SMSQ...

The Q40 is available now!

Contents

Advertisers

in alphabetical order

QL *Today*

ISSN 1432-5454

German office & Publisher:

Jochen Merz Software	Tel. +49 203 502011
Im stillen Winkel 12	Fax +49 203 502012
47169 Duisburg	Box1 +49 203 502013
Germany	Box2 +49 203 502014
	email: JMerz@j-m-s.com

English office:

Q Branch	Tel. +44 1903 200212
The Bank Volt	Mobile +44 836 745501
6 Coronation Buildings	Fax +44 1903 208070
Ham Road	Fax +44 1273 381577
Sussex BN11 2NN	email: qbranch@qbranch.demon.co.uk
United Kingdom	

Editor:

Dilwyn Jones	Tel. +44 1248 354023
41 Bro Emrys	Fax +44 1248 354023
Tal-Y-Bont, Bangor	email: dilwyn.jones@dj.softnet.co.uk
Gwynedd	
United Kingdom LL57 3YT	

QL *Today* is published bi-monthly, our volume begins on 15th of May. Subscriptions begin with the current issue at the time of sign up. Please contact the German or English office for current subscription rates.
Payment in DM (drawn on a German bank) can be made by either Cheque or Euro-Cheque, Payment in £ (drawn on an English bank) can be made by Cheque. Cheques should be made payable to Jochen Merz Software (German office) or QBranch (English office).

Credit Card holders may subscribe by either calling or sending their Credit Card number and Expiry date. Credit cards will be charged in DM (German office) or in £ (English office).

We welcome your comments, suggestions and articles. YOU make **QL *Today*** possible. We are constantly changing and adjusting to meet your needs and requirements. Articles for publication should be on a 3.5" disk (DD or HD) or sent via Email or into one of the JMS-BBS's. We prefer ASCII, Quill or text87 format. Pictures may be in _SCR format, we can also handle GIF or TIF To enhance your article you may wish to include Saved Screen dumps. PLEASE send a hardcopy of all screens to be included. Don't forget to specify where in the text you would like the screen placed.

Article and Advertising deadlines are as follows:

Issue 1: 30 April	Issue 2: 30 June
Issue 3: 30 August	Issue 4: 30 October
Issue 5: 30 December	Issue 6: 30 February

Editorial

Dilwyn Jones

The Q40 is with us at last and a few lucky customers already have theirs to tinker with, albeit with a virtually-finished operating system at the time of writing. Problems like producing the EPROMs for it have meant the launch has not been that easy for TF Services and Q-Branch (when did you last see a computer launched without problems?) There have also been sticky little points like the incompatible hard disk formats used by QDOS Classic and SMSQ/E, although this is being addressed. All this is snipping about the edges and splitting hairs though, for the Q40 is one of the most significant advances for the QL scene for a long time. Peter and Claus Graf have put a lot of time and effort into the design of the Q40 and it's here at last - more on pages 7 and 57. Although seeming to take an eternity to arrive, the colour drivers, the sound sampling and playback of the Q40 and other features will make this THE product to ask for as a Christmas or birthday present this year. Hopefully, we'll have a user's report on the Q40 in the next issue.
I have finally managed to twist Jochen's arm into writing an article on Things, and he even threw in an article about Events for good measure. And for those who think this sort of thing may be too advanced, try Mark Knight's new series on BASIC programming. In this issue he deals with FOR and REPeat loops. Thankfully, our cover disk with the last issue went out with no major problems, for once. As usual, some managed to find areas for improvement within some of the programs and the authors (e.g. Geoff Wicks and Beginners' Club) moved to produce new versions of some of the programs, see the news pages for details.

Do you have any ideas for what we can put on a future cover disk? Would you like a games collection for example? Write and let us know.
Our competition on page 42 of the last issue to make a meaningful sentence of QL Keywords produced a disappointing number of entries, not enough to award a prize in fact. We'll hold the competition open for this issue to see what happens. All you have to do is make up a sentence from the QL BASIC keywords and you could win a prize.
Meanwhile, Jochen and my girlfriend have conspired to make a competition out of embarrassing me - see the competition on the next page. I make an easy target, and you can win a little prize out of it, so let's hear from you. I'll take it all on the jaw!
Hope you've all taken note from last issue's editorial column that the publication date of QL Today has moved - if you thought this issue was late, now you know why!

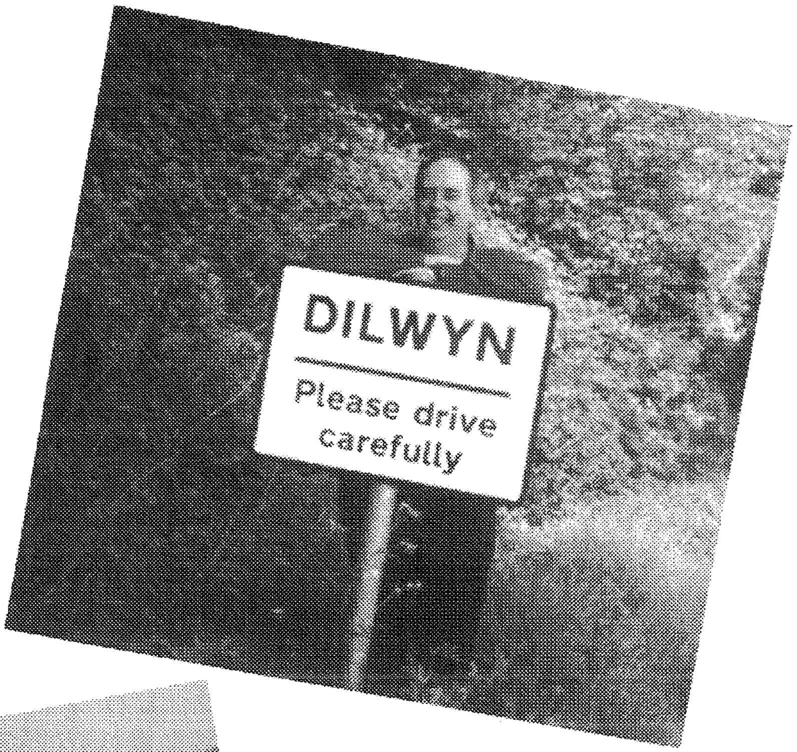
Cartoon

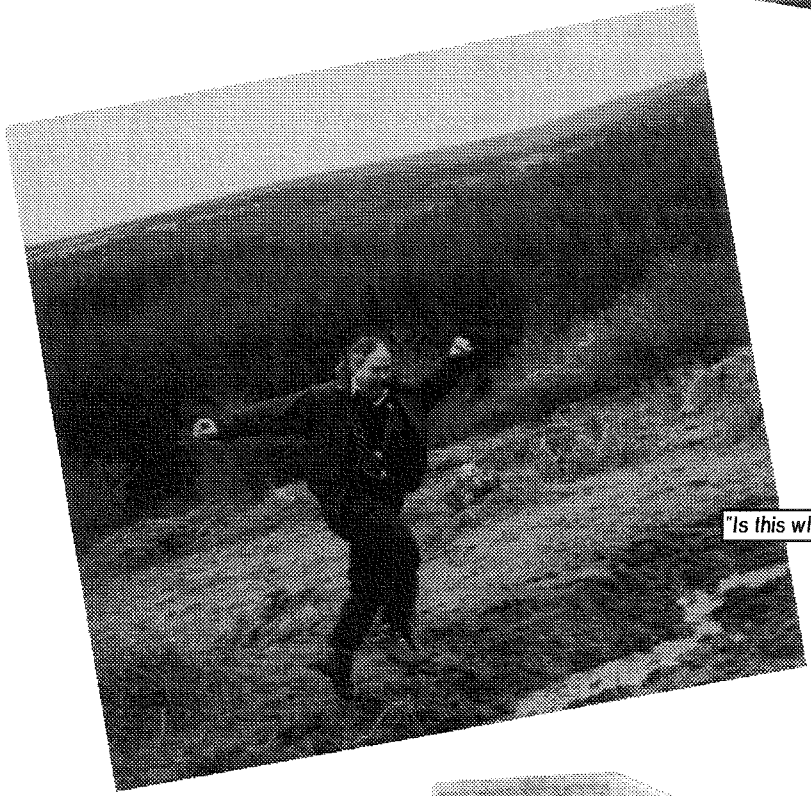
Bad Hair Day at TF Services

Fiona Jennings

Embarrass the Editor Page

Since the editor is normally rather shy about appearing on these pages, here are 3 pictures he would rather didn't appear in QL Today, taken by his girlfriend Chris.

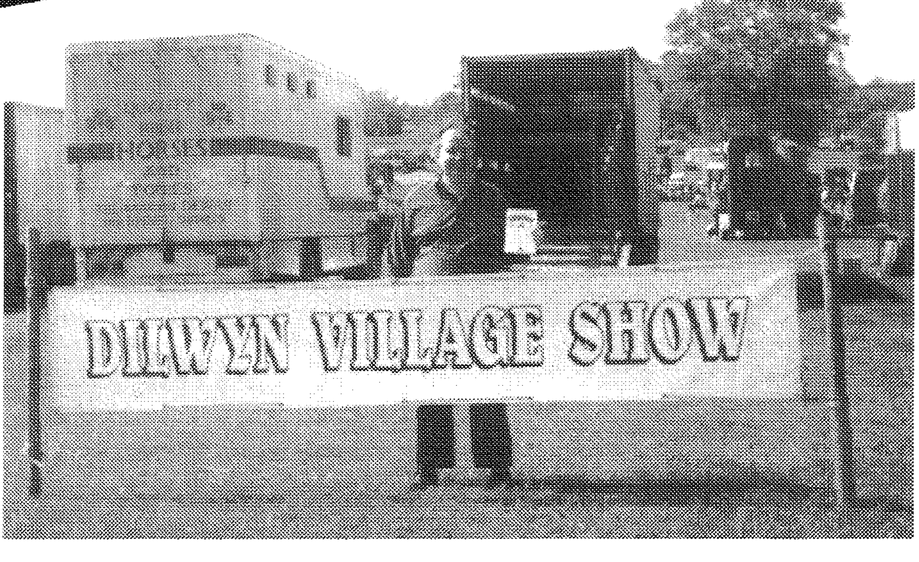




A small free software voucher from JMS is on offer for the best captions for these photos. Chris herself has suggested one for the middle photograph, taken when the editor "escaped" from QL Today duties to Ireland recently, to get you started!

"Is this what he means by 'the Quantum Leap'?"

The other two photographs were taken during a visit to his namesake village in Herefordshire, England, recently.



News

Qubbesoft & Qubides

Ron Dunnett writes:

It has come to the point where I now have no new QUBIDE's left for sale, but there are still QL/Aurora Users out there who require a QUBIDE or two. Unfortunately at the moment there is not enough interest to warrant another batch being built.

I therefore am appealing to anybody out there who has a QUBIDE which is surplus to requirement could they please contact me.

I can be contacted in various forms as follows:

Tel/Fax: **+44 (0)1376 347852**
Mobile: **+44 (0)780 1641616**
Email: **ron@qubbesoft.freeserve.co.uk**

PARAGRAPH

Version 1.04 of the Prowess based pointer driven wordprocessor Paragraph is now available, containing many bug fixes over the previous versions.

V1.04 may be freely downloaded from Thierry Godefroy's Web site:

http://www.imaginet.fr/~godefroy/english/ download.html

For users who registered their free trial versions, v2.00 has now been sent to them.

Francois Lanciault, the author of Paragraph, may be contacted by email: **franat@aei.com**

New Software from Tim Swenson

FileConfig

Based on BasConfig (the Config Block creator for Compiled SuperBasic programs). Instead of having to hand enter all of the data, the data is put into a file and the Config Block is generated from it.

If you are creating large Config Blocks, this will save you lots of time in re-entering the Config Block definition if changes need to be made.

Structured SuperBasic 2.6.1

New Features:

- 2nd command line argument of Starting Line Number. Now SSB can be used with the Make utility. Includes an article on how to use Make with SSB.
- SSBGO - A program that calls up SSB, and then Qlib. Designed to be used from inside MicroEmacs. Once you save a file, execute SSBGO, wait a bit, and you now have an executable to test.
- Included article on using the Revision Control System (RCS) with SSB.
- Improved error reporting.

If anyone is interested in beta testing these programs, let me know and I can send you a zip file.

Tim Swenson email address: **swensont@jack.sns.com**

RWAP Software

BIG Britain.map has now been released for Q-Route. This is THE latest map of Britain for use with Q-Route. It is twice the size of the original with much more detail and lots of improvements. The coastline alone is now nearly 2000 nodes instead of the original 500 (you will need to change this parameter in QRConfig for the map to load). Q-Route v1.07 and 1MB RAM is the minimum requirement. This map costs £5. Maps of Ireland (£5), West Yorkshire (£1), Catalonia (£1) and Belgium (£2) are also available for use with Q-Route. An update of the original map of Britain which can be used on smaller systems is also available for £1.

The following latest versions of software have now been released: NEMESIS MKII (v2.02), 3D TERRAIN (v1.24) and GREY WOLF (v1.8) - these all incorporate the latest version of the Turbo Toolkit and fixes to ensure they will work on the latest QL versions.

Q-Help has now reached v1.01 (upgrades to this version are free - send an SAE and master disk). This now fully implements the features described in Volume 3 Issue 6 of QL Today. People who already own the SBASIC/SuperBASIC Reference Manual and wish to use this program with it need to upgrade to the latest support disks (cost £2). These support disks were last changed on 26/9/98.

Q-Index has now reached v1.02 (users of Q-Index v1.01 can upgrade to v1.02 by sending their master disk and an SAE).

Open Golf v5.20 has now been released. This fixes problems which the program had running under SMSQ/E with Aurora machines and other high-resolution displays.

Dave Walker

Dave Walker's new Web site **www.itimpi.freeserve.co.uk** includes details of his latest programming efforts, such as v4.17 of the QL-PC disk file transfer utility Discover (March 99) which fixes some bugs reported in earlier versions. Also available is version 4.8 of George Gwilt's assembler program (not the same version as the GWASL program on our last cover disk), and the 20/3/99 release of the C68 binaries which fix a couple of problems in the earlier v4.24f releases.

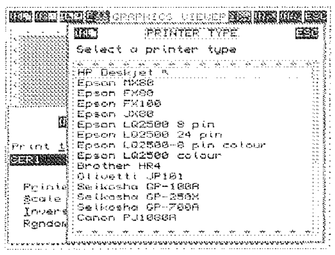
Jonathan Hudson

QTPI v1.66 (18/3/99) is now available, which consolidates the experimental 162x releases. It includes copious documentation and all XPR proto-

cols. Resize via escape sequence now works. V1.66 removes the dependency on the signal extensions.
QFAX v2.85 (30/3/99) includes some Year 2000 (Y2K) fixes. This version consolidates all minor releases since 2.80, full documentation (2.83) etc. Gives qfax MDM[12], adds user definable print filters and latest qfm (menu driven front end for QFAX) release.
QVM 30/3/99 also includes Year 2000 (Y2K) fixes. QVM (Quintessential Voice Mail) is a Beta voice mail utility.
All are available from the Web site address
www.jrhudson.demon.co.uk

Dilwyn Jones

GRAPHICS VIEWER V1.15.
Version 1.15 of my Graphics Viewer program is now available from Quanta and most sources of QL PD and Freeware.
This version fixes many problems in the printer drivers, most importantly the HP Deskjet screen dumps now work (previously failed to work on many systems).



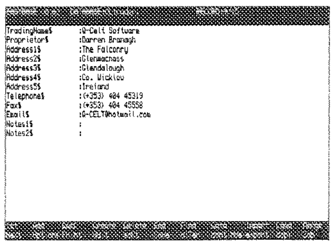
```
PRINTER TYPE
Select a printer type
HP Deskjet
Epson MX80
Epson FX80
Epson FX100
Epson JX80
Epson LQ2500 8 pin
Epson LQ2500 24 pin
Epson LQ2500-8 pin colour
Epson LQ2500 colour
Brother HR4
Olivetti JP101
Seikosha GP-100A
Seikosha GP-250X
Seikosha GP-700A
Canon PJ1080A
```

CLUB QL

Mike Kenneally, editor of Club QL International's disk based newsletter, has sent a note explaining that he has been in hospital for treatment to his eye, so current issues of their newsletter have been delayed. He hopes to be back in action by the second half of May.

Q-CELT SOFTWARE

Version 0.62 of the EasyBase database was released on 20th April. In addition to fixing minor bugs in previous releases, this version radically improves the Import and Merge facilities allowing a wider range of import formats, and allowing the use of Merge to completely redesign field layouts - creating new versions of databases with more or less fields than the old file simply by using Create to make a new database with the required layout, then letting the Merge command reorganise the old file to build the new database.
The manual is still supplied as a file on disk to keep costs down, but has now been split into sections to make it easier for users with Quill and smaller memory systems to load and print the manual. The manual now includes a list of files supplied, a section listing the file format, and a short list of Frequently Asked Questions about EasyBase.
A utility program is now supplied to allow renaming of fields within a database if required.
Bugs fixed include providing enough stack space to sort larger files, stray testing messages no longer printed, and the parameter errors in reverse order sorting have been fixed.
Contact Q-Celt for upgrade details.


```
TradingName$     :Q-Celt Software
Proprietor$      :Darren Branagh
Address1$        :The Falconry
Address2$        :Glenmacnass
Address3$        :Glendalough
Address4$        :Co. Wicklow
Address5$        :Ireland
Telephone$       :(+353) 404 45319
Fax$             :(+353) 404 45558
Email$           :Q-CELT@hotmail.com
Notes1$          :
Notes2$          :
```

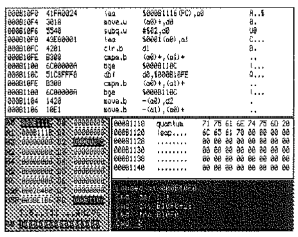
IRISH QL SHOW

Later this year, Darren Branagh is hoping to organise another Irish QL show, this time near the border between Northern Ireland and the Republic of Ireland to make it easier for visitors from the north to attend. Please contact Darren to encourage him if you hope to attend or would like to help.
Q-Celt Software, The Falconry, Glenmacnass, Glendalough, Co. Wicklow, Republic of Ireland.
Tel: +353-404 45319
Email: **Q_Celt@hotmail.com**

QUANTA NEWS

Quanta have bought the copyright to the Assembler Workbench program (as previously sold by Talent) and it is now available to members free of charge from the library. Although the original version is still available to those who want it (ie, v1.4 on protected microdrive) it has some limitations when running on modern systems. Therefore a new Quanta version (V2.02) has recently been released which overcomes these problems and in addition has a some new useful features. (Please don't confuse the Quanta release with other non-Talent versions).
Assembler Workbench is a complete machine code development package. It consists of an assembler, disassembler and machine code monitor program. It is ideally suited to beginners since it has an easy to use front end; small assembly programs can be entered directly into the monitor program and then immediately run and traced through whilst watching the effect on the 68000 CPU registers (and optionally an area of memory). The users program can be stepped through one instruction at a time by simply hitting the space bar or by toggling on/off a free running mode with the F1 key. Invaluable for finding where your program goes astray!

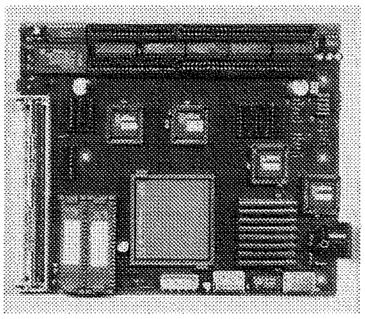
Notswithstanding its ease of use it is quite possible to use Assembler Workbench to assemble, disassemble and debug large programs. It has some novel features like the ability to trace through Supervisor code (ie, Traps) and interrupt users machine code programs with the ESCape key. The monitor's windows can be dynamically linked to any output devices/files (ie, output sent to both window and device/file simultaneously). There is also a useful dual screen mode option (when not working under the PE) which allows you to switch the display between the monitor and the users program display (automatically when tracing). With Minerva this function also works in the second screen (v2.02).
V2.02 has been tested and works on QL/GC/SGC QDOS/Minerva with or without PE/SMSQE and with QPC. It has not been tested on QXL or Atari's. Although it appears to run OK on a Q40/SMSQE it has not been fully tested yet (especially with the caches). All the changes made since the last commercial release V1.4 are listed in the changes_doc (or _txt) on the library disk. Comments or bug reports welcomed.
Assembler Workbench comes with an indexed on-line help manual, this help file has also been provided in Quill and plain text formats. Since the manual assumes some knowledge of assembler and the 68000 processor some articles will shortly be published in Quanta to introduce novices to these subjects through practical example specifically using the program.
dave.westbury@ btinternet.com



```
000B10F0  41FA0024   lea      $000B1116(PC),a0     A..$
000B10F4  3018       move.w   (a0)+,d0             0.
000B10F6  5540       subq.w   #$02,d0              U@
000B10F8  43E80001   lea      $0001(a0),a1         C...
000B10FC  4201       clr.b    d1                   B.
000B10FE  B308       cmpm.b   (a0)+,(a1)+          ..
000B1100  6C00000A   bge      $000B110C            l...
000B110C  51C8FFF0   dbf      d0,$000B10FE         Q...
000B10FE  B308       cmpm.b   (a0)+,(a1)+          ..
000B1100  6C00000A   bge      $000B110C            l...
000B1104  1420       move.b   -(a0),d2             .
000B1106  10E1       move.b   -(a1),(a0)+          ..

000B1118  quantum    71 75 61 6E 74 75 6D 20
000B1120  leap....   6C 65 61 70 00 00 00 00
000B1128  ........   00 00 00 00 00 00 00 00
000B1130  ........   00 00 00 00 00 00 00 00
000B1138  ........   00 00 00 00 00 00 00 00
000B1140  ........   00 00 00 00 00 00 00 00
```

Q40 NEWS

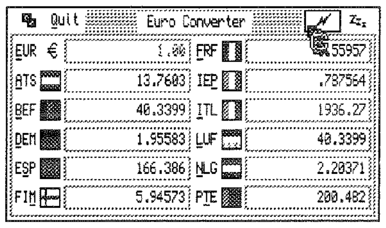
The Q40 system is now available from TF Services and QBranch, at prices starting from 310 pounds for a Q40 board with I/O card and no RAM on board, or 330 pounds with 16MB RAM. Up to 32MB of RAM can be used. The QDOS Classic operating system can be supplied with at as what TF Services call a "no cost option", while SMSQ/E is available for an additional 30 pounds. Q40s are at long last being sent out to customers and there have been some pretty encouraging comments from users, especially about its speed. At the time of writing, SMSQ/E was not quite complete, but certainly useable. There is a lot of information available on Tony Firshman's Web site
www.firshman.demon.co.uk/q40.htm
including two Q40 specific downloads, one of which allows you to unlink QLiberator compiler runtimes from older compiled BASIC programs (pre-version 1.35 compiled applications seem to use high memory address bits as a temporary store) and the other a short text file which gives loading hints for the ProWesS system.
TF Services will also quote you a price for building a complete cased Q40 system to your specification.



Last minute news: SMSQ/E 1.92 for the Q40 is ready and being shipped. It is very stable and also provides access to the sampling sound system of the Q40.

BEGINNERS' CLUB

Beginners' Club have released version 1.20 of the Euro Currency Converter featured on our cover disk with the last issue.
In this version, pressing F2 displays the currency conversion rate.
The program is freely downloadable from their Web site:
www.geocities.com/SiliconValley/Lab/5011/soft-e.htm
Beginners' Club, Via Crispi 20, 13100 Vercelli VC, Italy
Email: **beginners@geocities.com**



EUR €	1.00	FRF	[illegible]55957
ATS	13.7603	IEP	.787564
BEF	40.3399	ITL	1936.27
DEM	1.95583	LUF	40.3399
ESP	166.386	NLG	2.20371
FIM	5.94573	PTE	200.482

JUST WORDS!

CHEAPER BUT BETTER

With the release of STYLE-CHECK 3 JUST WORDS! has completed the upgrading of its program range for use with the pointer environment. The price of the programs has also been reduced. QL-Thesaurus, Style-Check and Solvit-Plus are now £10 each or £25 for the complete set.

STYLE-CHECK 3

STYLE-CHECK 3 has now been released. This is a major upgrade of the program including the following new features:
Pointer driven and mouse compatible. High resolution screen compatible. User configurable screen colours. QMENU compatibility. Upper case character support. Additional on-screen help customised to most sentences. Updated manual.
Among the new checks are:
Adjacent word repetition. Split passives. Split infinitives. Hidden verbs. Mismatched quotes and brackets. Negatively formulated sentences. Homophones.

The upgrade price is £2.50 + return of the master disk. If required the upgraded manual is an additional £1.50. (Payment can also be made in UK first class stamps or I.R.C.s at the rate of 2 I.R.C. = £1)

QTYP SPANISH DICTIONARY AVAILABLE

A QL-user in Spain has sent me Spanish and French QTYP dictionaries of almost 175,000 and 209,000 words respectively. He was not sure of the accuracy and so I tested a random sample of every tenth word in each dictionary using the spell checker of Lotus Word Pro. There was a matching rate of 94% for the Spanish dictionary and 96.7% for the French dictionary. Given that a mismatch does not necessarily mean an incorrect word this is a high rate of accuracy.

Should you wish to have these dictionaries please send me an addressed envelope enclosing a disk and either 2 first class UK postage stamps or 2 I.R.C.s

FRENCH AND ENGLISH DICTIONARIES

From two separate sources I have received the same comments on the French dictionary supplied with QTYP. This is that many verbs are not fully conjugated in the second half (I to Z) of the dictionary.

This has stimulated me to look at the 65,000 word QTYP English dictionary supplied with Spelling-Crib. A weakness of this dictionary is that it contains both UK and USA spellings. There were also more errors in it than I had expected. I have now removed the USA spellings and made some further corrections. By the time this appears I hope to have a version of the dictionary available conforming to USA spellings. The revised version of the dictionary is available with version 2.10 of Spelling-Crib.

SPELLING - CRIB v. 2.10

Even the best efforts of the Beta-testers are nothing compared with the experience of the first users in testing a program, and Spelling - Crib was no exception. Not long after the last issue's cover disk landed on the doormats I received several telephone calls reporting problems. By the time you read this version 2.10 will be available from the usual sources.

CONFIGURATION BLOCK: I am not sure how it happened, but the program acquired a rogue configuration block, which gave problems for people using Menuconfig. This has now been corrected, although the size of the input window will restrict the device and dictionary name to 31 characters.

WORD LISTS: Several users reported difficulties in loading a new word list, although some had obviously not read the manual, or had not understood how QTYP works. All were Text87 users.

When you use QTYP among several programs only one dictionary can be active at a time. It would not be possible to have, say, an English dictionary in Spelling-Crib and a French dictionary in Text87. They must share the same dictionary. One of the programs operates as "mother" program, and this program must be used to load another dictionary.

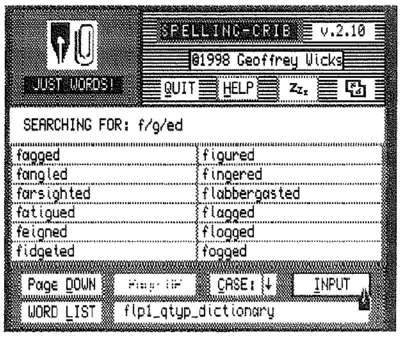
Suppose you first load the QTYP extensions, then Text87 and finally Spelling-Crib. At this stage Spelling-Crib is the mother program. If you type a document, Spelling-Crib remains the mother program. However, the moment you press Shift + F1 or F6 in Text87 to spell check, then Text87 takes over as mother program. Should you close Text87 down, then Spelling-Crib again becomes the mother program.

To add to the complications loading new dictionaries is badly explained in the Text87 manual. The keypresses are F3, Config, Spelling, Close, Spelling, Open.

When Spelling-Crib 2.00 discovered it was no longer the mother program, it made the WORD LIST command inactive and displayed a message, "Word list set by other program." In version 2.10 I have kept the WORD LIST command active at all times to avoid complications if Spelling-Crib has again become the mother program. SUPER-CRASH: I am grateful to John Gregory for discovering that entering f/g/ed, causes a complete program crash. This is because the program attempts to check the word "fed", which is shorter than the search string. There was a length check made at one point of the program, but this had to be repeated in a slightly different form at another point. This is corrected in version 2.10, but please let me know of similar crashes if you find them.

I have also revised and corrected the QTYP dictionary.

Geoff Wicks, 28 Ravensdale, Basildon, Essex SS16 5HU, United Kingdom.
Tel: +44 (0)1268 - 281826.



```
SEARCHING FOR: f/g/ed
fagged          figured
fangled         fingered
farsighted      flabbergasted
fatigued        flagged
feigned         flogged
fidgeted        fogged
```

Q-Branch

Q 40 is now shipping with SMSQ/E V2.92. Some problems which occurred in the first release are fixed in this version. The Turtle Graphics is now working and there is support for the BEEP functions. At

the moment I have no real details on what this version supports apart from that. There are still some delays in shipping because of EPROM problems.
We have made an arrangement with Francois Lanciault to distribute Paragraph and collect registration fees. We will supply the demo version for 1.50 UKP + postage and the full registered version for 18.00 UKP.
For address see QBranch advert.

Religion Catalogue from George Morris

George Morris has sent us a copy of his latest catalogue of religious materials, listing some 48 products in all, at PD-type prices.
It lists various Bible versions available as text files or DOC files including the King James Bible, New King James Bible, NIV Bible, The Concordant Literal New Testament, Hebrew and Koinee Greek testaments, Bible Dictionary, Cross References, Clipart disks, Sermons disks, Hymnal database, Bible Companion, QTYP Anglicised Greek dictionary and other study aids, plus various leaflets and even a cassette based course in Greek to help with the Koinee Greek based Bible studies, all presented and explained in the catalogue in a friendly and helpful manner. If you are religious, or wish to obtain such material for use in your local church, for example, you could do worse than get in touch with George Morris and ask for a copy of this catalogue - most of the products are at PD library-type prices. The catalogue includes an order form.
George Morris,
67 Wood Lane,
Sutton Coldfield,
West Midlands,
England, B74 3LS
Telephone: (+44) 121-353 8571

Jochen Merz Software

QPC2 News

A downloadable Pre-Beta2 test version of QPC2 is now available from Marcel Kilgus's QPC support pages. While most features now seem to be workable, Marcel says he wishes to do more work on improving the serial and parallel ports. Another breakthrough: I was able to format disks properly with the Beta version of QPC which Marcel sent me 2 days ago, so there is not much left to be done and QPC2 will be ready soon. Therefore, QPC2 will most likely be shipped from End of June. Price and upgrade price from QPC1 is not set yet. All JMS customers will receive a special newsletter when QPC2 is ready.
Version 1.45 of QPC1 is available from Jochen Merz.

QSpread99 is ready too. Not much has happened since Oliver Fink stopped working on QSpread and passed the sources over to JMS. About 2 or 3MB to work through, which took quite a while. In the past three years, there were just a few bug fixes here and there, but Bernd Reinhardt and myself did some major rework on QSpread during our skiing holiday (as usual, the only time to program - no phone which interrupts, no email to be answered etc.). QSpread99 was brought up to "QD standards". Not only became a number of customer enquiries reality, we fixed a lot of other, known bugs, added a new QD-style toolbar, QD's nice and useful help popup windows, improved the menus, rewrote the whole print menus and the way printing worked (you can now easily write your own printer driver filters if you like), added other bits here and there and updated the manual. More details on what is new in QSpread99 can be found in the JMS ad elsewhere in this issue of QL Today.

Other software updates were done to FiFi (which is now V4.19), QD98 (now A.04), DISA (now V3.04). Roy Wood has tracked down a bug in MenuConfig at the Clevedon Workshop which only affected configured files where a single keystroke was configured (the only two we know of are QPAC2 and QTYP). You had to configure the keystroke in order to corrupt the configuration item. As MenuConfig comes free with more or less every piece of JMS software, you will get it automatically with updates of QD, FiFi, Upgrade of QSpread etc.

MicroEMACS v4.00 News

Just to let you know that an updated MicroEMACS v4.00 release (13/05/99 release) is available at the following URL:
http://www.imaginet.fr/~godefroy /english/download.html
You may also look at:
http://www.imaginet.fr/~godefroy /english/QLnews.html
for a log of the changes since the last release.

EEC (Bill Richardson) is searching for ED Disks

If you have ED disks which you do not need anymore, then please contact Bill Richardson (see his ad in this issue for details). ED disks are not manufactured anymore, and many QLers use them as a kind of small harddisks. So if you converted your system to real harddisk and you have some surplus disks sitting somewhere in your cupboard, do others a favour and contact Bill.

Quanta AGM and show review

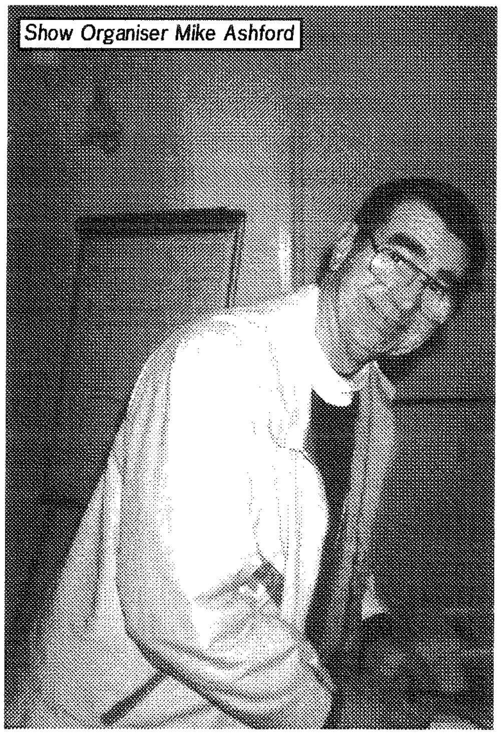
Darren D. Branagh

On 25th April last, the annual pilgrimage to Clevedon took place, courtesy of Quanta. This was the first time I have attended a Quanta AGM meet, and I enjoyed the occasion immensely.

I travelled with Dilwyn Jones and Chris yet again, staying in the fabulous Walton Park hotel - the same venue as the show and AGM itself. We travelled down on the Saturday evening, and met up with the usual QL entourage in the bar. After a round or two of drinks, we decided to find the local Indian restaurant. This proved a lot more difficult than we first thought, taking at least 45 minutes to locate! Roy and Steve decided to opt for a local pub meal, while the rest of us struggled on hungry and tired.

Several minutes later, we had a close call when a speeding Audi *[not Jochen! - Editor]* raged around a corner and crashed on the opposite side of the road, taking out a couple of concrete bollards and a pub menu sign in the process. A close call, if he had swerved to the other side instead, he'd probably have killed us all. Stuart Honeyball made sure all was OK, then we continued on our way. Thankfully nobody was hurt.

Soon after, we found the restaurant - I joked that we should ask Rich Mellor to amend Q-Route to find all Indian restaurants in a certain area, for future use! Still we got there in the end. At this stage there were 8 of us - Dilwyn, Chris, Stuart, Dave Walker, Alf Kendall, Jochen Merz, Bernd Reinhardt, and myself. Q40 chat dominated, as did QL Internet possibilities, and a fairly diverse range of other topics. After a really good meal (worth waiting for) we returned to the Hotel.

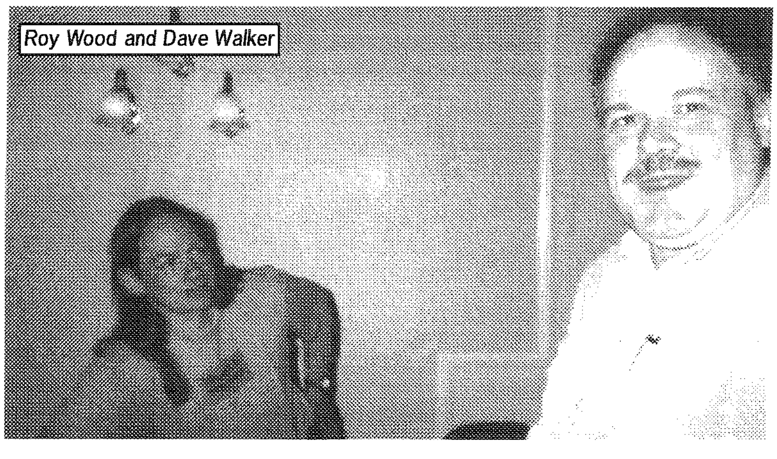

Show Organiser Mike Ashford

Sunday morning soon arrived, and after breakfast the traders gathered downstairs to set up shop. We had access to two rooms, one for the traders and the other for talks and demonstrations, as well as the AGM itself later in the afternoon.

All the usual vendors were present. Q-Branch took pride of place just inside the door, and rightly so as this was where the new Q40 could be seen actually working and running software. Roy had been telling us in the bar the previous night just how fast this hardware was in comparison to even a Super Gold Card, but you really have to see it to believe it. It can only be described as blisteringly quick. It uses standard RAM SIMMs, so upgrading it to more RAM is easy and readily available. I read recently via the QL-users email list that one user in Italy has just received his, with 32 Meg of RAM, and a 2 Gigabyte Hard drive, all in a Tower. This really brings the QL into the Millenium with pride, all we need now is internet access, which is being worked on. Details of the Q40 are available from Roy or Steve at QBranch.

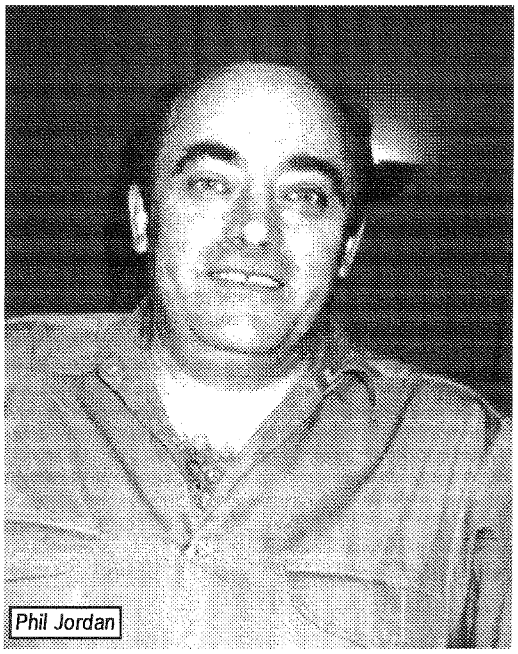
Tony Firshman was next to Roy, sporting a Q40 T-Shirt, showing where his allegiance lies! Tony was also doing good business in his primary products - namely the Super-Hermes, Minerva and the excellent RomDisq, and excellent alternative to a hard drive on a QL.

Next was Rich Mellor of RWAP. The Superbasic Reference Manual was available, as was Rich's varied collection of great games and adventures for the QL. The second installment of my QL adventure series features another RWAP game, NEMESIS - see elsewhere in this issue.

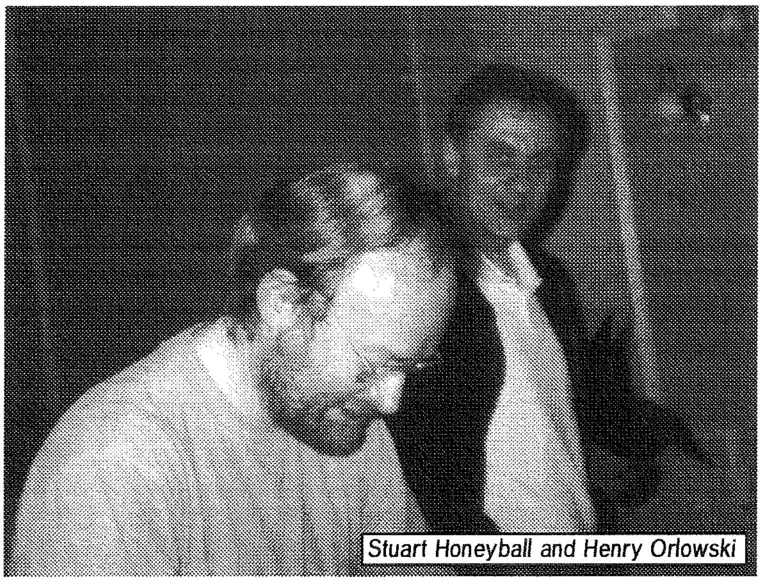

Roy Wood and Dave Walker

Quanta were at the top of the Room. Bill Newell and John Taylor were on hand as

usual, Bill taking subscriptions for membership renewals and John was in charge of the varied collection of items Quanta had for sale, including sweatshirts, books, hardware and software. John Gregory was there with the Quanta Library, accepting new programs for addition to it, and copying disks for members.
Ron Dunnett of Qubbesoft was having a "Spring Sale" - many of Ron's hardware and software Items were greatly reduced, including Aurora motherboards for £70 instead of the usual £100. Also Ron had several empty IBM desktop cases with PSU's for only £10 each, and his vast PD library was available, and catalogues for this were free on his stand.
I was on hand at the Q-Celt stand. I gave a demo of EasyBase, and astounded some people by creating a name and address database from scratch in only 3 minutes!! It really is that easy to use. Many people bought it for its array of easily configurable options - right down to the colours on the screen can be configured. The Dilwyn Jones Collection was also on sale, the usual 11 programs for the price of just one. Dilwyn was set up with his laptop next to my stall and was able to answer some of the more complex questions about this excellent compilation, and EasyBase.
There was a varied collection of Hardware and Software on the bring and buy stand, including RGB Monitors for £10, dot matrix printers for a fiver, and Trump Cards for around a tenner. Many items were free to anyone who wanted them There was even an old OPD machine looking for a good home!!


Phil Jordan

Jochen Merz was next to the Bring and Buy, where he launched the new version of Qspread, called Q-Spread 99. Q-Spread is a pointer-driven spreadsheet program, and this is a marked revision and improvement on the original, and was selling well. QD98 was also available, and has undergone further minor improvements since the Hove Show in February. A Pre-Beta test version of the new QPC, called QPC2, was on Jochen's stall too. This was a demo test version obviously, with the more worthwhile features missing, but it gave some of us the chance to see how work is progressing on this new piece of software. QPC2 will multitask with Windows easily - switching between QL and PC will be as easy as pressing ALT&TAB. Also lower resolutions will be able to be "stretched" to fit the whole screen, allowing for larger fonts - a great feature if your eyesight is a bit lacking with age!!
George Morris was also present with printed catalogues of his range of religious software for the QL, ranging from Clipart to various versions of the Bible, quite a collection of software.
Lastly, and certainly not least, was Bill Richardson of EEC. He had new Z88's and QL's for sale, including the new 512K RAM versions of the Z88. He was taking subscriptions for Z88 USER magazine, and also now has obtained a supply of new Z88 CD-ROM's. These are of use to people who have a PC with a CD-ROM drive on it, and they contain over 1,200 files for use on a Z88!! These include a Windows Z88 Emulator, the Z88 SourceBook (also available for the QL) and hundreds of eprom images for transfer to the Z88. In all, there are over 12.5 Megabytes of data on the CD, a huge amount in Z88 terms, as most Z88 eproms are only 32K in size! It


Stuart Honeyball and Henry Orlowski

costs £18 and is available to Z88 USER subscribers for only £12 for a limited time.
The AGM itself started at 4pm, the main proposals passed were ones to reduce slightly the size of the committee in view of the somewhat smaller membership, and thanking two committee members who had resigned recently for their service. The main officials were re-elected to their posts unopposed. During discussion at the AGM itself, we discovered from Cyril Phillips that a couple of new faces, actually a couple of young lads, had attended the workshop and gone home the proud owners of their first QLs, having bought one each. A happy note to end a great weekend on. Just goes to show how much life there still is in the QL, and with developments like the Q40, QPC2 and (hopefully) internet access, long may it all last.

So what are these Things? - Part 1

Jochen Merz

I know I should have written this article some time ago, but I was not sure what to write and how to start.

I decided I will start with some general explanation about Things, tell you what you could do with them (most likely, you are using them for quite a while without knowing) and carry on explaining how to use them and even how to create them. Of course, that's not going to fit into a single issue and as we don't want to fill a complete issue with assembler listings only, I'll split it into parts. Take it as an additional tutorial part to Norman's assembler series.

To make sure you don't get too confused, I will put the term "Thing" in upper case whenever I mean the SMSQ meaning of Thing.

The name Thing was used because a Thing can be virtually everything. It can be a device driver, data area, it can be a menu, it can be a system extension, it can be a program ...

The first advantage of a Thing is that it has a unique name. You can identify a Thing by its name, and find it in the system's memory. A Thing can be positioned anywhere in the memory of your computer, so the name is the only way to find it (similar to how you refer to a file by specifying its filename).

If a job wants to use a Thing (there can be different ways of using a Thing depending on what it actually is, but more on this later). First of all, the job has to try to find out whether a particular Thing exists in your system - otherwise it can't use it. Similar to a file - you have to try to open it in order to find out if it exists.

If the Thing exists and the job is allowed to use it (there may be limitations which depend on the Thing, some Things may be used by one user at a time, others may have more than one user ...) then the job is marked as a user of this particular Thing and it now "knows" that the Thing exists and gets the address of the Thing. It is even possible to specify a timeout when trying to use a Thing (something which is not possible when opening a file - it either works or fails immediately) - useful for Things which may have only one user at a time, so you can try using it for a predefined time and only then give up.

The address of the Thing is what you get, and that's all you need, because this is the main idea: a kind of "filing system" for parts of your computer's system, together with a user registration. Sounds simple, doesn't it? And it is simple.

A job is a user of a Thing until it frees (similar to "close") it. As long as a job is a user of a Thing, its "life" depends on the presence of this Thing. As SMSQ is, just like QDOS, a self-cleaning system (which means that if you remove something from the system, all items related to this "something" are also released/closed etc.). If you remove a job in SMSQ/QDOS, then all channels which are owned by this job are automatically closed by this system, all memory owned by this job is returned to the free memory list, all jobs owned by this job are removed as well and this works recursively until all resources owned or used by this job are freed.

Things go a step further. When a job becomes the user of a Thing, it seems logical that it wants to do something with it or relies on its presence. If the Thing is removed, then naturally all jobs which are registered as users of this Thing will be removed by the system as well - how could a job continue using the Thing if it is gone? This may lead to unpredictable results and we prefer having a stable system. It may sound a bit tough to remove jobs but think about it - what else can be done?

Different Things...

As I have already mentioned ... it is possible to have different kinds of Things. Theoretically it would be possible that every piece of used memory could be a Thing. This would probably be "overkill", but it would be possible. Not every memory allocation needs to be recognisable by name, but if you think about it, you will find that it could be useful if larger parts of your memory, which contain different parts of the operating system and device drivers were recognisable by name. You could customise your system while it is running, add drivers, remove drivers, update drivers.

In the past, adding another RAM-Disk driver while another RAM-Disk driver is used will not lead to a clean system. The old RAM-Disk driver remains in the system, you will find two "RAM" entries in the QPAC2 list of devices, but the first one is not removed. You can still access it - it may be a bit tricky, but it is possible. And what about files being open to the first RAM-Disk driver? They remain open until you close them. If you open the same filename again it will be opened to a different RAM-Disk ... ooops, ambiguous, isn't it? If the RAM-Disk-driver would be a Thing, then, when the new RAM-Disk-Driver is loaded, the old RAM-Disk-Driver will be removed from the system, the whole memory used by it will be released and the jobs with open channels to this RAM-Disk will be removed. What else can be done? There is no half-way tidy-up.

Things can also be system utilities or system extension. The Menu extension is a very good example. If you replace the Menu extension by a more recent version, then the old Menu extension goes away and is fully replaced by the new one. Try it: start a QD, try loading a file but when the file-select window appears (which is not part of QD, it is part of the Menu extension and QD is a user of the Menu Extension at this moment), then switch back to BASIC and LRESPR MENU_rext once again. First, the old Menu extension is removed (and therefore QD will be removed as well - remember, all users are zapped when a Thing is zapped) and then a new Menu extension is installed.

QD (and other programs) only use utility Things at the time they really use them - this means you can load a new Menu extension without jobs being removed as long as no pulldown windows contained in the Menu extension are open in other jobs. It would be possible (and maybe a little bit faster) for a job to register itself as a user of the Menu extension (you know, it then gets the address of the extension) and then remain being a user and call the extension every time it needs it. This will save a few use and free calls, and it is guaranteed to work too (the address CANNOT change: replacing the Menu extension, which would lead to a change of this address would not do any harm because at this time QD would not exist anymore - remember, users of the old Thing are gone...).

I hope you are not too confused by now - once again think of the Thing system as an identification and registration system.

There is no limit on what Things can be. It seems logical that if we can have extension Things, we can have executable Things as well (just like files, you see the analogy all the time?). It is possible to have many input-channels open to the same file, so why not have many users who use the same program code which exists only once in memory? This is the concept of executable Things. The program code, which is the same all the time no matter if zero, one, or any number of the same jobs use it, needs to be in memory only one time. All QPAC2 programs are executable Things, and you can execute as many as you like until you run out of memory. Now you will definitely agree that users have to go away if the Thing they are using is removed - how can a program execute when its program code is gone? You see, the concept does not sound too drastic, does it?

The third general Thing type is the "data" Thing. Once again it is entirely up to the creator of the Thing what kind of data is stored in here, how it is organised and used. I can't repeat if often enough: the Thing system is the tool to help a job finding the Thing and registering for it.

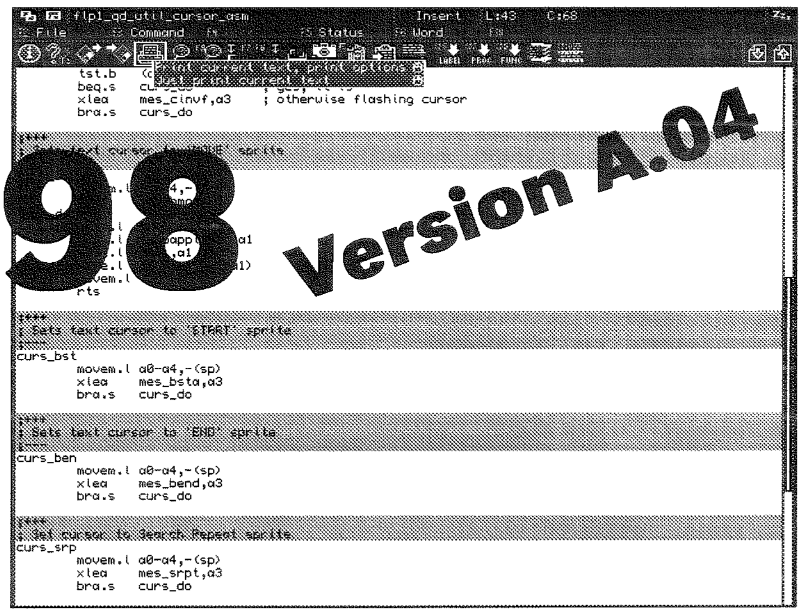
I think that's enough for this time. You may like to read this article a second time. If this was too difficult, write and tell me what you have not understood. In the next part, we will have a look at the Things in your system (provided you have loaded QPAC2) and one of the various ways to create Things yourself.

■

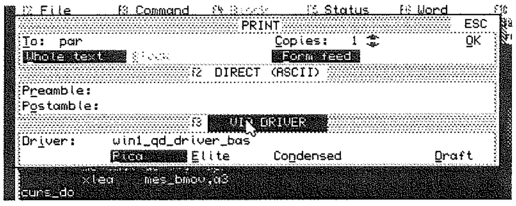
JOCHEN MERZ SOFTWARE

Im stillen Winkel 12 • 47169 Duisburg • Germany
☎ 0203-502011 (Fax 0203-502012 Mailbox 0203-502013 & 502014)
http://www.j-m-s.com/smsq/ email: smsq@j-m-s.com

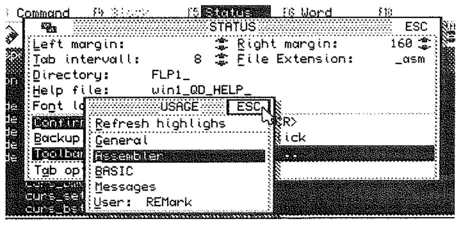
QD 98 Version A.04


```
flp1_qd_util_cursor_asm                    Insert  L:43   C:68
        tst.b   (c
        beq.s   cu
        xlea    mes_cinvf,a3    ; otherwise flashing cursor
        bra.s   curs_do

;+++
; Sets text cursor to 'MOVE' sprite

        rts

;+++
; Sets text cursor to 'START' sprite
;---
curs_bst
        movem.l a0-a4,-(sp)
        xlea    mes_bsta,a3
        bra.s   curs_do

;+++
; Sets text cursor to 'END' sprite
;---
curs_ben
        movem.l a0-a4,-(sp)
        xlea    mes_bend,a3
        bra.s   curs_do

;+++
; Set cursor to Search Repeat sprite
curs_srp
        movem.l a0-a4,-(sp)
        xlea    mes_srpt,a3
        bra.s   curs_do
```

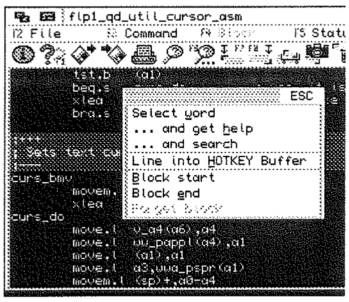
This ad does not give a list of all the features of our excellent Editor QD98, it will mainly list the new features which were introduced from the previous version QD Version 9 to the current one.
Here are its main features: "Hints" are given on menu items and the toolbar. Quite useful, because we have added many extra items and also added some additional features on "DO"ing some existing icons. Of course, this feature can be turned off.
The toolbar can be turned off and on while QD is running. A number of QL users asked me to do this because they gain two extra lines - it is done now!
Lines can be ordered, you can specify the start column in the text which is used as the sort criteria.



The GOTO label/proc/fn are now better accessible. We have also introduced a GOTO User-defined list. This means, you define a string for which QD looks and all the lines containing this text are put into the GOTO list (in fact, only the bit of the line after the string). This is very useful to create lists of all sorts, just use REMarks or other language-dependent combinations (C...)
We have added scroll arrows next to every numerical value entry in all menus - you will find this very useful if you use the mouse quite a lot.
PRINTing has been completely redone. You can print as before, or via driver. You can program the driver yourself, because it can be a BASIC filter. BASIC and Assembler examples are on the disk.



Line highlights are now possible and VERY, VERY useful. You tell QD what kind of highlights are required and it will highlight these lines in red paper. It can, for example, highlight all assembler comments (; or *), all BASIC Functions and Procedures (which is a kind of visual "folding"), or any lines with ">" in the first three characters (useful for message editing, PBOX etc.) and, of course, user-defined string. This will highlight all lines containing the string you gave - quite a neat way of doing a search, if you think about it.
"Context" menu which can be popped up with the right mouse button has also been introduced. This is much easier to handle than previous cursor-dependent functions.
These are the major changes, plus various minor ones, of course. You will get a new manual, not just additional pages.


```
flp1_qd_util_cursor_asm
        tst.b   (a1)
        beq.s
        xlea
        bra.s
;+++
; Sets text cu
curs_bmv
        movem.
        xlea
curs_do
        move.l  v_a4(a6),a4
        move.l  wu_pappl(a4),a1
        move.l  (a1),a1
        move.l  a3,wwa_pspr(a1)
        movem.l (sp)+,a0-a4
```

The upgrade price is DM 39,90 for owners of QD9, and DM 49,90 for owners of older versions. Please return master QD disk for upgrade. A new QD98 still costs only DM 125,-

TERMS OF PAYMENT

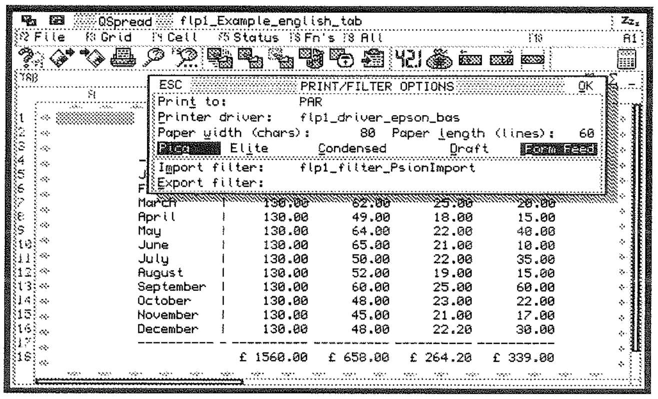
Postage and package **[Germany]** DM 8,99 (if total value of goods is up to DM 50,- then only DM 5,99). **[Europe]** DM 14,50 (if total value of goods is up to DM 50,– then only DM 9,50). **[Overseas]** between DM 14,50 (1 item) and DM 35,– (maximum). All prices incl. 15% V.A.T. (can be deducted for orders from non–EEC–countries). E&OE. Cheques in DM, £'s, Eurocheques and Credit Cards accepted.

JOCHEN MERZ SOFTWARE

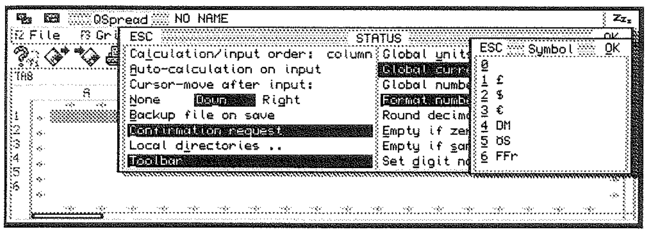
NEW!

QSpread99

It is done! A new major release of QSpread is ready!
We have added many features:
QD-style toolbar with many additional functions.
QD-style menus, shortcuts and keystrokes.
SHIFT TAB gives very long formula entry line.
ALT TAB moves pointer outside grid - if you don't have a mouse
ESC puts QSpread to sleep.
QDs printer driver filters can be used for QSpread as well.
More intelligent confirmation request settings.
All currency symbols are configurable now, including the EURO.
CNTNUM() just counts numerical fields in a range.
Multiple "—" and "====" are automatically treated as text.
Some special single characters like "|" are treated as text.
Automatic cursor-move after edit configurable, can also be turned off.
New DATE macro function which inserts current date.

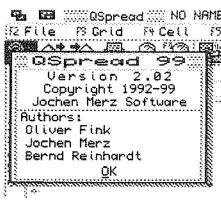


```
QSpread  flp1_Example_english_tab                              Zzz
F2 File  F3 Grid  F4 Cell  F5 Status  F6 Fn's  F8 All   F10     A1
TAB
      ESC          PRINT/FILTER OPTIONS                       OK
      Print to:            PAR
      Printer driver:      flp1_driver_epson_bas
      Paper width (chars):      80  Paper length (lines):    60
      Pica      Elite      Condensed      Draft      Form Feed
      Import filter:       flp1_filter_PsionImport
      Export filter:
 7    March     |   130.00     62.00     25.00     20.00
 8    April     |   130.00     49.00     18.00     15.00
 9    May       |   130.00     64.00     22.00     40.00
10    June      |   130.00     65.00     21.00     10.00
11    July      |   130.00     50.00     22.00     35.00
12    August    |   130.00     52.00     19.00     15.00
13    September |   130.00     60.00     25.00     60.00
14    October   |   130.00     48.00     23.00     22.00
15    November  |   130.00     45.00     21.00     17.00
16    December  |   130.00     48.00     22.20     30.00
17    ---------- - --------- --------- --------- ---------
18               £ 1560.00  £ 658.00  £ 264.20  £ 339.00
```



```
QSpread  NO NAME                                              Zzz
F2 File  F3 Gr  ESC                STATUS                      OK
                Calculation/input order:  column  Global units   ESC  Symbol  OK
                Auto-calculation on input         Global curr    0
                Cursor-move after input:          Global numbe   1 £
                None     Down     Right           Format numbe   2 $
                Backup file on save               Round decima   3 €
                Confirmation request              Empty if zer   4 DM
                Local directories ..              Empty if sam   5 öS
                Toolbar                           Set digit no   6 FFr
```

DATE format is configurable, it can be US-format or 4-digit year.
Default DATE separator character configurable.



```
QSpread  NO NAME
F2 File  F3 Grid  F4 Cell  F5 Sta
     QSpread 99
    Version 2.02
  Copyright 1992-99
 Jochen Merz Software
Authors:
 Oliver Fink
 Jochen Merz
 Bernd Reinhardt
         OK
```

In addition, lots of long-standing bugs are fixed, e.g.:
Window can be resized after QSpread was put to sleep.
DATE$ shows year-2k-dates correctly.
Double "." and other problems in small numbers fixed. Lots of bug fixes in the formula parser done.

These are the major changes, plus various minor ones, of course. You will get a new manual, not just additional pages.

The upgrade price is DM 39,90. Please return master QSpread disk for upgrade.
A new QSpread99 is price-reduced to only DM 149,-

CueShell

A quick reminder: you can get QPC together with CueShell by adding **DM 40,-** to the price of QPC. Even if you already bought QPC, you can add CueShell now if you like. Still only **DM 40,-**! Please return the original QPC master disk for an upgrade. If you do not own QPC and you like to purchase CueShell: **DM 89,-**

TERMS OF PAYMENT

Postage and package **[Germany]** DM 8,99 (if total value of goods is up to DM 50,- then only DM 5,99). **[Europe]** DM 14,50 (if total value of goods is up to DM 50,– then only DM 9,50). **[Overseas]** between DM 14,50 (1 item) and DM 35,– (maximum). All prices incl. 15% V.A.T. (can be deducted for orders from non-EEC-countries). E&OE. Cheques in DM, £'s, Eurocheques and Credit Cards accepted.

EUROCARD. VISA MasterCard Cards

My Boot Program

Wolfgang Lenerz

It seems that there is a certain interest in the way different people boot up their machines, hence here are a few words about my own Boot Program. I don't pretend that this is in any way particularly intriguing, but perhaps an aspect or two might be interesting. The program is, of course, specially tailored for me, so I don't know whether it can be of any use to YOU, but here goes. Please note that my Boot Program is an evolved program - I add to if from time to time whenever my needs change. This means that there are probably some bits in there that, if written from scratch, could certainly be improved upon. Other bits are probably obsolete. However, I firmly believe in the maxim : "if it works, don't change it".

As you will see, I make quite heavy use of the "dev" device, the virtual device. A few issues ago, somebody wrote in QL Today that he really couldn't see the use of such a virtual device. I personally couldn't live without it! (Just a little reminder for those that don't use this device: dev is a virtual device, which refers to an actual existing device. For example, one could set up "dev1_" to be "win1_". Then, typing something like "DIR dev1_" would actually give a directory of win1_. One can also set up "dev1_" to be something more complicated, like "win1_quill_" so that "DIR dev1_" would list the files in the "win1_quill_" subdirectory. This can come in quite handy with the old Psion programs.)

To explain why I really need such a virtual device, first a few words are needed on my setup, or rather my various setups as I have more than one. For various reasons, which I really needn't go into here (else I'd bore you to death), I have several QL-type machines: two PCs running QXLs, a laptop running QPC, an Atari Mega ST and, yes, an old black box itself together with an (original) GoldCard and 2.88 MB drives. I hope this doesn't seem I'm bragging. since, actually, I had a use for every single one of these machines: One of the QXLs is used at the office. Another is used at home, mainly to continue working on whatever I was doing when I got home. The laptop of course comes with me on any travel. The ST had long served me as my main computer until the QXL came along. The QL itself was used mainly for software libraries: at some time, I was a Quanta sublibrarian, and also the librarian for our french club, QLCF. Today, the St and QL are semi-retired, even though I still use them for testing purposes.

One problem with such a multimachine setup is that of consistency: I would really like to be able to switch from one machine to the other without really knowing (or needing to know) which machine I'm currently working on. This used to be a problem, since the PCs under QXL would run SMSQ, QPC works with SMSQ/E, the QL had a JS ROM, the Atari something else again and so on. Each of these operating systems had their own little quirks. This is why I was so glad when SMSQ/E came out - for there now is ONE AND THE SAME operating system for all of my machines - all of them thus now have the same capabilities. Sure, some run faster than others, but this is really secondary.

OK, so now I had the same machine everywhere - but something was still amiss: Indeed, all machines, save the QL, have harddisk drives - but they are not all "win1_". For some reason, on the PCs, the "QXL.WIN" file is not on the C: drive (which would make it "win1_), but on different drives (actually and just to make everything really complicated, the file resides on drive D: on one machine, on drive E: on another and on F: on the third). This means that whilst on the Atari, all my files reside on win1_, on the PCs they are on win2_, win3_ or win4_. On the QL, of course, everything is in flp1_. Imagine the headache to remember where everything is. Also, if you configure a program on one machine and tell it that it's data lies, for example, in "win1_aff_data_", on the other machine it would have to be configured for "win2_aff_data_" etc. Ugh!

This, of course, is where the "dev" device comes in: I just need to have "dev1_" refer to the actual WIN drive (or flp1_ on the QL) that I am using, and then I always refer to dev1_: the program thus can be configured only once to look for its data on "dev1_aff_data_", and it needn't care what exactly dev1_ is.

There remains one single little problem: how does dev1_ get configured? I'm of course much too lazy to want to type this in each time I use a machine (I mean, I'd have to remember which machine used what drive - unthinkable), and so the QL does it for me. This is the 'find_mydrive$' function, which is called early on in the program. The function lies at lines 3150 and following. As you can see, it first sets DATA_USE to an empty string (of which more later). It then tries, in a loop, to open a directory on win1_ to win8_, with the FOP_DIR command. As soon as this is successful, it knows that there is a valid hard disk, which necessarily is the first QL hard disk in the system, so it returns this and dev is later set to this. This is also why DATA_USE is set to an empty string. Indeed, FOP_DIR function will try to open the directory with the given name (eg. win1_ on the first go). If this isn't successful, it will attempt to open a directory with the given name and the DATA_USE default added to the front of the name (e.g., if DAtA_USE were set to flp1_, it would try to open flp1_win1_). Then, if a disk were in flp1_, the FOP_DIR function would return successfully (!) and the end result would be wrong. The end result is, anyway, that the computer sets "dev1_" to the correct hard drive and even to flp1_ on the QL.

Right, now let's go on to the Boot Program. First of all, I set a few variables in the "constants" procedure. Most of these will be explained later on. The "ws" procedure is then called, it sets up a totally black screen, and then proceeds to print a pretentious little message more or less centered in the screen. To do this, it uses the xlim and ylim variables, which, in the "constants" procedure, were set to the X and Y screen sizes - these vary from machine to machine, the QL of course being limited to the usual 512*256 size. On the PCs, I use 800*600, the Atari has yet another screen size. The "ws" procedure works OK whatever the screen size (provided it is not smaller than a standard QL screen).

I then reset the windows and print a small message (lines 150-190) - note how window#0 is made practically non-existent. This is done on purpose. Indeed, later on I load a large set of extensions. Setting window #0 to this size stops the extensions from printing their initialisation messages all over the screen, which I find to be aesthetically unpleasant. I know that the extensions are being loaded and don't need to be reminded.

After that, I find the drive to which I'll set my dev1_ device, through the use of the "find_mydrive$" function discussed above. If this is unsuccessful, the Boot Program complains and stops right there (lines 210-220).

In lines 240 to 380, I load all of the extensions I need. Here again, a few words might be beneficiary. This section was added to my boot file when the QXL came about. The speed at which the QXL reads/writes from hard disk can be described as dismal. I thus thought that it would be better to have all of the extensions in one single big file rather than in several small files (I load 35 extension files in all). So, with the help of a small procedure which we'll discuss later, I made a single file of all my extensions - they are just put one at the end of the other. The result file is called "dev1_booty_boot_rext". Its current size is denoted by the "tot_len" variable, initialised in the "constants" procedure and is currently 303152 bytes. This is loaded with a single LBYTES call. The individual extensions are then called through a loop which reads the name of the extension file and the length thereof and thus can call the exact place in this file where each extension is located. The name is printed out (this was inserted here for debugging purposes, so that, if the boot process suddenly stopped, I could know what extension causes the trouble since some of them are written by me, the risk is always there...).

The names of the extension (files) are held in the DATA statements at lines 2510 and following.

We are searching for programmers for the development of customer-specific applications under SMSQ

One of the main projects will be machine tool data recording via serial ports (not in assembler, mainly in BASIC).
If you are interested contact us at our office address (see right)
or send email to
vogel-isernhagen@t-online.de

VOGEL ANTRIEBS- UND STEUERUNGSTECHNIK

- **Planung und Projektierung von Maschinensteuerungen**
- **Programmierung von Steuerungssystemen (SPS, etc.)**
- **Anwendungsspezifische Antriebslösungen**
- **Inbetriebnahme, Service, Instandhaltung**

CONTROL TECHNIQUES Servicestützpunkt

Joachim Vogel • Hannoversche Str. 22 • 30916 Isernhagen
Tel. 05 11 / 6 10 57 - 11 • Fax 05 11 / 6 10 57 - 23

The only thing to watch is to make sure that all files to be loaded in such a manner are of even length. If not, the next file would be loaded at an uneven address, which will cause all sorts of problems!

In lines 380-430, after resetting window#0 to something suitable, I set the serial ports to whatever value suits me. This would actually give some problems on the QL, but I never really used that for serial communications anyway.

The "use_defaults" procedure called in line 440 sets the different devices for dev1_, the prog, data and dest defaults etc. It also sets the SUB device (yet another virtual device) to dev1_. This is purely a legacy command, I could reconfigure everything for dev1_.

After printing out the time (of which more later), I load a special keyboard layout which, on a PC keyboard, gives me a CTRL-C on F11 and a CTRL-SPACE on F12 (lines 470-510). The PUT_KBD keyword is part of the "clavier_bin" extension. "Clavier" is also available on several BBSes and allows you to remap your keys under QXL or QPC. Lines 520-560 load my accounting package. As you can see, whether or not it is loaded depends on the "do_all%" variable. If I take the REMark out of line 120, then do_all% is 0 on some machines with less than 5 MB RAM and thus some programs are not loaded.

Just after that, in lines 580 to 680, I load a certain number of programs: the QPAC 1 calculator and three programs of my own design. These are pretty straightforward HOTKEY calls. Note that I could replace the HOT_RES, HOT_REMV and HOT_WAKE combination with a single HOT_RES1, but inertia just keeps this as is...

After loading another program necessary for managing my office work (lines 690-720), I then set up some of the QPAC2

```
100 PRINT "Booting..."
110 do_all%=1
120 REMark IF FREE_MEM<5E6:do_all%=0
130 constants
140 ws
150 WINDOW 512,12,0+xx,244
160 WINDOW#0,0,0,XLIM,YLIM
170 BORDER 1,4
180 PRINT ' Date/Time: ';DATE$;'  -   Searching for drive...';
190 OVER 0
200 :
210 mydrive$=find_mydrive$
220 IF mydrive$="":PRINT "CAN'T FIND MY DRIVE!!!":STOP
230 :
240 CLS
250 PRINT ' Date/Time: ';DATE$;'  -   Loading extensions file now:';
260 a=RESPR(tot_len)
270 LBYTES mydrive$&"booty_boot_rext",a
280 :
290 RESTORE
300 REPeat lp%
310   READ a$,b
320   PRINT "calling ";a$;"...";
330   CALL a
340   a=a+b
350   PRINT "ok"
360   IF EOF():EXIT lp%
370 END REPeat lp%
380 :
390 WINDOW#0,100,100,0,50:INK#0,4
400 BAUD 57600
410 SER_BUFF 2,1024
420 SER_BUFF 1,1024
430 :
440 use_defaults
450 time
460 :
470 addr=RESPR(808)
480 LBYTES dev1_booty_kbd_kbd,addr
490 PUT_KBD addr
500 LANG_USE gb
510 :
520 IF do_all%
530   CLS: PRINT" Loading Compta"
540   EX dev2$&"compta_obj"
550 END IF
560 :
570 CLS:PRINT" Loading Calculator"
580 ERT HOT_RES('C',dev2$&'calculator')
590 ERT HOT_REMV('C')
600 ERT HOT_WAKE('C','calculatrice')
610 CLS:PRINT" Loading memlib"
620 ERT HOT_RES(CHR$(9),dev2$&'MEMLIB')
630 CLS:PRINT" Loading date"
640 ERT HOT_RES(CHR$(27),dev2$&"date_bin")
650 CLS:PRINT" Loading sysdef2"
660 ERT HOT_RES('D';dev2$&"sysdef2")
670 ERT HOT_REMV('D'):ERT HOT_WAKE('D','Sysdef2')
680 :
690 IF do_all%
700   CLS:PRINT" Loading Affaire"
710   EX dev2$&"affaire_obj"
720 END IF
730 :
740 CLS:PRINT" Loading QPAC2"
750 qpac2
760 :
770 CLS:PRINT" Loading Linker"
780 ERT HOT_RES('l',dev2$&'Linker'):ERT HOT_REMV('l')
790 CLS:PRINT" Loading Make"
800 ERT HOT_RES('l',dev2$&'Make'):ERT HOT_REMV('l')
810 CLS:PRINT" Loading QMac"
```

Professional & Graphical Software

ProWesS

ProWesS is a new user environment for the QL. ProWesS is short for "PROGS Window Manager", but it is much more than that. Apart from a new window manager, it contains all the system extensions from PROGS, and is essential if you want to run programs which need these extensions.

The ProWesS reader is a major part of the package. It is a hypertext document browser. This means that text files which include formatting commands (including pictures) and possibly links to other files can be displayed and read in this program. This is used in ProWesS to read (and possibly print) the manuals, and display the help files. The hypertext documents which are used by the ProWesS reader are in HTML format, the format which is popular on Internet to display World Wide Web pages.

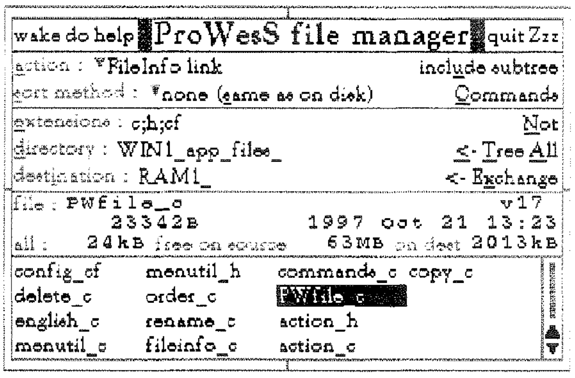
Another important aspect of ProWesS is the possibility to allow programs to automatically install themselves on your system, and to be able to run them without resetting the system. This means that, when you get a new program, all you have to do is insert the disk and indicate "start the program in flp1_", a menu option in the "utilities" button. To install a program, you indicate "install software", and the software can be added to your system. This way, you don't need to know how to write a boot file to use the multi-tasking capabilities of your computer.

ProWesS includes many programming libraries. These include syslib, an interface to the operating system, PROforma, a vector graphics system, allowing rendering both on screen and on paper (via a printer driver). The DATAdesign engine is also part of ProWesS. It is a relational database system with a bonus, as you don't even need a key field. You get a powerful record at a time data manipulation extension to the language you already use. Of course it also includes ProWesS itself, the new resolution independent window manager.

New ProWesS application

a powerful and very user

friendly file manager



```
wake do help ProWesS file manager quit Zzz
action : ▼FileInfo link                include subtree
sort method : ▼none (same as on disk)         Commands
extensions : c;h;cf                                Not
directory : WIN1_app_files_                 <- Tree All
destination : RAM1_                         <- Exchange
file : PWfile_c                                    v17
         23342B              1997 oct 21 13:23
all :    24kB free on source    63MB on dest 2013kB
config_cf     menutil_h     commands_c  copy_c
delete_c      order_c       PWfile_c
english_c     rename_c      action_h
menutil_c     fileinfo_c    action_c
```

PFlist

Easy to use program to create listings on any printer (especially inkjet and laser). This ProWesS application allows you to indicate the files which have to be printed. Each column contains a footer which can include the filename and filedate. The listings always allow perforation. PFlist can create your listings in two columns and in landscape (or both).

fsearch

File search utility with many useful options, like the choice to search only files with a certain extension, and whether or not the directory tree has to be scanned. All occurences of the searchstring will be displayed with line number or offset. You can also use special matching features, like case dependent, matching a space with a stretch of whitespace, and searching for a word dilimited string.

font-utils

manage your font collection. You can preview fonts on screen, see what characters exist in a font and convert Adobe Type 1 and similar fonts for use in ProWesS.

LINEdesign

Create artistic drawings, technical drawings, process bitmaps (even scale and rotate them!), and any kind of vector drawings. You can use grpahics objects to create the most fabulous drawings ever seen. Because LINEdesign is a vector drawing program, any part of the picture can be moved, scaled, rotated, slanted without any loss of precision or resolution. In LINEdesign, pictures are device independant, meaning that the printout will be the same on any printer (e.g. same size and position).

LINEdesign is good at handling text. You can easily put titles and full paragraphs on the page. All the fonts can be displayed at any size, rotation, etc. All the fonts which are available to ProWesS can be used in LINEdesign.

LINEdesign is a drawing program, but it can also be used by people who are not good at drawing. LINEdesign is a great program for making leaflets, posters, and any kind of printed work. Lots of clipart and extra fonts are available from public domain libraries and BBS's. You can even import Adobe Illustrator files.

DATAdesign

Never before has it been so easy to create, fill in and maintain your personal databases. To start a new file, just type the names of the fields. To add or delete a field, no problem, just do it. To change the name of a field, just indicate it. You can choose which fields are displayed and also which records. You can have a hidden comment for each record, look at the file in tabulated form and transfer data to the scrap or hotkey buffer. Files can be memory based (for speed) or disk based (for safety).

new address !!

Dr. Fr. Hemerijckxlaan 13 /1
2650 Edegem
Belgium

tel : +32 (0)3/ 457 84 88
fax : +32 (0)3/ 458 62 07
email : joachim@club.innet.be
www : http://www.club.innet.be/~year2827

ProWesS - BEF 2400	DATAdesign - BEF 1200	PWfile - BEF 900	PFlist - BEF 600
	LINEdesign - BEF 1200	fontutils - BEF 1200	fsearch - BEF 600

Payment terms :

You have to run ProWesS to make LINEdesign, DATAdesign, fsearch, fontutils and PFlist work (even though DATAdesign uses wman).

All our software is normally supplied on high density (HD) disks. However they can be obtained on double density (DD) disks at an extra costs of BEF 100. To use ProWesS and any of our other packages, you need a system with at least 2MB of memory. You should have a harddisk although a two disk system will also work. The use of SMSQ/E is strongly recommended for optimal use of ProWesS.

If you are VAT registered (specify registration number) or live outside the EEC, the amount to be paid is the total (including postage) divided by 1.21 (no need to pay too much).

Payment can be done by EuroCheque in BEF, or by VISA, EuroCard or MasterCard. Credit card orders can be handled by phone. For credit card, please specify name of card owner, card number and expiry date.

Postage : Costs of postage and packaging have to be added. You can choose the quality. Rate depends on no of programs.

	priority mail			*ordinary mail*		
copies	Belgium	Europe	World	Belgium	Europe	World
one	100	200	240	100	120	145
two	135	340	420	135	190	230
3 or 4	160	560	770	160	310	395
5 to 8	185	870	1250	185	550	705
more	295	1130	1610	295	800	1030

All prices are in BEF, including 21% VAT programs. I generally don't need all of them, but I still set them up. This is done via two procedures, "qpac2" (lines 1470 to 1590) and "qp2" (lines 1610 to 1680). The "qpac2" procedure just sets up the hotkeys. The "qpac2" procedure calls the "qp" procedure, which sets up the buttons in the button frame and starts the hotkey job. I don't put all of the Qpac2 jobs in the button frame, since all of my "professional" programs also place themselves into the button frame, and I just don't want to have too many buttons flying around - I find that they take away too much screen real estate.

After that, I load some programs needed for assembling assembler files. Note how I load them with HOT_RES, and then I throw the corresponding Hotkey away: the only purpose of this is to load the programs as executable things, where they stay in memory all the time. They are then called directly from memory whenever needed (lines 770-860).

I also load my invoicing program (lines 870-920) after which I set up a certain number of often used hotkeys, mostly used for putting my name at the bottom of letters... (lines (930 - 990). I then set up Hotkeys for QD (lines 1000 to 1020). The QD file itself is loaded as one of the extensions. I set up two hotkeys for QD, one for picking/waking an existing QD (line 1010), and another for getting a new QD (line 1020).

In line 1060, I load a small program called "blank". When executed by pressing the corresponding Hotkey ('&'), this blanks out the entire screen and waits until a password is typed in, for the obvious reason to stop people from reading whatever might be on the screen. I'm probably a bit paranoid... This little program also stops the machine from multitasking, so that you can't get around this protection by hitting CTRL-C.

```
820 ERT HOT_RES('1',dev2$&'Mac'):ERT HOT_REMV('1')
830 CLS:PRINT" Loading Qlib"
840 ERT HOT_RES('1','dev1_qlib_qlib_obj'):ERT HOT_REMV('1')
850 ERT HOT_WAKE ('1','Qlib_3.35')
860 :
870 IF do_all%
880   ERT HOT_WAKE('a','Affaire')
890   CLS:PRINT" Loading Facture"
900   EX dev2$&"facture_obj"
910 END IF
920 :
930 CLS:PRINT "Setting up hotkeys"
940 tab$=CHR$(9)&CHR$(9)&CHR$(9)&CHR$(9)
950 cr$=CHR$(10)&CHR$(10)&CHR$(10)
960 ERT HOT_WAKE('c','Compta')
970 ERT HOT_KEY('W',tab$&"W.
Lenerz"&CHR$(10)&CHR$(10)&CHR$(10)&tab$&"Avocat à la Cour")
980 ERT HOT_KEY('w',tab$&"W. Lenerz")
990 :
1000 IF do_all%:ERT HOT_WAKE('T','Xchange')
1010 ERT HOT_WAKE('q','QD')
1020 ERT HOT_THING('Q','QD')
1030 :
1040 IF do_all%:ERT HOT_WAKE ('g','Facture')
1050 :
1060 ERT HOT_RES('&',dev2$ &"blank")
1070 :
1080 IF do_all%
1090   CLS:PRINT" Loading print_models"
1100   EX printmodels_obj
1110   CLS:PRINT" Loading xchange"
1120   EX xchange
1130   ERT HOT_KEY('b',CHR$(240)&'tdev1_quill_startup_tsl'&CHR$(10))
1140 END IF
1150 :
1160 ERT HOT_KEY('‹','cs'&CHR$(10)&'di2 *'&CHR$(10)&'ct'&CHR$(10))
1170 ERT HOT_KEY('v',"Veuillez agrâer, Cher Monsieur, l'expression de
mes sentiments distinguâs."&cr$&tab$&"W. Lenerz"&cr&tab$&"Avocat à la
Cour")
1180 ERT HOT_KEY('V',"Veuillez agrâer, ChÉre Madame, l'expression de
mes sentiments distinguâs."&cr$&tab$&"W. Lenerz"&cr$&tab$&"Avocat à
la Cour")
1190 IMP_NAME "par":PRT_USE "par"
1200 SECURE 3
1210 :
1220 IF do_all%
1230   REPeat lp%
1240     IF BTN_APX('Compta') AND BTN_APX('Affaire') AND
BTN_APX('Facture'):EXIT lp%
1250     PAUSE 10        : REMark wait till all finished
1260   END REPeat lp%
1270   ERT PICK (11,10) : REMark pick Xchange
1280   PAUSE 60
1290   HOT_DO 'b'
1300   ERT HOT_KEY('b',CHR$(236)):PAUSE 40:HOT_DO 'b'
1310   PAUSE 20
1320   HOT_DO sleep$
1330   ERT HOT_REMV('b')
1340 END IF
1350 :
1360 ERT HOT_PICK('B','')
1370 ERT HOT_WAKE('b','Sbasic')
1380 POKE_L !!$8C,HEX('00100001')
1390 PAUSE 40
1400 HOT_DO 'B'
1410 WSET_DEF 256,34,XLIM-256,YLIM-34,256,YLIM-bto-34,XLIM-256,bto,
XLIM-256,YLIM-bto,0,bto
1420 OUTLN#0,XLIM,YLIM-bto,0,bto
1430 WM:PAUSE 1
1440 HOT_DO sleep$:PAUSE 4
1450 EXEP 'sbasic';'lrun dev1_booty_sbasic'&CHR$(10)
1460 :
```

```
1470 DEFine PROCedure qpac2
1480 REMark set up hotkeys for qpac2
1490   ERT HOT_WAKE('f','files';'\sfn\ddev1_')
1500   ERT HOT_WAKE('h','hotkeys')
1510   ERT HOT_WAKE ('t','things')
1520   ERT HOT_WAKE ('k','channels')
1530   ERT HOT_WAKE( 'r','rjob')
1540   ERT HOT_THING(sleep$,'Button_Sleep')
1550   ERT HOT_THING('.','Button_Pick')
1560   ERT HOT_WAKE('e','exec')
1570   ERT HOT_WAKE ('j','jobs')
1580   qp2
1590 END DEFine qpac2
1600 :
1610 DEFine PROCedure qp2
1620 REMark set up buttons for qpac2
1630   HOT_GO
1640   BT_SLEEP 'channels'
1650   BT_SLEEP 'rjob'
1660   BT_SLEEP 'exec'
1670   BT_SLEEP 'files';'\sfn\ddev1_'
1680 END DEFine qp2
1690 :
1700 DEFine PROCedure no_hotk
1710 REMark delete all hotkeys!
1720 LOCal hname$,key$,lp%
1730   FOR lp%=33 TO 255
1740     key$=CHR$(n)
1750     hname$=HOT_NAME$(key$)
1760     IF hname$ <>'':ERT HOT_REMV(key$)
1770   END FOR lp%
1780   ERT HOT_REMV('B')
1790 END DEFine no_hotk
1800 :
1810 DEFine PROCedure make_file
1820 REMark make my boot_rext file
1830 LOCal add,a,b,lp%
1840   constants:RESTORE
1850   add=ALCHP(tot_len)
1860   a=add
1870   REPeat lp%
1880     READ a$,b
1890     LBYTES mydrive$&"booty_"&a$,a
1900     a=a+b
1910     IF EOF():EXIT lp%
1920   END REPeat lp%
1930   SBYTES_O mydrive$&"booty_boot_rext",add,tot_len
1940   sa
1950   RECHP add
1960 END DEFine make_file
1970 :
1980 DEFine PROCedure sa
1990   SAVE_O mydrive$&"booty_boot_bas"
2000   QSAVE_O mydrive$&"booty_boot"
2010 END DEFine sa
2020 :
2030 DEFine PROCedure use_defaults
2040 REMark set up default dirs & devices
2050   DEV_USE 1,mydrive$
2060   DATA_USE "dev1_basic_"
2070   DEST_USE "dev1_"
2080   PROG_USE "dev1_progs_"
2090   dev$="dev1_booty_"
2100   dev2$='dev1_progs_'
2110   SUB_DRV mydrive$(1 TO 4)
2120 END DEFine
2130 :
2140 DEFine PROCedure verif
2150 REMark check file length consistency
2160 LOCal lp%,a,b,c,a$,chan%,erfl%
2170   RESTORE :constants
2180   c=0:erfl%=0
```

The next step (lines 1080 - 1140) is, optionally, to load Xchange, for which I then set up a small hotkey, which will later be inserted into its keyboard queue, and causes Xchange to run a small TSL program. This in turn will set up a certain number of preferences in Xchange, call up Quill and automatically load a default document which, without much originality, I call default_doc. That way, Quill is all set up and waiting exactly as I want it.

The next few lines (1160-1190) set up a certain number of other hotkeys, again for signing off letters and set a few special print buffers. In line 1200, I set up another security feature, much like the 'blank' program mentioned above. This sets up another small program, called 'secure' which, after a certain time of inactivity (here three minutes), also blanks the screen and waits till a password is typed in. In other words, if I don't move the mouse or type anything on the keyboard for three minutes, the screen goes black. Just another paranoid security feature, but useful when leaving the office for a few moments.

Lines 1220 to 1340 are, again, conditional. If they are performed, there first is a loop that waits until my office programs all have initialised themselves and are set up as buttons. This is achieved with the BTN_APX function. That is part of a pointer toolkit I have written. The function returns 1 if a job with the given "approximate" name is set up as a button. Once my jobs are set up as buttons, Xchange is picked and the programs waits a few heartbeats till Xchange is really there and then does the Hotkey mentioned above causing Xchange to load and execute the TSL program. After that it resets the hotkey to another string (just F2, to get rid of the help window in Quill) and also does that. Once this is done, Xchange is sent to sleep in the button frame by executing the Button_sleep program of Qpac2.

I then set up two hotkeys, one ('B') to call the main Sbasic, i.e. job 0, in line 1360 and the other to wake an SBasic daughter job. Remember, till now, this program runs in job 0. In Line 1380, I do some poking into the system variables, to speed up the rate at which the keyboard responds to keys typed in (and to speed up the repetition rate - that makes it much faster to zoom around a line with the cursor keys...).

In the last few lines, I set up some window sizes with the specially written "WSET_DEF" keyword, and set the outline for window#0 of job 0. Then I send job 0 to sleep behind a button (line 1440) and call up an Sbasic daughter job, which is then my main command window. This is done to avoid crashing job 0 whenever I experiment with loading some new extension I'm writing. This way, if I load an extension and try it, and it crashes the program, it only crashes an SBasic daughterjob... I tell this Sbasic to run another small program, which just sets its outline and windows to a size I find suitable. This is where the boot process ends, leaving me with a single row of buttons and a large Basic window.

The Boot Program also contains a few procedures which weren't mentioned yet, notably the "verif" (lines 2140 to 2410) and "make_file" (lines 1810 to 1960) procedures. The "make_file" procedure makes the large boot_rext file mentioned earlier. It just loads the extension files into memory one at the end of the other and then SBYTES the whole shebang as one single file. The "verif" procedure checks whether the file lengths of all of the extensions files are still correct. If not, it shows what file is the culprit and it also shows the correct total file length for the cumulated file, so that I can just change this in the "tot_len" variable.

Finally, the "time" procedure (lines 2790 to 2840) makes a date string in the format DD.MM.YYYY and sets up a hotkey that types in "Paris, le " and that date. It uses the "make_date$" and "make_all_months$" functions. These are some of my standard functions, used for handling dates. They may seem a bit cumbersome. The purpose is to obtain a string in the DD.MM.YYYY (day month year) format - the difficulty lies in the MM bit of it. Indeed, the standard QL DATE$ function returns the month as a three letter code (Jan, Feb etc...). How do you convert that into a number (01, 02 etc...)? Simple, you say: just make a string with all three letters of all months:

"JanFebMarAprMayJunJulAugSepOctNovDec" and find out, via the INSTR function, where in this string the month abbreviation lies. Then, with the formula res=(res+2)/3 (where res was the result returned by the INSTR function) you get the number of the month. This is what the "make_date$" function does. Pretty straightforward, except for one problem: If the user's language is not english, the month string above is not correct: "Feb" in english becomes "Fev" in french - and, when looking for "Fev" in the above month string, the INSTR function would return 0 since it's not in there... Thus, first of all, you must get the computer itself to generate the months string, which will be in the language used by the user. This is achieved with the "make_all_month$" function, which, in a loop, gives the DATE$ function some simulated parameters which successively and necessarily lie in all twelve months, thus getting the months from there...

```
2190   REPeat lp%
2200     READ a$,a
2210     chan%=FOPEN(mydrive$&"booty_"&a$)
2220     IF chan%<0:
2230        PRINT "error in ";a$;": ";chan%:erfl%=1
2240     ELSE
2250       b=FLEN(#chan%)
2260       CLOSE#chan%
2270       IF b<>a
2280          PRINT "error in ";a$;": true length= ";b
2290          erfl%=1
2300       END IF
2310       c=c+b
2320     END IF
2330     IF EOF():EXIT lp%
2340   END REPeat lp%
2350   IF (erfl%) OR (tot_len<>c)
2360     PRINT "Some errors here!"
2370     PRINT "True length of combined file should be ";c
2380   ELSE
2390     PRINT "all ok!"
2400   END IF
2410 END DEFine verif
2420 :
2430 DEFine PROCedure constants
2440   tot_files=35:tot_len=303152
2450   XLIM=SCR_XLIM:YLIM=SCR_YLIM
2460   xx=(XLIM-512)/2
2470   sleep$=CHR$(232)
2480   bto=28
2490 END DEFine constants
2500 :
2510 DATA "menu_rext",28216
2520 DATA "qmon_bin",12260,"jmon",15684,"qpac2",38430,"qptr_bin",9620
2530 DATA
```

Right, that's the end of my Boot Program. It used to contain a section for backing up a certain number of files. I have to take data security pretty seriously, since I keep accounting and other professional data on the QL. However, today my main machines all run on some kind of PC, either with QXL or the laptop with QPC. My QL hard disk is 70 MB long. To back it up, I simply copy the "QXL.WIN" file onto a ZIP cartridge. This takes about 2 minutes, and at the end of that, I have backed up my entire data!

From there I simply copy it onto the other machines, to spread it around as much as possible, to avoid any data loss. I even regularly give the ZIP cartridge to a friend for safe keeping (he doesn't have a QL or similar machine) just in case my house and office should burn down at the same time with all computers in them. See, I told you I was paranoid!

■

QUANTA

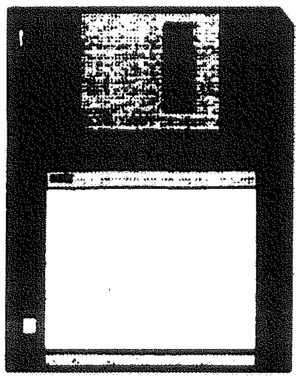
Independent QL Users Group

Worldwide Membership is by subscription
only, and offers the following benefits:
Monthly Newsletter - up to 40 pages
Massive Software Library - All Free !
Free Helpline and Workshops
Regional Sub-Groups. One near you?
Advice on Software and Hardware problems
Subscription just £14 for UK members
Overseas subscription £17

Barclaycard: Visa: Access: Mastercard

*** Now in our SIXTEENTH successful year ***

Further details from the Membership Secretary



Bill Newell
213 Manor Road
Benfleet
Essex
SS7 4JD
Tel. (01268) 754407

CueShell

A powerful Alternative to QPAC II

Rich Mellor takes a look at an established piece of software from highly respected QL software author Albin Hessler.

QPac II has been around for a long time now and although users have become accustomed to it, in many ways its operation can be long winded.

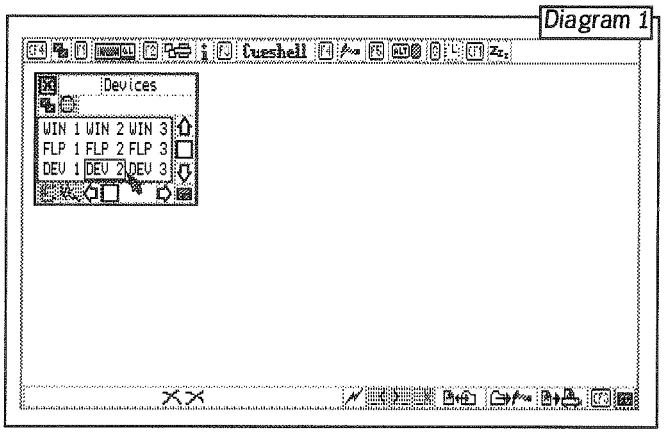
Although Cueshell replaces much of the function of QPac II, it is really intended to provide an alternative to the ways of doing things under Qpac II. Cueshell offers some features not available in Qpac II and vice versa.

Cueshell offers options which allow you to manage files; change mouse, keyboard and clock parameters; manage Jobs and manage Hotkeys.

The main use of Cueshell is as a desktop file manager. Many users will have used the Windows File Manager program, whereby you can select files and 'Drag and Drop' them to the required desination. Cueshell mimics this to some extent, whilst overcoming some of the problems inherent in Drag and Drop (for example, where you inadvertantly move a whole set of files to another directory).

When Cueshell is first started up, you will see a banner of the various options across the top of the screen (see Diagram 1). You are also presented with a window which contains a list of all recognised directory devices. Selecting one of these devices with the mouse or keyboard opens a further window which contains a directory listing of that device.

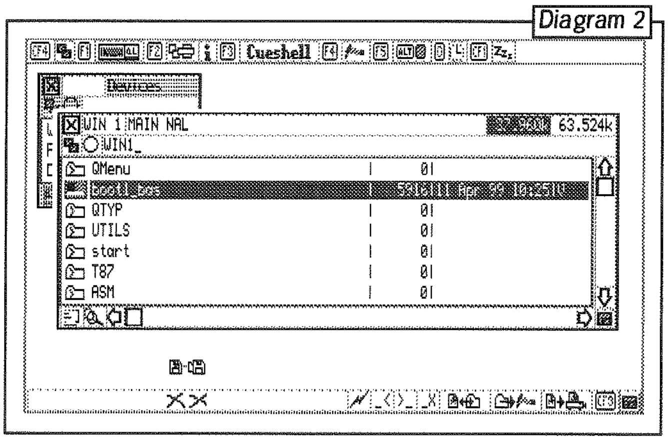
Diagram 1



You can select any file by HITting the area in front or behind the filename - the filename is then highlighted. HITting the actual filename just places that filename in the Hotkey Stuffer Buffer. The effect of DO depends whereabouts on the line which contains the filename, the pointer is placed.

If you DO the area to the left of the filename, then the mouse pointer changes shape to become a moving symbol. You simply move this pointer onto a destination directory (in any of the windows which Cueshell has opened on screen), or onto the two crosses at the bottom of the window (which will delete the file or sub-directory) - see diagram 2. If the destination directory is the same as the source, it is presumed that a rename is required.

Diagram 2



At all stages, you are asked to confirm the action before it is carried out (for example, when copying a file, you are asked whether you wish to Update the file or Backup the file).

One of the main advantages of Cueshell's copy and backup commands over QPac II is the fact that the directory structure is maintained. There is no need to re-create the sub-directories with a separate program when you restore your hard disk from backups anymore!!

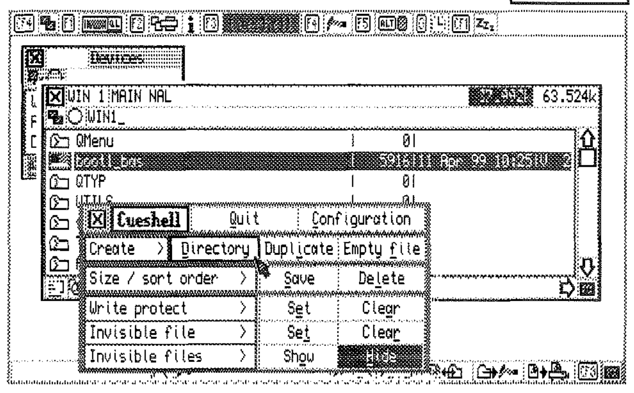
You can view a file by simply DOing the area to the right of the filename, or by selecting the filename and then HITting the magnifying glass icon. If you DO the actual filename, this will have one of two effects:

- If the file is the name of a sub-directory, another window is opened with the directory of that sub-directory.
- If the file is not a sub-directory, you are presented with the FileInfo II options (if available). Otherwise, if the file type of the program is recognised (eg. executable, basic extension), then Cueshell will try to run that file. If you keep the right hand mouse button or ‹ENTER› key depressed, you will be offered a menu with various options relating to how that file should be called (such as providing a Guardian window for an executable file).

You can sort filenames in each window by name, type and size (in any combination). The file size, version number and last date the file was altered (the update) is always displayed for each file. You will also notice the medium size and free space for each device. Unfortunately the amount of space taken up by a selection of files or a sub-directory is not readily available.

Another useful feature of Cueshell is the ability to set sme file attributes, such as making files read only, or invisible (see Diagram 3). Unfortunately, these attributes are not recognised by the QL operating system and therefore the files can still be seen and deleted from outside Cueshell (although I have been unable to test whether the System and Toolkit III file attribute system will also recognise these settings).

Diagram 3



It is also possible from within Cueshell to create a new directory, duplicate an existing file and also to create an empty file if you wish. Unlike QPac II, the new directory is not made the default data device.

Cueshell even allows you to print out a chosen file and use wild cards to some extent. You can add or remove an underscore to the current directory to change the wildcard provided by Toolkit II (similar to DDOWN and DUP) and also elect to show only those files which either have the same extension as the selected file or those which do not. This provides a great deal of flexibility.

Cueshell is however more than a file manager.

You can use it to set keyboard and mouse configurations (such as repeat delay and wake up speed) as well as changing the date and time of the clock.

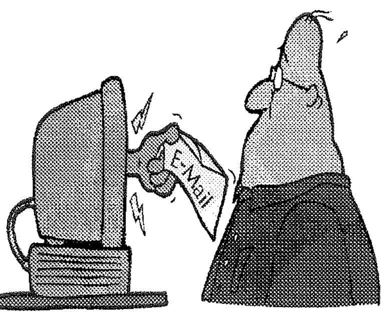
Cueshell will also provide a list of all current Jobs on the QL, allowing you to Pick, Remove and change the Priority of any Job. This is similar to the QPac II Pick, Rjob and Jobs Menus.

Although QPac II allows you to execute a hotkey, Cueshell has one main menu which provides you with much more control over hotkeys (and ALT-KEYs). You can execute them (the same as pressing them from the keyboard) as with QPac II, but you can also remove them, as well as switching them OFF temporarily and back on. This can be useful where you are using a program which clashes with certain of your standard hotkeys.

Although the effects of selecting HIT or DO on the various areas of the filenames takes some getting used to, Cueshell provides a very flexible approach to managing your system. I would thoroughly recommend this program to anyone who finds QPac II hard to get to grips with and are looking for something which is a little more intuitive.

Cueshell is available from both Jochen Merz Software and QBranch for the price of £30, but can be included as an extra module when bought with QPC for just £20. Oddly the Pointer Environment is not provided with the program - this should be included as a matter of course!! *[It is included now, Jochen]* ■

Letter-Box



Graham Bindon writes:

Your article in QL Today this month *[Printer Control Codes]* gives me hope of getting to grips with printer mysteries. I was grateful for your help before Christmas and thought it might be of interest to relate some of my adventures in printing.

Some advice and discussion on the telephone with interested folk helped me and after much pondering and with some reluctance I decided on an Epson Stylus 740; chiefly because I was assured it could work with ESC/P2 drivers obtainable from Roy Wood. Also I discovered that up-to-date drivers were available for my Acorn Archimedes.

The latter arrived before the former and included an entirely new Print Manager together with numerous files of various sorts. Loading the Ma-

nager presented no problems but there was some difficulty in choosing which printer definition to use because there wasn't one specifically for the 740. First I tried one for Stylus 800. This produced a page of absolutely beautifully printed nonsense. The next definition was for "StyCol 800". This worked extremely well.
And now came the really interesting bit - would the QL produce any intelligible print? In exchange for a string of digits supplied by me Roy sent a copy of Text87's Plus4 which I guessed would have a better than average chance of producing intelligible prints. The same uncertainty obtained as to which driver to use, Stylus_P87, Stylus_New_P87 or ESC/P2. The first two, which seenmed the obvious ones to use, appeared not to work very well. The last produced the goods.
I shall devote some more time to the Stylus drivers: it may be that in making the effort to learn the use of a new word processor I got something horribly wrong. Anyway, Plus4 and the ESC/P2 driver produce excellent prints. I made a list of all the typefaces using for each one the typeface named, but one learns to be abstemious in their use.
The results when using Xchange are less certain even after configuring the printer with the full preamble and post-amble codes. There remains a tendency to print part of the document as typed but to break off suddenly and print line after line of (- C cedilla rather decorative, but very boring reading. The only way round this is to print to a disk file and then

```
OPEN #n,file_name
OPEN #m,ser1
SPLF #n TO #m
```

This works - (almost always anyway) ***[I wonder if it might be worth asking a person knowledgeable in serial ports if your port settings in the Quill printer driver are correct - this sort of problem can sometimes be caused by serial port handshaking problems, for example. Can anyone offer any advice on this to Graham? - Editor]***
The other thing to do is to import a Quill document into Plus4 and print from there, but I feel that is a kind of surrender to irrational forces and should be resisted if at all possible.
The situation with me is that good prints can be obtained if I am using proprietary software; I still do not know how to use a driver to print direct from programs written by me (if any) or even (short of importing again) how to print an SBASIC program. But I hope to learn all this (and much more) from your forthcoming articles.
It occurs to me that necessary aspects of a driver need to be included in one's own programs? As a matter of interest the Archimedes printer definition file print is enclosed. Amongst this is data which I had originally assumed would be incorporated in a manual accompanying the printer, but that was before I became aware of the real world.
[Regrettably, the listing was too long to print here - Editor]
You see how vital to the unknowledgable your articles will be. Please do not shrink from technicalities but be sure to make a glossary of terms & definitions as you go along (the general glossary you published in QLT was most gratefully studied). I have just been looking at an abstract from a book on Linux: the sheer gibberish in the form of unheard of acronyms and systems made it unreadable, so I shall not be purchasing the book.
Please continue your good work - I am sure there are many others who will be as grateful as I am.

Some Early Remarks about Paragraph Word Processor v2.01 - Ian Pizer

First there was Quill then came Text87 and after Prowess appeared we wondered who would write a Word Processor running under Prowess. Well Francois Lanciault has announced Paragraph and Wolfgang Lenerz is working on Proverb. It might seem odd to have these two new word processors soon fully available. We have, or will have, a multiple choice of hardware QL, AURORA, Q40, MILAN, plus emulators on various platforms) so why not a multiple choice of word processors. PC and Mac users are similarly confronted with WP choices.
I will not mention the bells and whistles which Francois is promising but mostly speak about my experience using Paragraph.
Being Prowess based you get the Font and Raster Manager features of Proforma. This means easy choice of font, character size, and eventually you can include images and diagrams. Presumably Proverb will have these features too. You must use the latest version of Prowess.
Using Paragraph v. 2.01 I typed in my first one page letter. Nothing sophisticated but it was a joy to use as it is simple to organise the layout and you can make and store templates for further use, really user friendly. I cannot compare with Quill as I have not used it for years. When I use Text87 I quite often lose my way and need to seek documents to get help. It would seem that all Paragraph features are readily avalable from the screen and the keyboard.
With version 2.01 there are naturally still some bugs and Francois is very active making improvements and adding intended features, and features requested by users. You need to register to get the latest versions as they become available.
■

Volume Listings

P.H. Tanner

I have always been wary of QDOS volumes containing files in nested subdirectories. To the computing rabbit they appear as an impenetrable morass: an impression not helped by all this talk of Levels, which only heightens, or should I say deepens, the resemblance to a Cambridgeshire fen.

It is my customary habit on receiving a disc so organised to put it on one side, unread, to be reformatted for other uses. But habits are made to be broken. I am currently awaiting the arrival of my Q40, and I have received in advance the support disc which contains the single ZIPped file QZ.ZIP. And this is one item that I am most certainly not going to junk.

However it UNZIPs into a plethora of subdirectories. My own home-brewed routines for file handling rely on directory files prepared by WDIR. And WDIR lies down and dies when presented with a structure like this. DDOWN, DLIST, et al, are all very well, but they operate on the default devices, which one may not wish to change while looking at a non-default volume. So something has had to be done.

That something was the writing of the attached subroutine wd (a$), which merges path information with the individual file specifications. A simple tweak to my procedures allows this additional gen to be assimilated. There are no checks on a$, and if it is passed to the procedure explicitly rather than as a variable, then it must have the trailing "_", and be enclosed between quotes.

This is in no way competitive with programs like The CAT: its only aim is to produce a flat file such as is shown in the listing. The code may be tweaked to do other things. For instance, if the full file statistics are required then WSTAT may be used in lieu of WDIR. Similarly, the paths could be stored in the PATH list, using PTH_ADD.

```
30000 DEFine PROCedure wd (a$)
30010  OPEN_OVER #5, ram7_scrat : STAT #5, a$ : PRINT #5 : WDIR #5, a$ :
       PRINT #5 : CLOSE #5
30020  REPeat
30030    OPEN #5, ram7_scrat : OPEN_OVER #6, ram7_dir : flag = 1
30040    REPeat
30050      IF EOF (#5) THEN EXIT : ELSE INPUT #5, b$
30060      IF ("->" INSTR b$) > 1 THEN
           b$ = b$ (1 TO LEN (b$) - 3): PRINT #6, b$; " =>":
           WDIR #6, a$ & b$: PRINT #6: flag = 0:
           ELSE PRINT #6, b$
30070      END REPeat : CLOSE #5, #6 : DELete ram7_scrat
30080    IF flag THEN EXIT :
         ELSE RENAME ram7_dir, ram7_scrat
30090    END REPeat : END DEFine
```

(Readers: please note, this is an SBASIC-specific listing, it does not work on SuperBASIC due to the use of unnamed repeat loops and multiple channel number CLOSE statements, for example)

Listing of ram7_dir, generated by calling "wd 'ram1_'", QZ.ZIP having previously been UNZIPped to ram1_:

```
RAM1
2406/8476 sectors

QZ =>
QZ_BOOT =>
QZ_BOOT_MSW =>
QZ_BOOT_MSW_----BOOT-MSW---
QZ_BOOT_MSW_BOOT
QZ_BOOT_MSW_QLIB_runtimes
QZ_BOOT_MSW_RAMDISC_cde
QZ_BOOT_MSW_RANDOM_code
QZ_BOOT_MSW_SEARCH2_code
QZ_BOOT_MSW_Turbo_TK_CODE

QZ_CLSC =>
QZ_CLSC_UBIK =>
QZ_CLSC_UBIK_CLASSICj_ROM
QZ_CLSC_UBIK_CLASSICh_ROM
QZ_CLSC_UBIK_CLASSICi_ROM
QZ_CLSC_UBIK_CLASSICk_ROM

QZ_CLSC_DOC =>

QZ_CLSC_DOC_QL =>
QZ_CLSC_DOC_QL_QL_txt

QZ_CLSC_DOC_AMG =>
QZ_CLSC_DOC_AMG_Classic_readme

QZ_CLSC_DOC_Q40 =>
QZ_CLSC_DOC_Q40_KEYTBLDT_TXT
QZ_CLSC_DOC_Q40_KEYTBLUK_TXT

...
```

Loops and knots

Mark Knight's guide to loops in SuperBASIC.

Two types of loop are available in SuperBASIC; many users of other BASIC languages may not think much of this, since most modern BASICs have at least three; FOR, REPEAT and WHILE or DO loops. SuperBASIC FOR loops offer far more than equivalents in other languages, and so do REPeat loops. The REPeat loop in SuperBASIC provides more features and flexibility than both DO and REPEAT loops combined. The first part of this text describes FOR loops and their use while the second part describes REPeat loops.

An important point to keep in mind is that all the behaviour described for the SuperBASIC examples in this text is that expected on a JS system unless otherwise specified. Some points will cover behaviour under Turbo or Q-Liberator where it differs from that found under the JS interpreter, and some Minerva or SBASIC peculiarities or extras are covered. If in doubt test examples on your system.

The SuperBASIC FOR loop:

SuperBASIC programmers will be familiar with the FOR loop, in most cases using it frequently, but it is suprising how often it is misused, introducing what can be exquisitely subtle bugs into programs. The SuperBASIC FOR loop also possesses special features not found in other variants of BASIC, and some of these are seldom used even though they provide a programmer with extremely useful facilities.

It is intended here to introduce the FOR loop from scratch, assuming the reader has written SuperBASIC but at a fairly simple level. Even the most elementary features of FOR loops are explained, as well as the most common mistakes made by programmers. In addition some of the finer points will be explained so readers understand how to make better use of these loops in future. Some of what is written may seem blindingly obvious, but what is blindingly obvious to you may be new to others, and some of the things you may learn from this text will be blindingly obvious to others. Read it all and enjoy gloating over what you already knew, and try and appreciate the things you didn't.

To begin with we deal exclusively with the long form of FOR loops, the kind spanning several lines. Short form FOR loops are discussed more briefly later. The basis of a FOR loop is usually similar to this first example:

```
100 FOR Chan=0 TO 2
110   PAPER#Chan;0
120   INK#Chan;0
130   CLS#Chan
140   BORDER#Chan;1,2
150 END FOR Chan
160 STOP
```

This will clear the three SuperBASIC windows without using MODE, a desirable program fragment. Another much less useful but clear loop is as follows:

```
100 FOR n=1 TO 40
110   PRINT n
120 END FOR n
130 STOP
```

...which will print the numbers from 1 to 40. If we want to print just the odd numbers we can put:

```
100 FOR n=1 TO 40 STEP 2
110   PRINT n
120 END FOR n
130 STOP
```

...and it will work. All SuperBASIC FOR loops must have a step value, so if the programmer doesn't specify one a step of 1 is used. If we want a step value that is not a whole number this is no problem, thus:

```
100 FOR n=1 TO 10 STEP .5
110   PRINT n
120 END FOR n
130 STOP
```

This will print 1, 1.5, 2, 2.5 etc.

Simple SuperBASIC FOR loops therefore have three numbers set up at the start:

1. The starting value.
2. The ending value.
3. The step value.

What the interpreter does with these values is simple in essence. First it loads the starting value into a special FOR loop variable named in the first statement of the loop. Then it runs through the loop, carrying out any further program statements it finds. When the matching END FOR is found it adds the step value onto the variable, chacks it against the ending value and makes a choice:

1. If the new value of the variable is outside the range set by the start and end values the program closes the FOR loop, continuing with any program statements after the END FOR.
2. If the new value is within the range set by the start and end values then the program loops back and continues from the statement after the FOR initiation statement (Line 110 in the latest example).

All three values are floating point values in standard SuperBASIC, though later some ways of using integer FOR loops will be explained. Integer arithmetic is quicker than floating point, so in theory integer FOR loops should be faster, though in practice there is often a very modest difference unless you compile your pro-

CAMBRIDGE Z88

OFFICE/FAX 01494-871319 (EEC) *W.N. Richardson & Co.*

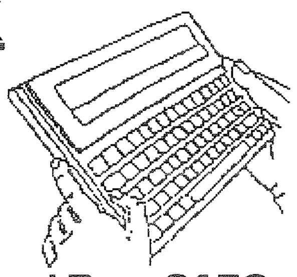
MOBILE: 07808 576118

E-MAIL: WNR@COMPUSERVE.COM

6 Ravensmead
Chalfont-St-Peter
Buckinghamshire SL9 0NB

THE IDEAL PORTABLE COMPANION FOR THE QL, PC AND MAC.

THE CAMBRIDGE Z88 A4 NOTEBOOK



WITH BUILT-IN WORD PROCESSOR, SPREADSHEET, DATABASE, BASIC, CALCULATOR, CLOCK, ALARM, CALENDAR & VT52 TERMINAL.
USES 4XAA ALKALINE CELLS (ca. 20 HOURS) ✱

WE DO REPAIRS, PART EXCHANGES, AND BUY Z88's & PARTS

NEW! CAMBRIDGE Z88 WITH OZ4+512k Internal Ram £150

ALSO AVAILABLE, 512k RAMPACK £40

DESCRIPTION	UK	USA
CAMBRIDGE Z88 ISSUE4-128K	**£120**	**$180**
CAMBRIDGE Z88 COMPUTER	**£99**	**$130**
☆ **RECONDITIONED Z88**	**£60**	**$90**
32k RAMPACK	£18	$25
128k RAMPACK	£38	$50
32k EPROM PACK	£18	$25
128k EPROM PACK	£24	$33
256k EPROM PACK	£55	$75
EPROM ERASER	£32	$43
PARALLEL PRINTER LEAD	£39	$50
SERIAL PRINTER LEAD	£12	$16
✱ **MAINS ADAPTOR (230vac; 6v,500ma)**	**£10**	
TOPPER (PROTECTIVE COVER)	£12	$16
CARRYING CASE (PLASTIC)	£10	$12

1/2 PRICE STARTER PACK
1 - Z88 ☆ PLUS TRANSFER KIT
+ 32K RAM
+ 32K EPROM
£72 ONLY!!

FILE TRANSFER TO OTHER COMPUTERS

SPECIAL Z88-QL SERIAL LEAD	£10	$15
COPY DISKS OF QUANTA PROGS IMP/EXP & ARCHIVE EXPORT	£2	$3
PCLINK KIT (For PCs)	£25	$35
Z88 TO MAC KIT	£25	$35
Z88 TO BBC KIT	£25	$35

} ALL WITH INSTRUCTION BOOKS, CABLES AND EPROMS AS REQUIRED!

POSTAGE UK UP TO £5. EEC £15. USA £20. OTHER COUNTRIES £30.

ALL THE STOCK IS WARRANTIED FOR 90 DAYS. IN THE EVENT OF REPLACEMENT BEING ARGREED, BUT THE ITEM BEING OUT OF STOCK AT THE TIME, A REFUND WILL BE MADE PROVIDED THE ITEM IS RECEIVED IN GOOD CONDITION.

QL & PC COMPUTER USERS WILL FIND THE CAMBRIDGE Z88 ESPECIALLY USEFUL FOR WORK AWAY FROM THE DESKTOP, WITH TRANSFER PROGS DATA CAN BE SAFELY EXCHANGED WITH THEIR DESKTOP SYSTEM.

W.N.RICHARDSON & CO CONTINUES TO PROVIDE FULL SPARES AND SERVICES FOR SINCLAIR COMPUTERS, QL & THE CAMBRIDGE Z88 gram. To obtain a true integer FOR we must be using Minerva, SBASIC or one of the SuperBASIC compilers.

OK, so far so good; what if we wish to count down rather than up? This is simple, just make the start value higher than the end value and set a negative step value, thus:

```
100 FOR n=100 TO 1 STEP -1
110   PRINT n
120 END FOR n
130 STOP
```

All very well so far. What happens if we specify a range that doesn't work for some reason? For example this:

```
100 FOR n=100 TO 1
110   PRINT n
120 END FOR n
130 STOP
```

Because no step is specified we get a step of 1, but this is no use, this range needs a negative step. Well the SuperBASIC interpreter steps straight to the statement after the END FOR if there is one, and in this case nothing is printed at all. A related but subtly different effect comes from loops like this one where the range is smaller than the step value:

```
100 FOR n=1 TO 2 STEP 5
110   PRINT n
120 END FOR n
130 STOP
```

...which will print 1 and finish. The loop is set up with n equal to 1, then runs through to the END FOR; at this point it adds 5 to the stored value and checks to see if it is in range. Naturally when 6 is checked against 2 the statement after the END FOR is executed and the programs stops.

Note it is perfectly legitimate to use variables for all three values, the start value, end value and step value thus:

```
100 StartValue=1
110 EndValue=10
120 StepValue=1.125
130 FOR n=StartValue TO
EndValue STEP StepValue
140   PRINT n
150 END FOR n
160 STOP
```

Obviously it is important to make sure if you do this that all the values work together to achieve what you want. Some programs may have reasons for using SuperBASIC code to calculate the start, end and step values; debugging such code embedded in a large program needs to be approached with care.

Another issue that causes a lot of apparently mysterious trouble debugging SuperBASIC programs is the confusion between END FOR and NEXT. For example the following is wrong:

```
100 FOR n=1 TO 10
110   PRINT n
120 NEXT n
130 STOP
```

...this should be changed to:

```
100 FOR n=1 TO 10
110   PRINT n
120 END FOR n
130 STOP
```

This is not a matter of style or of opinion, the first listing is wrong and should be changed. In this example there is no problem, but when using FOR loops in more sophisticated ways it is essential to use END FOR and NEXT properly or bugs will creep in that will be incredibly hard to find if you think NEXT and END FOR are equivalent. To understand why, look at the following listing, type it into your QL, but don't run it - yet. (This is an odd example in one way because a REPeat loop is actually better in this application than a FOR, but nevertheless it will make clear the correct use of END FOR and NEXT). Note the NEXT in line 500, type this as shown rather than using the correct END FOR as we want to see how this program goes wrong:

```
100 WINDOW 252,202,256,23
110 WINDOW#2;252,202,4,23
120 WINDOW#0;504,32,4,224
130 FOR Chan=0 TO 2
140   PAPER#Chan;0
150   INK#Chan;7
160   BORDER#Chan;1,2
170   CLS#Chan
180 END FOR Chan
190 FirstShow=0
200 SecondShow=0
210 Starting=1
220 Ending=1000
230 Stepping=1
240 TempFile$="ram1_Temp"
250 FileDev$="flp1_"
260 Separate$="_BAS"
270 DELETE TempFile$
280 OPEN_NEW#3;TempFile$
290 DIR#3;FileDev$
300 CLOSE#3
310 OPEN_IN#3;TempFile$
320 FOR ShowLoop=Starting TO Ending STEP Stepping
330   IF EOF(#3) THEN EXIT ShowLoop
340   INPUT#3;FileName$
350   IF Separate$ INSTR FileName$ THEN
360     FirstShow=FirstShow+1
370     PRINT FileName$
380     IF FirstShow=20 THEN
390       PAUSE
400       FirstShow=0
```

```
410       END IF
420     ELSE
430       SecondShow=SecondShow+1
440       PRINT#2;FileName$
450       IF SecondShow=20 THEN
460         PAUSE
470         SecondShow=0
480       END IF
490     END IF
500 NEXT ShowLoop
510 CLOSE#3
520 DELETE TempFile$
530 PRINT\"Finished"
540 STOP
```

After a careful look at the listing it is fairly obvious what the program is designed to do; it makes a directory of all the files on flp1_ but PRINTs all the _BAS files in a separate window. Each time either window is full the program will pause in the manner of a WDIR command, and when the end of the temporary file on ram1_ is reached line 330 is supposed to EXIT from the FOR loop.
EXIT can be used to exit earrly from any named FOR or REPeat loops, and is usually used inside an IF structure.
After the loop in this program the temporary file is closed and deleted then a message is printed to the screen to let the user know the program has finished.
Run the program as listed with a disk of mixed files in flp1_ and see what happens. No message is printed (line 530) and the temporary file remains open on ram1_ (try DIR ram1_) because the interpreter never exits from the FOR loop in spite of line 330. The command line cursor flashes as soon as all the filenames have been printed and lines 510 to 540 are skipped. This is one area where SBASIC differs from SuperBASIC as if you miss out line 540 you will get an error message at the end of the program run.

The reasons for this are simple; when SuperBASIC was designed the FOR loop was intended to be closed at a single point by END FOR, so when the EXIT statement is reached the interpreter starts to look for the end of the loop so it can exit from it. It searches through all the program statements and finally reaches line 540 without having found an END FOR ShowLoop, so the program finishes. As far as the interpreter is concerned, lines 510 to 540 are inside a half finished FOR loop, not outside the loop at all.
If you simply change line 500 to:

```
500 END FOR ShowLoop
```

...then the program works perfectly, though you may have to type CLOSE#3 from the command line first if you have run the previous version.
After that you may be left wondering what NEXT is actually intended for; well if you add the following line to the program (as well as making the suggested change to line 500) it becomes obvious:

```
425  NEXT ShowLoop
```

...run the resulting program.

The FOR loop is short-circuited if the file is not one with "_BAS" in it, so only those files are shown. It is just as important not to use END FOR here as it is not to use NEXT in line 500; this is because a FOR loop is only supposed to be closed in one place. So END FOR is intended to close a FOR loop, while NEXT is intended to "short circuit" the same loop: EXIT can make a premature jump to the statement after the END FOR. NEXT and EXIT are normally used inside an IF or SELect structure.
As well as correcting line 500 we'll do things wrongly and change line 425 to:

```
425  END FOR ShowLoop
```

...then run the program again, several times, each time with a different disk in flp1_ and we will soon see another problem.
A filename is printed to #2 at the program end even though it shouldn't be. This is because after the EXIT at line 330 the interpreter is looking for the close of the FOR loop, and when it finds line 425 it allows the program to continue from the next valid program statement - in this case line 430. So the END FOR at line 500 would be redundant as the loop has already closed by this point.
SBASIC is extra helpful and gives an error message while Turbo will comment and try to correct the program, warning the user about the action taken. Put it right instead of relying on Turbo, as the Turbo corrective action won't work in this case anyway.
Let's sum up what we have learned so far:

1. Every FOR loop has a start value, an end value and a step value.
2. Every FOR loop requiring termination should have and END FOR not a NEXT as the termination.
3. Shortcut to the start of the loop with the next FOR value is done using NEXT.
4. Early exit from the loop can be done with EXIT.

One thing you need to be aware of when using NEXT is what happens if the loop value has already reached the ending value of the loop; NEXT then doesn't jump back to the beginning, but falls through to the next statement. If it is likely this will cause problems then follow the NEXT with an EXIT from the same loop.

Now let's look at another useful feature, one which at first glance may not seem useful at all. Recall that if the step is not supplied SuperBASIC uses a value of 1 for the step; in addition FOR loops will work without an ending value, taking it from the starting value instead.

Try this:

```
100 FOR n=0
110   PRINT n
120 END FOR n
130 STOP
```

This uses 1 as the step value but sets the ending value the same as the starting value; naturally the loop runs through only once and stops, but it does work. So it works, what? We could just type this instead:

```
100 n=0
110 PRINT n
120 STOP
```

Why is it useful to have the FOR loop work with just a single value? For testing purposes, that's why. If you are developing a large complex program with a FOR loop in it you can set it up like this for one test (note this is not a useful example on its own, it represents fragments from a larger program):

```
10100  FOR WeightFactor=StartValue
10110  REMark some useful code goes in here
10530  END FOR WeightFactor
```

...then for another test change line 10100 to this:

```
10100  FOR WeightFactor=EndValue
```

...and run the test again.

You finish knowing the loop works for the two extreme cases, which saves a lot of testing time and often helps speed up development of large programs. Only when the program is believed to be ready would we insert the "StartValue TO EndValue" required in line 10100.

Another feature of a SuperBASIC FOR loop is rarely used, though it has even more potential than the above example; in fact it is an extension of the above idea. So far we have looked at FOR loops with a single start, ending and step value, in other words, a single range; but SuperBASIC allows multiple ranges sepearated by commas, thus:

```
100 FOR n=1 TO 5, 20 TO
    25, 40 TO 45
110   PRINT n
120 END FOR n
130 STOP
```

...try it. You can even mix ranges with positive and negative step values, like so:

```
100 FOR n=1 TO 10, 20 TO
    11 STEP -1
110   PRINT n
120 END FOR n
130 STOP
```

An extension of this idea, using the fact that we need only specify the start value for each range, is this:

```
100 FOR n=10,20,30,40
110   PRINT n
120 END FOR n
130 STOP
```

...or even this:

```
100 FOR n=1,100,3,47,96,
    17,34,12
110   PRINT n
120 END FOR n
130 STOP
```

...or indeed any other organised or random list of values! We can also mix ranges and single values:

```
100 FOR n=1,10 TO 20,7,47,3,
    6 TO 25,19,99 TO 79 STEP
    -1
110   PRINT n,
120 END FOR n
130 STOP
```

These variations of FOR are not possible in any other version of BASIC that I am aware of, and certainly were not possible in any version of BASIC around when the QL was launched. Along with using END FOR, the unique NEXT and the EXIT statement these implied range and multiple range facilities make SuperBASIC FOR loops extremely flexible and powerful.

Short form FOR loops:

The form of FOR that causes a lot of trouble is the short form, which uses multiple statements on a single line to define the loop. This is intended for very short simple tasks and often works well, like this:

```
100 FOR n=1 to 20: PRINT
    n,SQRT(n)
```

Because the interpreter finds more statements on the line after the FOR has been defined it counts the rest of the line as the inside of the loop and no END FOR is required. If no more statements exist after a FOR definition it is assumed to be the long form and an END FOR on a separate line is needed.

In many early BASIC variants the short FOR was the only available form; this made complex FOR loops a nightmare mess of multiple statement lines, often including IF structures, difficult to read and therefore to debug. Not only is it difficult to debug an overcomplicated short form FOR loop, but it's also very hard to ensure that the interpreter to run it will work properly too.

Home Electronics Service

Home Electronics Service is proud to announce that we are now the North American distributor for the QL, and Z88. We provide Sales, Service, and software. We also provide service for the Spectrum and ZX81.
We have been a part of the Sinclair scene since 1982, repairing ZX Spectrums for Sinclair Research in England.

Look at our web page at
www.members.tripod.com/hes_computing/hes1.html
E-Mail: 74601.1535@compuserve.com

Hours of operation is
Monday-Friday
1:00 pm to 9:00 pm
central time zone.

Phone 210-661-4376
John R. Rish
Home Electronics Service
5222 Kazen Dr.
San Antonio Tx 78219 USA

WE SUPPORT SINCLAIR

QBOX USA

COMPUTER BULLETIN BOARD SERVICE
(810)254-9878

We support *all* SINCLAIR COMPUTERS
(QL, SPECTRUM, ZX81, Z88, Thor, QXL)

- ❑ Now in our 6th year on-line round the clock since October, 1993
- ❑ Full message area and File Download areas
- ❑ We carry all popular SINCLAIR message areas from Europe
- ❑ Calls from 14.4k--300 baud are welcome
- ❑ QBOX - USA runs on a SINCLAIR QL with Super Gold Card, Hermes, QUBIDE, 200MB drive, USR sportster 14.4 modem

→ NO FEES CALL US

Because of this, many short form FOR loops that include IF or further nested FOR loops don't work properly on the QL. Note this doesn't just apply to the Sinclair ROMs, like AH, JM, JS and MG, it includes Minerva (and probably SBASIC) as well. Th best known bug is one that would never have been discovered if programmers on the QL all did things properly, but often they don't. If you use:

```
100 FOR Test=1 to 50:GO SUB
    120:PRINT "."
110 STOP
120 RETurn
```

...this will only print one full stop, not 50! This is because the GO SUB also acts like an END FOR and terminates the FOR loop, a weird effect certainly not intended by the designers of Super BASIC. GO SUB should never be used in SuperBASIC and SBASIC anyway as named PROCedures make a program far easier to read, but this example is just one. Because of the various bugs it is best to use short form FOR loops only for simple tasks and use the long form most of the time.

Minerva and SBASIC extras:
The Minerva ROM permits integer FOR loops, so you can type:

```
100 FOR Chan%=0 TO 2
110   PRINT#Chan%;"This is
      channel ";Chan%
120 END FOR Chan%
130 STOP
```

...not only does this occupy slightly less RAM (but more storage space when SAVEd) than it would without the "%" symbols but it runs faster too. Note that AH and JM systems will let you enter such a program but it will generate errors if you try to run it. SBASIC also allows integer FOR loops and runs them even faster.

Q-Liberator and Turbo:
Both Q-Liberator and Turbo permit integer FOR loops, though you can't always enter them into the interpreter (you may be using a JS ROM or similar) so they can make use of compiler directives to achieve this. To make a FOR loop into an integer loop in Q-Liberator on a JS or MG ROM system use this:

```
100 DEF_INTEGER Chan
110 FOR Chan=0 TO 2
120   PRINT#Chan;"This is
      channel #";Chan
130 END FOR Chan
140 STOP
```

...with Turbo the technique is identical but the keyword is different:

```
100 IMPLICIT% Chan
110 FOR Chan=0 TO 2
120   PRINT#Chan;"This is
      channel #";Chan
130 END FOR Chan
140 STOP
```

In both cases, using DEF_INTEGER or IMPLICIT% makes any variables listed in the parameter list into integers, not just those used in FOR loops. All kinds of arithmetic operations can then be done using integers and this can offer a considerable speed increase in some cases.

The SuperBASIC REPeat loop:
As with FOR loops SuperBASIC offers a form of loop considerably more flexible than that available in other BASICS. Both long and short forms exist; as before we look first at the long form. The definition of the REPeat loop is deceptively simple:

```
100 REPeat Program
110 Choice=CODE(INKEY$(5))
120 IF Choice=32 THEN NEXT Program
130 BLOCK RND(10 TO 30),RND(10 TO 30),RND(0 TO 100),RND(0 TO
100),RND(0 TO 255)
140 IF Choice=27 THEN EXIT Program
150 END REPeat Program
160 STOP
```

The REPeat can be typed as REP and the interpreter will add the rest of the keyword. The loop must be given a name in SuperBASIC though SBASIC allows unnamed loops. Just as with FOR loops EXIT can be used to leave the loop at any time using any condition or no condition at all. Next can also be used to jump to the beginning of the loop at any stage, so holding down the spacebar in the above program will stop plotting, pressing the ESCape key will exit to the STOP statement and holding any other key down will speed up the plotting. It is quite normal to nest REPeat loops inside other REPeats or FOR loops, and a good use of such loops (along with the SELect structure) might be something like this:

```
1000 REPeat Program
1010   BORDER 1,2
1020   CLS
1030   PRINT\\"ESCape key to exit"\\"F1 to beep"\\"F2 to redraw menu";
1040   REPeat GetKey
1050     Choice=CODE(INKEY$(#0,-1))
1060     SELect ON Choice
1070       =27
1080         REMark ESCape key pressed
1090         EXIT Program
```

```
1100        =232
1110          REMark F1 pressed
1120          BEEP 10000,10
1130        =236
1140          REMark F2 pressed
1150          EXIT GetKey
1160      END SELect
1170    END REPeat GetKey
1180 END REPeat Program
1190 STOP
```

In a real program it would not be normal to EXIT without prompting the user as this listing does in line 1090. Note that NEXT and EXIT can refer to any loop as long as they are inside it, so it is fine in the above example to exit from GetKey or from Program as long as the EXIT statement used is inside the loop it exits from. The fact that it is also inside another loop (or loops) will not matter as the EXIT will automatically exit from more than one loop.

This also applies to the NEXT statement so line 1150 can be changed to:

```
1150          NEXT Program
```

...and no change in the way the program works will be seen. Although it would only be noticed in a large complex program using "NEXT Program" in line 1150 would actually work slightly faster than "EXIT GetKey" does. This is because of the precise way in which these statements work.

Using "EXIT GetKey" sets the SuperBASIC interpreter searching through the program for the END REPEat so it can exit the loop. The next statement found, "END REPeat Program" sets it off looking through the program from the start for the opening of the REPeat loop. If "NEXT Program" is used instead then the first of these searches is skipped; the interpreter only has to search for the opening "REPeat Program".

Short form REPeat loops:

Just as with FOR, a REPeat loop can be defined on a single line. If the interpreter finds more program statements after a REPeat definition no END REPeat is needed and the remainder of the line is treated as the inside of the loop. So the following works:

```
1000 REPeat GetKey: Choice=
     CODE(INKEY$(-1)):IF
     Choice<>0 THEN EXIT
     GetKey
1010 PRINT Choice,CHR$(Choice)
```

Although I know of no bugs in the short form of REPeat there are numerous bugs in the short form of IF so beware of using complicated short form REPeat loops containing IF structures. If you do need to use them test carefully and if something seems to be going wrong because of a short form bug change the program to use the long form of REPeat.

SBASIC extras:

SBASIC allows the following:

```
1000 REPeat
1010   IF INKEY$=CHR$(27) THEN EXIT
1020   PRINT CHR$(RND(0 TO 255));
1030 END REPeat
```

...in other words, a REPeat loop with no name. This is something to be used with great caution in my view, if it is used at all. First it represents a great way to make a program less readable; imagine two or three nested unnamed REPeat loops! A second problem is it makes loops that use EXIT and NEXT much harder to read and debug. In my view the programming effort required to provide the unnamed structures in SBASIC would have been better spent on something else, but if you disagree and wish to use them do be extra careful with your testing and debugging. For more details about unnamed loops read the SBASIC text and examples in the SMSQ/E manual.

Q40 Building

Tony Firshman

The first I really contacted with the Q40 was when Peter Graf emailed me last September. He developed the Q40 a few years ago, but did not have the time to make it himself, and asked me if I wanted to take it on to production.

After prevaricating for far too long (sorry Peter), I finally decided that because parts costs were high, I could not market it myself at a sensible price after adding VAT. A lot of parts had already been bought by Peter, and the VAT could not be reclaimed on these. Roy Wood though is not VAT registered, so we arrived at a compromise - Roy would market it and I would manufacture it.

OK - so we got down to some hard searches for good prices. Two hundred and forty pounds for the 68040 processor - what!!! Peter's target price was £20. Urmmm. £16 pounds for the Timekeeper RAM. £35 for the four logic chips etc etc. Parts costs were coming out more than twice target, even without the 68040!

Looks like it is going to die before it even starts.

However we got a chance of fifty 68040 processors for 'only' £75 each. Roy snapped them up. Peter Graf found some cheaper ones very soon afterwards. Still more than the target, but well...... We managed to get the £75 processors resold, logic chips and timekeeper RAM came out cheaper, so the project was at last on.
However to get good prices, most parts had to be bought in quantity, so we decided to get advance firm Q40 orders before buying any parts. At last earlier this year we had enough orders to jump. Still no money in the bank, but we ordered the initial production parts on credit. We held our breath as we did not even have operating systems anything like ready.
I had this mountain of parts - and lots of holes - some 75,000 of them. "After all those dots you will hate the Q40" Peter said.

I started climbing the mountain.....

There were a few hurdles on the way, like my programmer not liking 16 bit eproms. Only one in 4 programmed and there were two per board - you can work out the chances of a working pair. That was compounded by SMSQ suffering from permanent IRQ7 from the I/O board and crashing, and two out of three ram chips not working in the Q40. It was a long nervous while before I got SMSQ working.
Along the way, a debug version of SMSQ arrived. It was a while before I realised how brilliant it was. Tony Tebby instructed "Connect a VT100 terminal to the Q40s serial port. ENTER to the first two strings, and then type 'g' ‹ENTER›" Have you guessed yet?
First string was 'AT'. A glimmer of understanding
Second was 'ATDT 0033 ‹*TTs tel no snipped*›'
Are you there yet? I wasn't quite.
Third was 'QMON›'
.... and there was QMON sitting at the start of ROM, so 'G' started.
Then I twigged - I could connect the Q40 to a modem, and then power up with Tony Tebby sitting with a modem/qtpi in rural France. He could spend hours debugging the Q40 at international phone rates. Very clever indeed.
Fortunately for my bank balance he decided he could do nothing anyway as breakpoints cannot be set in the ROM area. The Q40 crashed without any useful QMON data. Great debugging tool though, and one Tony uses himself locally. The machine can be practically dead, and certainly have no working screen.
I am now near the summit, first orders are being shipped, SMSQ and QDOS Classic work pretty well, and I don't hate the Q40!

.... now back to those ELEVEN boards that don't work.

Assembly Language Programming - Part 5

Norman Dunbar

The first part of QLTdis is as follows. I have decided to split the project over a number of asm files. This reduces the amount of memory required when editing and allows those of us with smaller memories (QL that is !) to hopefully manage to keep the assembler and editor in memory at the same time.

The files are as follows:

QLTdis_asm - The main file. This is the only filename that you specify to GWASL as this file automatically inserts the others at the appropriate places when it is assembled.

Equates_asm - This file defines a number of names for numbers. This makes the code more readable and, hopefully, easier to understand and less prone to errors. For example, I use the name 'green' for the number 4 - when specifying colours.

Init_asm - This file holds the source for the initialisation routines. These being the 'once only' and the 'once per loop. initialisation that is required by QLTdis.

Diss_asm - This file holds the routines that disassemble and print the instructions. At present these are simply defined as 'stubs' which do nothing. We are not advanced enough to write the full routines. These will be built up as the series progresses.

Utils_asm - This file holds the source code for various sub-routines that are used from various places in all the above files. This too will be built up as we identify other useful routines.

Term_asm - This file holds the source for the routines used to terminate each loop and those which terminate the entire program.

As I am a very firm believer in commenting source code, there are numerous comments provided along the way. These should be fairly self explanatory.

As we are writing a useful QDOS application, we must indulge in a little foray into the inner workings of QDOS from time to time. Opening files or clearing screen channels are all handled by QDOS. We need do nothing more than set up the correct set of parameters and call the appropriate routine.

To call QDOS, we can use a TRAP #1, TRAP #2 or TRAP #3 instruction with the appropriate function code in D0 - or - call a QDOS Vectored Utility by setting an address register - usually A2 - to the contents of a memory location and using JSR (A2) to call the actual routine.

At this stage in the proceedings you will have to trust me with this because we don't really have the space to cover the full set of TRAP or Vectored utilities that QDOS provides - suffice to say there are dozens. Jochen will sell you a wonderful REFERENCE manual for QDOS. Later on, after the tutorial part of this series is over, I may (real job & my wife Alison permitting!) cover QDOS in more depth.

At some places in the following code I have placed a NOP instruction - this is a place marker for me and we will be building more code into these places later on - probably in the next issue. The NOP instruction is a 2 byte 4 clock cycle 'waste of time'. It does nothing, changes no flags and simply serves to use up 4 clock cycles.

In addition I have used the TST.L D0 instruction - this too will be fully covered later, but for now treat it as a quick version of CMPI.L #0,D0 because that is what it does. It will set the Z flag if D0.L contains zero and reset it otherwise.

Preparation

Format a new disc and copy QED, its help file, all the GWASL files onto it. This is now your master disc. You should use this in flp1_.

Format a new floppy and place it in flp2_. This is your source disc.

Type the following commands:
`EXEC flp1_qed`

When prompted for a filename, type in `flp2_QLTdis_asm` and press ENTER.

When asked for a workspace size, type in 20K and press enter.

Type the following into the editor. Comments are indicated by an asterisk in column 1 or after a semi-colon after the various instructions.

At various times, press F3 then type SA to save your work. Here then is the first part of QLTdis_asm.

By the way, where a line such as 'IN WIN1_SOURCE_QLTDIS_DISS_ASM' appears, you should replace the device name with flp2 - or wherever you are saving the files to.

```
* =====================================================================
* QLTdis_asm
* =====================================================================
* A simple disassembler program for the 68008 instruction set.
*
* Written for QL Today using George Gwilt's GWASL assembler.
* Thanks to George for his free donation of this assembler.
*
* QLTdis is copyright - Norman Dunbar 1998/1999
* However, permission is given for unlimited use and abuse by whoever
* provided that the original author (me) gets some credit !
*
* ---------------------------------------------------------------------
* WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
* ---------------------------------------------------------------------
* GWASL does not like TAB characters in its source files. Make sure that
* your chosen editor is set to convert TAB into a number of spaces (I use
* 4 spaces) AND/OR that you simply type spaces instead of TAB to line up
* columns. This is VERY IMPORTANT.
* ---------------------------------------------------------------------
* WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
* ---------------------------------------------------------------------

            in win1_source_QLTdis_equates_asm

* =====================================================================
* CODE STARTS HERE.
* First part of any job file is a standard QDOS Job header. When EXEC'd
* the first instruction executed will be the one at 'start' which is a
* 2 byte short branch.
*
* QDOS JOB HEADER FORMAT :
*
* 6 bytes starting on an even address (most assemblers handle this for
* you). These 6 bytes are usually a short branch instruction followed by
* 4 bytes of padding (as in the following) or a JMP.L label.
* The id word $4AFB - this indicates a QDOS job to the system.
* A single word holding the size of the job name in bytes.
* The actual job's name.
* =====================================================================
start       bra.s   qltdis          ; 2 bytes short jump
            dc.l    0               ; 4 bytes padding
            dc.w    $4afb           ; Job identifier must be at location 6

            dc.w    6               ; 6 bytes in job's name
            dc.b    'QLTdis'        ; 6 bytes of job's name


* =====================================================================
* The job's code proper starts here. This is the 'main control' part and
* defines the program's operation from a top down viewpoint.
*
* On entry here, the following registers are set up :
*
* A6.L = Base address of the Job's header ie, the address of 'start'.
* (A6,A4.L) = Bottom of the job's dataspace area - the first usable byte.
* (A6,A5.L) = Top of job's dataspace area - the last usable byte.
* A7 = The job's USP (User Stack Pointer)
*
* There may also be data on the job's stack, but we are ignoring it here
* and will discuss it later when examining QDOS.
* ======================================================================
qltdis      bsr     job_init            ; Once only initialisation
            bne.s   exit                ; Errors - cannot continue

main_loop   bsr     loop_init           ; Once per disassembly initialisation
            bne.s   exit                ; Errors - cannot continue
            cmpi.l  #-1,pc_addr(a4)     ; Has user quit ?
            beq.s   exit                ; Yes - all done

dis_loop    bsr     diss                ; Disassemble one instruction
            bsr     print               ; Print disassembly
            bsr     escape              ; Check ESC or done (Z = carry on)
            beq.s   dis_loop            ; Not done - do some more


* ======================================================================
* User pressed ESC or we are past the stop address. Terminate this loop by
* closing the 'printer' file etc. Then go back and ask the user for more.
* ======================================================================
main_end    bsr     loop_term           ; Once per disassembly termination
            bra.s   main_loop           ; Prepare for another disassembly


* ======================================================================
* The program is done or a massive error has occurred. Bale out.
* ======================================================================
exit        bra     all_done            ; We are finished.


* ======================================================================
* This is the end of the 'main control' part of the program. Following on
* from here are the various routines used by the above.
* ======================================================================

            in  win1_source_QLTdis_init_asm
            in  win1_source_QLTdis_diss_asm
            in  win1_source_QLTdis_term_asm

* ======================================================================
* And following on from here is the various utility sub-routines that are
* used from any part of the above code.
* ======================================================================

            in  win1_source_QLTdis_utils_asm
```

This is the end of QLTdis_asm so save it using F3 followed by SA and enter. We now need to start on a new file, so press F3 and this time type R/flp2_equates_asm/ and press ENTER - type the following into the new file. Don't forget to save as you go. This is EQUATES_ASM.

```
* ======================================================================
* EQUATES_ASM.
*
* This file is included by QLTdis_asm and is used to define various
* constants that are used within the source code. This is instead of
* a whole lot of 'magic' numbers.
*
* For example, using GREEN instead of 4 is a bit more meaningful.
* ======================================================================

* ----------------------------------------------------------------------
* These are offsets from the start of the job's dataspace where working
* variable are stored. The dataspace is held at (A4) in the job's code.
* ----------------------------------------------------------------------
con_id      equ     0                   ; Id for title channel
con_id2     equ     4                   ; Id for main output
prt_id      equ     8                   ; Id for printer channel
pc_addr     equ     12                  ; Start address of next instruction
pc_end      equ     16                  ; Stop after this address

* ----------------------------------------------------------------------
* These are simply user friendly names instead of numbers for various
* bits and bobs, colours etc.
* ----------------------------------------------------------------------
black       equ     0                   ; Colour code for mode 4 black
red         equ     2                   ; Red
green       equ     4                   ; Green
white       equ     7                   ; White
linefeed    equ     10                  ; Linefeed character

* ----------------------------------------------------------------------
* Constants for use with job control commands. (It doesn't matter if I
* have two names with the same value ! )
* ----------------------------------------------------------------------
infinite    equ     -1                  ; Infinite timeout
me          equ     -1                  ; Id for 'this' job
```

And here endeth EQUATES_ASM. Save it and use the R command to start a new file called INIT_ASM, then type in the following code.

```
* ======================================================================
* INIT_ASM.
*
* This file is included by QLTdis_asm and is used to define various
* initialisation routines and parameters used by them. The jobs main
* initialisation and each loop initialisation are defined in this file.
* ======================================================================


* ======================================================================
* One off job initialisation :
* ======================================================================
* 1. Set job's dataspace pointers etc
* 2. Set mode 4 if not already set
* 3. Open title CON_ channel
* 4. Open output CON_ channel
* 5. Check errors - cannot continue if detected.
*
* Returns Z set if all ok.
* ======================================================================
job_init    adda.l  a6,a4               ; A4.L = start of dataspace

* ----------------------------------------------------------------------
* Set mode 4 if not already set. Do not change from TV to monitor or
* vice versa. We must preserve the display type if we reset the mode.
* ----------------------------------------------------------------------
            moveq   #mt_dmode,d0
            moveq   #-1,d1              ; Read current mode
            moveq   #-1,d2              ; Read current display type
            trap    #1                  ; Do it
            tst.l   d0                  ; Did it work ?
            bne.s   con_done            ; No, bale out

            tst.b   d1                  ; 0 in D1.B = Mode 4
            beq.s   mode_4              ; No need to set mode 4

            moveq   #mt_dmode,d0
            clr.l   d1                  ; We need mode 4
            trap    #1                  ; Set mode 4 (d2 = display type)
            tst.l   d0                  ; Check it
            bne.s   con_done            ; Bale out if errors detected

* ----------------------------------------------------------------------
* Mode 4 is current mode. Open the title window at the top of the screen.
* ----------------------------------------------------------------------
mode_4      lea     con_def,a1          ; Window definition
            movea.w ut_con,a2           ; Utility to define a window etc
            jsr     (a2)                ; Do it
            tst.l   d0                  ; Did it work ok ?
            bne.s   con_done            ; No, exit routine
            move.l  a0,con_id(a4)       ; Store console id for title channel

* ----------------------------------------------------------------------
* Now open the output window underneath the title one.
* ----------------------------------------------------------------------
            lea     con_def2,a1         ; Output window definition
            movea.w ut_con,a2           ; Utility again
            jsr     (a2)                ; Do it
            tst.l   d0                  ; Did it work ?
            bne.s   con_done            ; No, exit routine
            move.l  a0,con_id2(a4)      ; Store console id for output channel

            moveq   #0,d0               ; No errors detected
con_done    rts
```

```
*-----------------------------------------------------------------------
* Definition for title window channel
*-----------------------------------------------------------------------
con_def     dc.b    red                 ; Border colour
            dc.b    1                   ; Border width
            dc.b    white               ; Paper/strip colour
            dc.b    black               ; Ink colour
            dc.w    448                 ; Width
            dc.w    24                  ; Height
            dc.w    32                  ; Start position x
            dc.w    16                  ; Start position y


*-----------------------------------------------------------------------
* Definition for output window channel
*-----------------------------------------------------------------------
con_def2    dc.b    red                 ; Border colour
            dc.b    1                   ; Border width
            dc.b    white               ; Paper/strip colour
            dc.b    black               ; Ink colour
            dc.w    448                 ; Width
            dc.w    200                 ; Height
            dc.w    32                  ; X org
            dc.w    40                  ; Y org


* =======================================================================
* Once per disassembly initialisation :
* =======================================================================
* 1. Clear screen & show headings etc
* 2. Ask for start address, validate
* 3. Ask for end address, validate
* 4. Ask for 'printer' channel
* 5. Clear screen
* 6. Show details
*
* Returns Z set if no errors.
* If user quits, sets pc_addr(a4) = -1
* =======================================================================
loop_init   bsr     headings        ; Display headings etc in title window

*-----------------------------------------------------------------------
* Clear screen & ask for start address
*-----------------------------------------------------------------------
            movea.l con_id2(a4),a0      ; output console id
            bsr     cls                 ; Clear screen
get_start   lea     start_addr,a1       ; Our prompt
            bsr     prompt              ; Print it
            bsr     input               ; Wait for user to type something
            tst.w   d1                  ; Anything typed ?
            bne.s   got_start           ; User typed something

            move.l  #-1,pc_addr(a4)     ; Store a 'user quit' address
            bra     init_done           ; Finished

*-----------------------------------------------------------------------
* User has typed an address. Convert from a string into a long.
*-----------------------------------------------------------------------
got_start   nop                         ; WE COME BACK TO THIS NEXT TIME

*-----------------------------------------------------------------------
* Ask for end address - if we get here
*-----------------------------------------------------------------------
get_end     lea     end_addr,a1         ; Our prompt
            bsr     prompt              ; Print it
            bsr     input               ; Wait for user input
            tst.w   d1                  ; Anything typed ?
            bne.s   got_end             ; User typed something

            move.l  #$ffffffff,pc_end(a4)  ; Very high end address
            bra     get_printer            ; Go for the printer name next
*-----------------------------------------------------------------------
* User has typed an address. Convert from a string into a long.
*-----------------------------------------------------------------------
got_end     nop                         ; WE COME BACK TO THIS NEXT TIME

*-----------------------------------------------------------------------
* Ask for additional file/printer device
*-----------------------------------------------------------------------
get_printer lea     print_dev,a1        ; Our prompt
            bsr     prompt              ; Print it
            bsr     input               ; Wait for user input
            lea     pr_dev,a0           ; Storage for printer name
            move.w  d1,(a0)+            ; Save filename length
            subq.w  #1,d1               ; Adjust for dbra loop

*-----------------------------------------------------------------------
* At this point :
*
* A1.L = first character in input buffer for printer name
* A0.L = first character position in pr_dev buffer
* D1.W = Number of character in filename minus 1
*-----------------------------------------------------------------------
move_prname move.b  (A1)+,(A0)+         ; Move a single character
            dbra    d1,move_prname      ; And the rest

*-----------------------------------------------------------------------
* Clear screen again - in case user typed silly stuff !
*-----------------------------------------------------------------------
            movea.l con_id2(a4),a0      ; Output console id
            bsr     cls                 ; Clear screen

*-----------------------------------------------------------------------
* Show addresses and filenames again prior to start.
*-----------------------------------------------------------------------
show_start  lea     start_addr,a1       ; Our prompt
            move.w  (a1),-(a7)          ; Stack string length
            move.w  #14,(a1)            ; Reduce string length
            bsr     prompt              ; Print it
            lea     start_addr,a1       ; A1 is trashed by 'prompt'
            move.w  (a7)+,(a1)          ; Restore original length

*-----------------------------------------------------------------------
* Print start address here - WE WILL DEAL WITH THIS NEXT TIME
*-----------------------------------------------------------------------
            nop
            bsr     line_feed           ; Print a new line

show_end    lea     end_addr,a1         ; Our prompt
            move.w  (a1),-(a7)          ; Save current string size
            move.w  #12,(a1)            ; New 'pretend' length
            bsr     prompt              ; Print it
            lea     end_addr,a1         ; A1 is trashed remember
            move.w  (a7)+,(a1)          ; Restore original length

*-----------------------------------------------------------------------
* Print end address here - WE WILL DEAL WITH THIS NEXT TIME
*-----------------------------------------------------------------------
            nop
            bsr     line_feed           ; Print a new line

show_prname lea     print_dev,a1        ; Our prompt
            move.w  (a1),-(a7)          ; Store original string length
            move.w  #15,(a1)            ; New length
            bsr     prompt              ; Print it
            lea     print_dev,a1        ; Prompt trashes A1
            move.w  (a7)+,(a1)          ; Restore original length
            lea     pr_dev,a1           ; Printer device name
            bsr     prompt              ; Print printer filename
            bsr     line_feed           ; And a final linefeed

init_done   moveq   #0,d0               ; No errors
            rts


start_addr  dc.w    37
            dc.b    'Start address (ENTER only to quit) : '

end_addr    dc.w    39
            dc.b    'End address (ENTER only = until ESC) : '

print_dev   dc.w    32
            dc.b    'Printer device (ENTER = none) : '

pr_dev      ds.w    26          ; 52 bytes for max file name & length
```

And this is the end of INIT_ASM. Save it and re-enter the editor with another new file. This time it is a short one called DISS_ASM. Type the following into it.

```
* ===================================================================
* DISS_ASM.
*
* This file is included by QLTdis_asm and is used to define the main
* disassembly and printing routines.
* initialisation and each loop initialisation are defined in this file.
* This file may be further sub-divided as each routine for the various
* instruction types are coded - this depends upon the size of said
* routines.
* ===================================================================


* ===================================================================
* Disassembly routine(s) :
* ===================================================================
* 1. Save current address (start of the new instruction)
* 2. Decode current word - and any subsequent words of data etc
* 3. Fill buffers etc
*
* Returns Z set if any errors.
* ===================================================================
diss        moveq    #0,d0                   ; No errors
            rts


* ===================================================================
* Print routine(s)
* ===================================================================
* 1. Print current address
* 2. Print hex data
* 3. print instruction
*
* Returns Z set if no errors.
* ===================================================================
print       moveq    #0,d0                   ; No errors
            rts
```

As you can see, that was quite short. The next file is called TERM_ASM and it too is quite short. Type this in - don't forget to use the R command to create it!

```
* ===================================================================
* TERM_ASM.
*
* This file is included by QLTdis_asm and is used to define various
* termination routines and any parameters used by them. The jobs main
* termination and each loop's termination are defined in this file.
* ===================================================================


* ===================================================================
* Terminate main loop :
* ===================================================================
* 1. Close 'printer' file
* 2. Tidy addresses etc in storage ready for next time (if required)
* ===================================================================
loop_term   moveq    #0,d0               ; No errors
            rts


* ===================================================================
* All_done :
* ===================================================================
* 1. Remove the current job - forcibly
* 2. Loop around just in case something went wrong and the job was not
*    properly removed.
* ===================================================================
all_done    moveq    #mt_frjob,d0  ; Force Remove a job
            moveq    #me,d1        ; The current job
            move.l   d0,d3         ; Any error code to send to Superbasic
            trap     #1            ; Kill this job, its channels and its
memory

            bra.s    all_done      ; Should never get here - sanity check!
```

And finally, use F3 and R again to create UTILS_ASM - this will be quite a large file. Type the following into it. Don't forget to save as you go along.

```
* ===================================================================
* UTILS_ASM
*
* This file is included by QLTdis_asm and defines a large number of sub
* routines that are called from various parts of the program. These are
* called more tahn once and so have been defined as sub-routines to avoid
* duplicating code. In addition, any errors will only need to be fixed
* here once.
* ===================================================================


* ===================================================================
* Are we done yet routine(s)
* ===================================================================
* 1. Is new current address >= stop address ?
* 2. Has ESC been pressed ?
*
* Returns Z set if not done and if ESC not pressed.
* ===================================================================
escape      moveq    #1,d0                   ; ESC pressed
            rts


* ===================================================================
* Headings :
* ===================================================================
* 1. Clear title window
* 2. Print titles
* ===================================================================
headings    movea.l  con_id(a4),a0       ; Title channel id
            bsr.s    cls                 ; Clear screen
            lea      mes_title,a1        ; Title string
            bsr.s    prompt              ; Print title string
            rts

mes_title   dc.w     55
            dc.b     'QLTdis - copyright Norman Dunbar 1998/1999',linefeed
            dc.b     'Version 1.00'


* ===================================================================
* CLS :
* ===================================================================
* 1. Clear the (screen) channel whose id is in A0.
* ===================================================================
cls         moveq    #sd_clear,d0        ; CLS
            moveq    #infinite,d3        ; Infinite timeout
            trap     #3                  ; CLS title window
            rts


* ===================================================================
* Prompt :
* ===================================================================
* 1. Print the string at (A1) to the channel in A0.
*
* Z set if all ok, unset if not.
* ===================================================================
prompt      movea.w  ut_mtext,a2         ; Print a string utility
            jsr      (a2)                ; Print it
            tst.l    d0                  ; Check for errors
            rts


* ===================================================================
* Line_Feed :
* ===================================================================
* 1. Print a linefeed to the channel in A0.
*
* Z set if all ok, unset if not.
* ===================================================================
line_feed   movem.l  d1/a1,-(a7)         ; Preserve any registers reqd
            moveq    #io_sbyte,d0        ; Send one byte to channel
            moveq    #linefeed,d1        ; Byte to send = linefeed
            moveq    #infinite,d3        ; Timeout
            trap     #3                  ; Do it
            movem.l  (a7)+,d1/a1         ; Restore
            tst.l    d0                  ; Set Z if errors
            rts

* ===================================================================
* Input :
* ===================================================================
* 1. Wait for user input from the channel id in A0.
*
* Returns the input length (not counting the ENTER character) in D1.W
* Returns the address of the first character in the buffer in A1.L
* Preserves the channel id in A0.L
* Z set if all ok, unset if not.
* ======================================================================
input     lea     buffer+2,a1         ; Our buffer address plus 2
          move.l  a1,-(a7)            ; Save it on the stack
          moveq   #io_fline,d0        ; Input some bytes (inc linefeed)
          moveq   #60,d2              ; Buffer size maximum
          moveq   #infinite,d3        ; Inifinite timeout
          trap    #3

          move.l  (a7)+,a1            ; Restore buffer pointer
          subq.w  #1,d1               ; Subtract the linefeed character
          move.w  d1,-2(a1)           ; Store length in buffer
          tst.l   d0                  ; Did it all work ?
          rts


buffer    ds.w    16        ; 60 chars for input plus 1 word for size.
```

And this is finally the end. You should now have 6 individual files.

Assemble it

So we have our source ready to assemble. Try this then:

```
EXEC flp1_gwasl_bin
```

When prompted press 1 (one) to start assembling. When prompted for a filename, type FLP2_QLTdis_ASM and press ENTER. (If FLP1_ or something different is printed, simply rub it out and type the correct device use CONFIG to set the default device name. CONFIG is available on most Pointer Environment pograms if not on the cover disc).

Assembling will now start and PASS 1 will be printed followed by PASS 2. The options to assemble or quit will then be shown. What happened?

On FLP2 you will now have a file called QLTdis_LST which you MUST load into the editor - F3 r/flp2_qltdis_lst/ - so that you can check for errors. If there are errors in your code, check that it matches mine exactly and change it if not comments don't make any difference so don't worry about them.

Keep editing and assembling until you get no errors. You now are able to run your first assembler program.

EXEC flp2_QLTdis_bin and be amazed !!!

QLTdis

The screen shows a 'split' window with the copyright message and program title at the top. Under this is a prompt asking for a 'Start Address' - type ENTER only to quit or type anthing you like. At this stage the program reads what you typed but does nothing with it.

You are now prompted for an end address, as above, type anything and press ENTER. Alternatively, just press ENTER.

Finally, you are asked for a 'printer' file. Type in anything here as well and press enter.

And you are now back at the 'Start Address' prompt. Press ENTER alone to quit from the program.

What is going on?

The program, as it stands, is a functional one which does nothing. The start addresses etc you type in are not acted upon. The program simulates the once only initialisation stage - the split window shows this. Then the loop initialisation stage - printing the titles and asking for addresses etc.

What you don't see is the next stage where the parameters you gave it are displayed (but not the addresses - that comes later). You don't see it because the main loop does nothing and then simulates ESC being pressed to quit from the main loop. The loop initialisation is then run again - this does a CLS and this is why you don't see the output.

This will continue until you press ENTER only for the start address.

Congratulations on getting this far!

By the way, the code above was imported directly from my source files. These were assembled using GWASL and have been through exaustive testing using QMON2 to monitor what was going wrong - yes there were a few errors - so it should be correct. If your code refuses to co-operate then I'm afraid it will probably be down to a typing error on your part - sorry! ■

Using ZIP and UNZIP

Dilwyn Jones

Unless you have the Archivers Control Panel, trying to learn to use the ZIP and UNZIP programs from a BASIC command can be daunting, thanks to the sheer number of options available. Dilwyn explains some basic commands of ZIP and UNZIP.

These simple commands will achieve the most important tasks, namely zipping up all files on a disk into one large compressed archive files, and that of decompressing an archive file.

To zip the entire content of a disk in FLP1_ into a file called SMALL_zip on FLP2_ (assuming the ZIP program itself is also on FLP2_)

```
EX FLP2_ZIP;'-9 FLP2_SMALL_ZIP FLP1_*'
```

The FLP1_* at the end just means all files on FLP1_. If you prefer to supply a list of files instead, this is easy too:

```
EX FLP2_ZIP;'-9 FLP2_SMALL_ZIP FLP1_BOOT
FLP1_EXAMPLE_TXT FLP1_EXAMPLE_EXE'
```

Note that there should be a space between the '-9' (which tells ZIP to go for maximum compression rather than speed), and a space before the FLP2_ and FLP1_ parts at the end, but NOT before the *

The -9 part is optional - compression will be slightly faster if this is omitted, but the resultant files will be slightly larger as a result.

To decompress the files, you need to use the UNZIP program. This extracts files from the archive and decompresses them to the drive named in the DATA_USE command.

For example, to decompress all files from FLP1_SMALL_ZIP to a disk in FLP2_, assuming UNZIP is also on FLP1_:

```
d$ = DATAD$ : REMark remember what it
                         was before
DATA_USE FLP2_
EX FLP1_UNZIP;'FLP1_SMALL_ZIP'
DATA_USE d$ : REMark restore default drive
```

The last example extracted all files from the archive called SMALL_zip. Occasionally, you may wish to extract just a single file from an archive, e.g. an instructions file, or one file to replace a damaged copy on your working disk. This can be done simply by adding the name of the file or files at the end of the command.

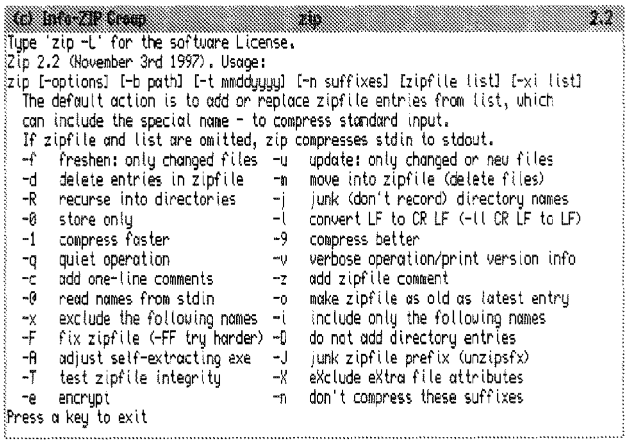
```
EX FLP1_UNZIP;'FLP1_SMALL_ZIP manual_txt'
```

If you'd just like to have a look at a list of files in the archive, this can be done using the '-l' switch (N.B. lower case letter L not number 1):

```
EX FLP1_UNZIP;'-l FLP1_SMALL_ZIP'
```

These programs use a system of 'wildcards' or vague names when referring to filenames. By using the character '?' in a filename, you can be vague about a spelling or any single character you were unsure of. By using the '*' character any text will match up to it, e.g. B* could match up all files which had filenames ending with 'B', or FLP2_*_DOC would match up any _DOC files on FLP2_. This is rather hard to get used to at first, but can be a powerful facility and is quite commonly used on other computers.

There is a simple method getting a help screen up in these programs, which will list a short text file showing the main commands and their useage. If you simply type in EX FLP1_ZIP or EX FLP1_UNZIP the program has no commands to tell it what to do, so it shows you a help text instead.



```
(c) Info-ZIP Group                zip                              2.2
Type 'zip -L' for the software License.
Zip 2.2 (November 3rd 1997). Usage:
zip [-options] [-b path] [-t mmddyyyy] [-n suffixes] [zipfile list] [-xi list]
  The default action is to add or replace zipfile entries from list, which
  can include the special name - to compress standard input.
  If zipfile and list are omitted, zip compresses stdin to stdout.
  -f   freshen: only changed files  -u   update: only changed or new files
  -d   delete entries in zipfile    -m   move into zipfile (delete files)
  -R   recurse into directories     -j   junk (don't record) directory names
  -0   store only                   -l   convert LF to CR LF (-ll CR LF to LF)
  -1   compress faster              -9   compress better
  -q   quiet operation              -v   verbose operation/print version info
  -c   add one-line comments        -z   add zipfile comment
  -@   read names from stdin        -o   make zipfile as old as latest entry
  -x   exclude the following names  -i   include only the following names
  -F   fix zipfile (-FF try harder) -D   do not add directory entries
  -A   adjust self-extracting exe   -J   junk zipfile prefix (unzipsfx)
  -T   test zipfile integrity       -X   eXclude eXtra file attributes
  -e   encrypt                      -n   don't compress these suffixes
Press a key to exit
```

One very useful ZIP option can be to FIX a problematical zipped file - I have had to do this to recover a zip file created by an old version of QL zip, which I was unable to access. Using the -F or -FF option, ZIP fixed the problem well enough for the files to be accessible. Try -F first; the -FF option is a 'try harder' one for when the simpler option fails. Where possible, always keep copies of these files in case something goes wrong again and you end up worse off than before!

```
EX FLP1_ZIP;'-F FLP1_PROBLEM_ZIP'
```

Both ZIP and UNZIP QDOS versions are configurable with the standard QL Config or MenuConfig programs; you can set defaults such as the length of time it stays on screen to display results after doing something.

The ZIP and UNZIP programs are free software which you can get from the usual sources of QL PD and Freeware software, or they can be downloaded from Jonathan Hudson's Dead Letter Drop Web site:

http://www.jrhudson.demon.co.uk

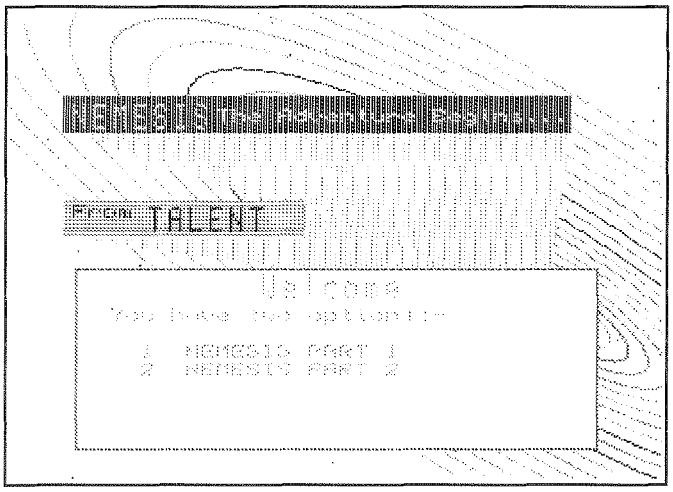
Please try to make sure you use the Info-Group Zip and Unzip versions, which are still maintained and "official" this is the version you'll get from Jonathan's Web site.

■

Adventures on the QL Part 2 -Nemesis

Darren Branagh continues his look at adventure games available for the QL user, with a review of Nemesis, from RWAP Software.

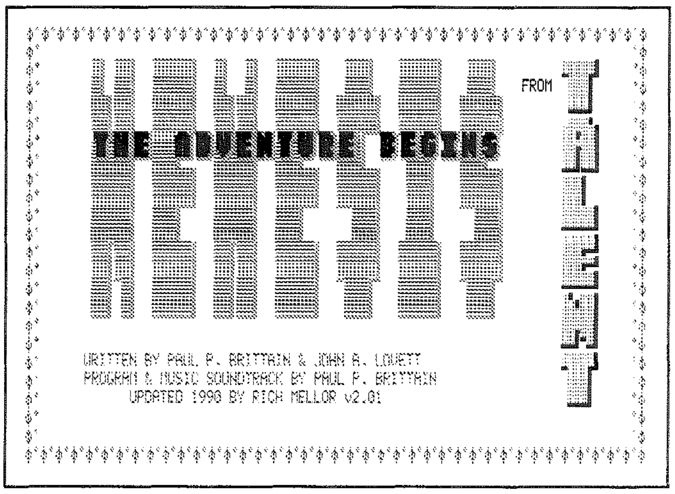
You sit at the helm of a small Landing pod that transported you down to the planet's surface from your orbiting ship. You are dressed in leather trousers and jacket, leather boots and a light cotton shirt....



```
FROM TALENT

Welcome
You have two options :-

1  NEMESIS PART 1
2  NEMESIS PART 2
```

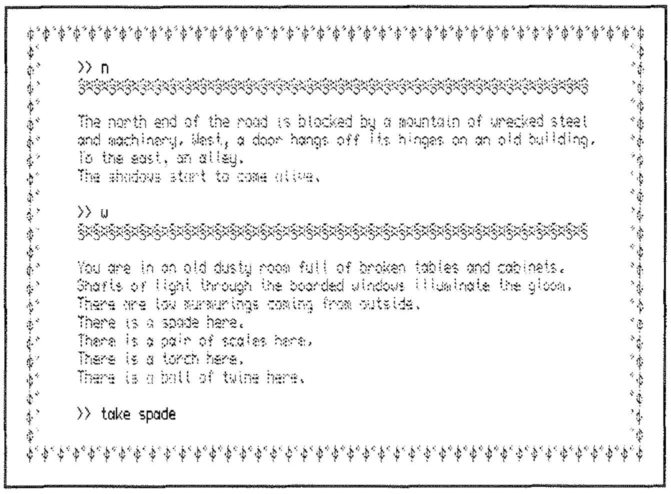
So begins NEMESIS, the second text adventure game to receive our attention in this series of reviews. Unlike THE PRAWN, this game is a much more serious affair. The plot is simple you are a butch hero sent to a distant planet to aid the long suffering inhabitants from their tyrannical overlord, called NEMESIS. He has been ruling this planet with an iron fist for years and is threatening to extend his grip across the entire galaxy - unless of course you can help, by seeking out NEMESIS and destroying him.
And help you must, but it won't be easy - at his disposal NEMESIS has hoards of troops, killer robots, carnivorous creatures, and piles of death traps littering the planet's surface.

GAMEPLAY
As I've come to expect from Rich Mellor's software at this stage, gameplay is surprisingly good. However, it is difficult, even



```
THE ADVENTURE BEGINS
                                              FROM

WRITTEN BY PAUL P. BRITTAIN & JOHN A. LOVETT
PROGRAM & MUSIC SOUNDTRACK BY PAUL P. BRITTAIN
     UPDATED 1990 BY RICH MELLOR v2.01
```

though the game gives you plenty of warning if you are heading into a dangerous area, with statements like "you hear noises and figures start to move in the shadows" - although you know that trouble is near, and your death is certain unless you change direction, you still end up heading that way and getting killed!! I tried the game on several platforms - It worked without difficulty on JS and JM rom QL's, both on Trump Card and Gold Card, and with some gentle persuasion on the QXL and QPC both on my Desktop Pentium and Laptop systems.
The problem with QXL/QPC systems (and all SMSQ/E systems I should think) arises with the file TC on the disk, which is called by the BOOT. Line 26 gives an error "misplaced END DEFine" - this is easily remedied by splitting the preceding DEFine PROCedure line in two separate statements after the first semi-colon, onto two lines instead (and renumbering the program). The game works perfectly on all systems thereafter - Thanks to Dilwyn Jones for the cure to this one, caught in his house while I started this review!!
Error reporting is quite good although I did find a few locations where my commands were ignored and a "I don't understand" message or similar is not displayed. This is a minor detail and didn't annoy me that much at all.

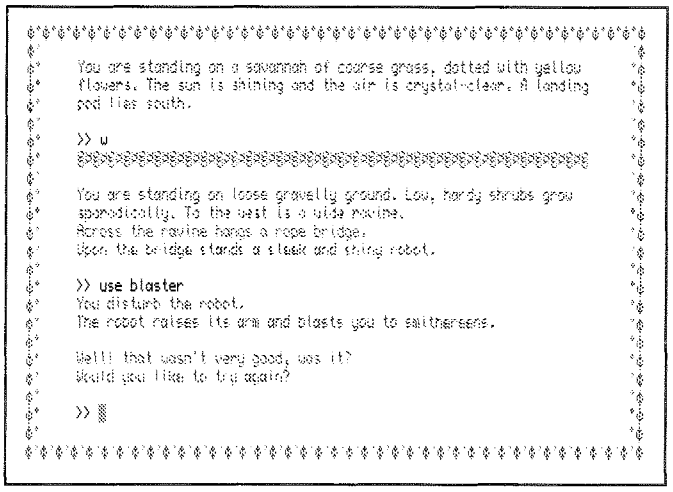


```
>> n

The north end of the road is blocked by a mountain of wrecked steel
and machinery. West, a door hangs off its hinges on an old building.
To the east, an alley.
The shadows start to come alive.

>> w

You are in an old dusty room full of broken tables and cabinets.
Shafts of light through the boarded windows illuminate the gloom.
There are low murmurings coming from outside.
There is a spade here.
There is a pair of scales here.
There is a torch here.
There is a ball of twine here.

>> take spade
```

Also, there are a couple of good features I hadn't seen before, as in THE PRAWN. These are two new commands, namely PON and POFF. These are very useful, and are used to control output to the printer - PON switches it on, POFF stops it. I enjoy making maps of adventure games, so I can find my way back to locations quickly, and get to various items I need at various points. This made this so much easier, printing out details as I proceeded.
Another nice command is SAY. This allows you to interact and talk to various beings in the game to futher your goal, and have a conversation with them.

THE MANUAL
The manual for the game is distributed on the disk, as a Quill_doc file. It is quite comprehensive, although concise at only 3 pages or so - a blessing for those of you (like me) who hate reading manuals. Although short, it tells you all you need to know - including the scenario, loading instructions, a few playing hints and tips and a general list of commands.

DISPLAY AND DESCRIPTIONS
The display is clear and readable, similar to the default QUILL colouring of green text on a black background. I liked the way THE PRAWN used multi coloured text, and feel that NEMESIS could have benefitted from the same treatment, if only to "jazz" it up a little.
The text descriptions of your locale are first rate brilliant. They are moody and grim, and get your heart racing at just the right moments - especially when danger is near. They are just enough to allow your imagination to run riot, and this is the primary objective of every good adventure game.



```
You are standing on a savannah of coarse grass, dotted with yellow
flowers. The sun is shining and the air is crystal-clear. A landing
pod lies south.

>> w

You are standing on loose gravelly ground. Low, hardy shrubs grow
sporadically. To the west is a wide ravine.
Across the ravine hangs a rope bridge.
Upon the bridge stands a sleek and shiny robot.

>> use blaster
You disturb the robot.
The robot raises its arm and blasts you to smithereens.

Well! that wasn't very good, was it?
Would you like to try again?

>> ▒
```

SHOULD I BUY IT?
Yes, you should!! NEMESIS is just as enjoyable as THE PRAWN, this one being more engrossing for its gaming appeal and hero aspect than the humour. Contact Rich and get a copy today. I haven't caught up with NEMESIS himself yet, but god help him when I do!!

[Nemesis was originally written by Paul Brittain and John Lovett, and originally published by Talent. Now re-released in an updated version by RWAP Software. Nemesis Mk II costs just £10.00]

Printer Control Codes - a dreaded Subject? - Part 2

Dilwyn Jones

Where can I get a list of Control for my Printer?

Most printer manufacturers include a list of control codes somewhere in the printer's manual. Or perhaps that ought to read 'used to include', as the modern trend is not to include user programming information any more. Some printer manuals include them, others don't. The reason for this is that with operating systems on certain other computers, the drivers take care of all that for you, and in theory you need not dabble with the command set, so you don't need to know them. Certainly, if you ring a printer company's helpline for programming information you tend to get blank looks and gasps of amazement that you'd actually want to tackle programming for printers.

In practice, you can get programmers' reference manuals from the main printer companies, as long as you know where to call. They tend to be expensive publications (often 20 to 30 pounds in this country), and my own experience tells me that although the printer manuals refer to these programming tomes, it can actually be good fun trying to find someone within these companies who knows (a) what they are, and (b) how to supply you with one. In trying to get hold of such a book for one of my printers, I was given a major runaround, being told to ring several numbers in turn, none of whom knew what I was on about and several trying to sell me a replacement manual instead (it just goes to show, you can actually get replacement manuals for those second hand printers whose greedy or thoughtless owners decided to keep the manuals!) until I eventually spoke to a very thoughtful and knowledgeable lady who listened sympathetically to my story and eventually sent me a complimentary copy, as by then I must have run up the book's value in telephone calls. Hewlett Packard, for example, have an extra approach. They operate a fax on demand service where you can call a service called HP First (U.K. fax number: 0800-960271) which will fax you pages of information about the products, which can include pages of programming information. Experience tells me you need plenty of time, and a new roll of paper in your fax machine, but at least it's a cheap and no fuss method of getting hold of information as long as you have access to a tone dialling fax machine (i.e. most modern types). You call the number, listen to the voice prompts, and press the number key corresponding to your choice from the list offered. The first time, you ask for a contents list, then the second time you choose the specific information you want faxed to you. It's such a good service (when not engaged) that I think there must be a QL somewhere in HP controlling it! Hewlett Packard also have a Web site where you can gain additional printer information, the home page address is **http://www.hp.com**. In fact, these days, many companies have Web sites and if you have a modem and Web access it can be quite helpful to visit these Web sites. Most printer manufacturers also publish their address and telephone number and Web site address in their manuals too.

How do I send these commands to the printer?

The examples I'll give are all using BASIC, but the principle is the same for most languages.

Before we can send commands to the printer, we need to open a channel to the port to which the printer is connected. A channel is simply a numbered pathway for information to travel from one part of the computer to another. For example, commands you type into BASIC on the QL normally appear in channel 0 (#0) at the bottom of the screen. Channel numbers 0, 1 and 2 are used for BASIC's screen display normally. We can open another numbered channel to the printer. Assuming you have your printer connected to serial port number 1, you could type in:

```
OPEN #3,SER1
```

(the symbol before the number 3 is known as a HASH character).

If your printer is connected to a parallel port on a Super Gold Card, for example, you could use:

```
OPEN #3,PAR
```

You don't have to use #3, you can use any reasonable value, but it is customary to use the lowest available channel number to save space in the channel table inside the computer. Other channels may be open to files, or other windows on the screen, at the same time as the printer channel is open, but you should always close such a channel when you have finished with it:

```
CLOSE #3
```

While a channel is still open, you can send command values to it as follows. Suppose we wanted to switch to italic text printing:

```
PRINT#3,CHR$(27);CHR$(52);
```

Note how each number is followed by a semi-colon. This helps prevent the PRINT command from messing things up by sending newline characters where they are not wanted, for example.

If you have Toolkit 2 active on your system, or you are using SMSQ, you can also use the BPUT command to send values directly, without having to use the CHR$ function. This time, we don't have to worry about semi-colons:

```
BPUT #3,27,52
```

If you remember what we discussed about CHR$ and CODE, we could also write the commands like this, where we know what value represents which character:

```
PRINT #3,CHR$(27);'4';
```

or

```
BPUT #3,27,CODE("4")
```

In certain contexts, any one of the methods of writing the command may appear more meaningful, but it is largely down to you as they all do the same job in the end.

In general, if you can switch something on, you may also need to switch it off. So many commands have two sets of values, with one of the numbers differing slightly depending on which option you want. In the Epson command set, for example, it is common for the last of the numbers in a given printer command to vary its value depending on which option you want to select. So if you are sending a command to change colours on a colour printer, you may find that the last value may be any of 8 (if it has 8 colours available!), so black might be 0, while yellow may be 6 (the actual colour numbers on a given printer may not match exactly with the QL number). This may be the same for selecting fonts on a printer which has a choice of them built in.

Q Branch

Shop hours :
Mon - Fri
10am - 6pm
Wed / Sat
10am - 2pm

SMSQ/E

Gold Card / Atari / QXL Version
£ 76.00
Various Atari versions : call for details

QPC

£ 90.00 (£ 70.00 SMSQ/E Owners)
With CUESHELL Module :
£ 110.00 (£ 90.00 SMSQ/E Owners)
We can now supply colour laptops with QPC already installed ! The systems are secondhand and vary considerably so you will need to call for details but a typical starting price for a 486 DX 70 Toshiba with 350 Meg hard disk and 12 Ram complete with QPC and Carrying case is
£ 500.00 (+postage)

Q Branch News

The Q 40 is now slowly being released and the first few customers have all expressed great satisfaction with it in spite of the fact that the principle operating system, SMSQ/E, is still lacking some elements and is still under construction.
We are now at v 2.92 of SMSQ/E for this and it now supports most QL programs although there are still a few problems to be solved. Check it out at a QL show when you have the chance.
Francois Lanciault has authorised us to collect royalties and register copies of Paragraph. We suggest that if you are interested in this program you first get the demo version from us and then return the disk and register if you want the full package. Why not update ProWesS at the same time because you will need the latest version to run the software.

Standard Q 40 package consists of :
Motherboard / 16Mb RAM / I/O card

Prices

Q 40 with QDOS Classic O/S	£ 330.00
Q 40 with SMSQ/E O/S	£ 360.00
Extra 16Mb RAM	£ 28.00
Floppy Disk Drive	£ 23.00
Hard Disk Drive	P.O.A.
In Tower Case (New)	£ 45.00
In Desktop Case (2nd User)	£ 20.00
Keyboard	£ 10.00
Mouse	£ 6.00

Prices do not include postage and packing.

PROGRAMMING

QD 98	£ 53.00
QD + QBasic	£ 69.00
QD + Qliberator + QBasic	£ 110.00
Qliberator	£ 50.00
Master Spy v 3.3	£ 30.00
QPTR	£ 37.00
Easyptr pt 1 & 2 (together)	£ 33.00
Easyptr pt 3 (C library)	£ 16.00
QMake	£ 18.00
QMon / JMon	£ 22.00
Basic Linker	£ 22.00
DISA 3	£ 37.00
QMenu	£ 16.00

SBASIC / SuperBASIC Reference manual £40.00
Update Sheets / disk from earlier versions £ 15.00
Postage extra : UK = £ 7.00 other places call for details.

New Versions
New Prices

Don't forget you can pop into our shop in sunny Worthing and have a day by the sea too ! You can get disk drives and all sorts of computer peripherals as well as your QL supplies.

UTILITIES

QSpread 99	**£ 59.00**
Updates	**£ 18.00**
FiFi 2	£ 22.00
QSup	£ 32.00
Cueshell 2	£ 30.00
Qload / Qref	£ 15.00
Disk Mate 5	£ 16.50
QPAC 1	£ 20.00
QPAC 2	£ 40.00
QTYP 2	£ 30.00
QLQ	£ 32.00
LDUMP	£ 26.00
Q Count	£ 25.00

Text 87

£ 79.00

Typset 94	£ 29.00
Fountext 94	£ 39.00
2488 drivers	£ 29.00
Epson ESC/P2 drivers	£ 26.00

Text 87 is the only QDOS / SMSQ wordprocessor capable of handliing the full screen on the Aurora / QXL / QPC systems. New drivers are currently being written. WATCH THIS SPACE !

'Just Words' by Geoff Wicks
THESAURUS, STYLE CHECK, SOLVITPLUS 3
£ 10.00 ea / ANY 2 PROGRAMS £ 18.00 / ALL 3 PROGRAMS £ 25.00

NEW VERSION --- STYLE CHECK 3
THESAURUS V4.01 / SOLVIT PLUS 3 v 3.00
Pointer drivens versions of Geoff Wicks popular programs. (needs Hot_rext, WMAN and PTR_GEN or SMSQ/E to run)
NOW ONLY £ 10.00
Upgrades from previous versions £ 2.50 + S.A.E.
New manual for Style Check 3 £ 2.00
Spelling Crib : PD program £ 1.50 +SAE
or Free if you buy all three programs

Q Branch

Feeling out on a limb ?
Reach out for Q Branch.
Suppliers of Quality QDOS/SMSQ products
Hardware and Software.

Tel +44 (0) 1903-200212 fax +44 (0) 1903-208070
email qbranch@qbranch.demon.co.uk web : http://www.qbranch.demon.co.uk

Q Branch

The Bank Volt, 6 Coronation Buildings
Ham Road, Worthing.
W. Sussex. BN11 2NN. UK.

Hardware

QXL II	£ 180.00
Super Gold Card	£ 160.00
Recycled Gold Card	£ 60.00 *
Aurora	£ 100.00
Qubide	£ 55.00
Qplane	£ 25.00
Aurora cables	£ 3.00
Aurora rom adaptor	£ 3.00
The 'Braquet'	£ 16.00

* when available.

ProWesS

ProWesS	£ 48.00
DATAdesign	£ 24.00
Fontutils	£ 30.00
File Search	£ 12.00
PFlist	£ 12.00
Fontpack	£ 60.00
LINEdesign v 2.16	£ 24.00
PWfile	£ 18.00

Paragraph

The ProWesS word processor

¶ Demo version £ 1.50 + postage

Full Registered version £ 18.00

There are currently available :
2 Used Gold cards
2 New Super Gold cards

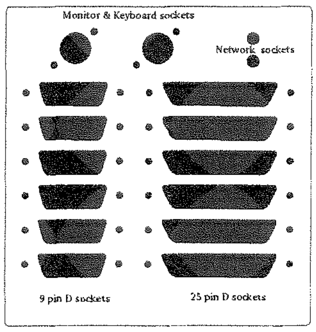
QXL Supercharge Kit

We have a few 33MHz 68040 processors, etc. which we can fit to your existing QXL to increase its power and speed - call us for details

Package deals

QXL II + SMSQ/E	£ 220.00
Aurora + SMSQ/E	£ 160.00
Above package + Monitor	£ 380.00

The MC - Plate



ONLY £ 6.50 !

All the socket holes you will ever need on the back of your tower case. Just rip off the silly PC thing and screw on this one. No cutting needed. What could be easier ?

Q Branch Programs

The Knight Safe 1 - standard version	£ 30.00
The Knight Safe 2 - with compression	£ 35.00
Q - Count Pointer driven home accounting	£ 25.00
Q - Route v1.07 Route finding programme	£ 25.00

Coming soon !
Mark Knight's Fractal Programs.
Stunning animated fractals on your QL !

2nd user 14" SVGA Colour Monitors ! (suitable for Aurora)
Ready for use and giving all screen resolutions from 512 x 256 to 1024 x 576
prices start at £ 25.00 + shipping *Only a few left !*

VISA

We can accept payment by VISA, Mastercard and Switch. You can also pay by Eurocheques made out in Sterling or a Sterling cheque drawn on a UK Bank. Prices include Post and Packing in Europe.

MasterCard

In our example of switching on italics above, the equivalent command to turn off italics may be:

```
PRINT#3,CHR$(27);CHR$(53);
```

It is very important to note that usually numbers should be sent as the values, not as a string of the individual digits. For example, to send the ESCAPE character, you must

```
PRINT CHR$(27);
```

not

```
PRINT '2';'7';
```

That may have seemed obvious, but sadly it's not always that easy. Some commands for Hewlett Packard Deskjets allow certain functions to be set using individual digits, with the printer making sense of it all at the receiving end. You have to study the commands and try to understand what is expected. Another possible source of confusion may arise where some commands allow single digit numbers to be sent as a value, or as the characters. Epson printers, for example, use the ESC '-' n sequence (where n is 0 or 1) to turn underline off or on. Many printers would allow both these options:

```
PRINT #3,CHR$(27);CHR$(45);
CHR$(1);

PRINT #3,CHR$(27);CHR$(45);
'1';
```

It is rather dangerous to assume that this is always the case and you should try to establish which is the preferred option wherever possible, in case you find that it works on one printer, but not on another supposedly compatible one! This is the sort of nightmare scenario programmers often find themselves contending with, that a facility originally included with the best of intentions ends up making life a little difficult for the programmer and the poor user who feels like throwing bricks at the programmer, who only got caught in the middle of such a no-win situation. On the whole, on the QL scene, we are actually quite lucky in having programmers who will respond quickly to problems caused by such an example, provided the source of the difficulty can be pinpointed.

I Intend to write a program which needs to send Printer Control Codes. Which Printer type should I support?

As many as possible, without making the printer driver installation procedure needlessly fiddly or cumbersome. If you really can't be bothered trying to cope with all types of printers, try to establish what kind of printer the target user of your software is likely to use. Most QL users have Epson-compatible printers (or at least most of their printers have Epson compatible modes) or HP-Deskjet compatible printers. The command sets for these two types of printers are very different, but by coping with both these types, there is a good chance your program will work with most of the printers likely to be used by QL users.

Gee Graphics! (on the QL?) - part 10

Herb Schaaf

From the LISTINGS_zip file of the QL Today Volume 3 cover disk get the file "Gee8_bas", or type it in from the listing that accompanied GG#8 in the Jan/Feb 1999 issue of QL Today. Then merge the listing from this article GG#10. Running the program creates two windows side by side. In the right window the tetrahedron is created as before. When you touch "P" a companion tetrahedron is created in the left window. When compared to the right image, the left image is seen to have been rotated on the Y-axis by a small angle. This discrepancy is used by our eyes and brain as a clue for depth determination and you now have a stereo-pair of images like those used in stereoscopes.

If you have binocular vision you can achieve the illusion of depth by letting your right eye see the right image only, and letting your left eye see only the left image. It can help if you have an opaque divider to enforce this condition. Alternatively you may wish to stare into the distance, seeing double as you may have done for those single-image random-dot stereograms, so that you see 3 tetrahedrons. The central one will seem to appear as a 3 dimensional object and the two small vertical lines over the center of each image will have fused to form an additional middle line over the pseudo 3-D image.

Check that the distance between the small vertical lines above the images agrees with the center to center value stated on the screen. To calibrate your screen choose [Q]uit; enter the command line BLOCK #0, 256,16,128,16,4 and measure the length of the green bar in millimeters. Edit the program and use this value on LINE 410.

After you have a pair of images you can change the spacing between them by use of the left and right arrows. See what range of values your eyes can accomodate. A spacing equal to your own eye-to-eye distance is suggested. My eyes are about 65 millimeters apart. You can set the default value for your eyes on LINE 175.

The angle of discrepancy can be changed by use of the up and down keys while viewing the pair. Explore other values, and notice that as you 'go-through-zero' the object will change so that it goes from convex to flat(at zero) to concave. See how extreme the angles can be in both directions, and also experiment with both the image-to-image and eyes-to-image distances.

You can figure what the angle would be based on your eye-to-eye distance and your eyes-to-object distance. My eyes-to-screen distance is about 42 centimeters so I set a default angle of -9 degrees on LINE 177.

If your lines of vision cross over in a 'cross-eyed' fashion you get a smaller inverted illusory image and will want to change the sign of the angle.

A delightful little paper-back book "Eye and Brain", now in a 4th edition by Richard L. Gregory has several pages about stereo vision, and lots of other interesting material on how we see. (ISBN 0-691-02456-1)

```
----- Listing --  Tetrahedra_stereo_bas --------------------

100 REMark Tetrahedra_stereo_bas
110 REMark H L Schaaf April 27, 1999
120 REMark Gee Graphics #10, QL Today
172  stereo_flag = 0
174  spin_flag = 0
175  default_c_c = 65
177  default_disparity = -9
178  default_size = 24
245   IF  ans$=="q"  :  EXIT  demo_loop
370 MODE  4
380  WINDOW #0, 512, 50, 0, 206 : PAPER #0, 0 : INK #0, 4 : CLS #0
390  WINDOW #1, 256, 202, 256, 0 : PAPER #1, 0 : INK #1, 4 : CLS #1
400  WINDOW #2, 256, 202, 0, 0 : PAPER #2, 0 : INK #2, 4 : CLS #2
405  horz_to_vert = 256/202 : REMark the window size in pixels
410 horiz_calib = 107 : REMark makes it come out in mm on my monitor; adjust to suit your setup.
415  vert_scale = horiz_calib / (graspix*horz_to_vert)
420  horz_width = (vert_scale / (202-1)) * (256-1) * graspix
425  SCALE vert_scale, -horz_width/2, -vert_scale/2
430  SCALE #2, vert_scale, -horz_width/2, -vert_scale/2
432  c_to_c = horz_width
434  CLS
436  CLS #2
515    t_v(i,j) = t_v(i,j) * default_size/2
640  PRINT #0;"[T]ranslation","[R]otation","[S]caling","[O]riginal data"
650  PRINT #0;,"[A]nimation","Stereo [P]air",,,,"[Q]uit"
705   = 16 : Stereo_pair
710   = 17 : CLS #0 : STOP
830  IF ans_num = 20 : measure$=" Millimeters "
1165  IF stereo_flag : find_face_centers2  : sort_face_zs2
1270 :
2025  INK 4
2102  stereo_flag = 0 : CLS#2
2104  motion$ = "spin on "
2225   IF INKEY$(8) == CHR$(32) : PAUSE
2270 :
2280 REMark  Stereo_pair
2290 DEFine  PROCedure  Stereo_pair
2300 REMark  make copy for left screen
2310  stereo_flag  = 1
2320  DIM t_v2(4,3)
2330  DIM t_f_c2(4,3)
2340  DIM t_f_v2%(4,3)
2350  DIM zsort2(4)
2360  FOR i  =  1 TO DIMN(t_v)
2370  zsort2(i)  = zsort(i)
2380   FOR  j = 1 TO  3
2390    t_v2(i,j) = t_v(i,j)
2400    t_f_c2(i,j) = t_f_c(i,j)
2410    t_f_v2%(i,j) = t_f_v%(i,j)
2420   END  FOR j
2430  END  FOR i
```

```
2440 REMark default of 65 mm for c to c
2450 c_c = default_c_c
2460 x_offset =  c_c - horz_width
2470 SCALE vert_scale, (-horz_width/2)-x_offset/2, -vert_scale/2
2480  SCALE #2, vert_scale, (-horz_width/2)+x_offset/2, -vert_scale/2
2490 c_to_c = c_c
2500 REMark and default of -9 degrees for disparity
2510 disp_ang = 0
2520 disp_angle  = default_disparity
2530 disp_angl  = disp_angle -  disp_ang
2540 rotate t_v2, 2, disp_angl
2550 disp_ang  = disp_angle
2560  show_pair
2570 END DEFine Stereo_pair
2580 :
2590 REMark pair in respective windows
2600 DEFine  PROCedure  show_pair
2610 IF NOT(spin_flag):CLS#0
2620 CLS
2630 draw_solid t_f_v%
2640 REMark put in top tick to aid fusion
2650 LINE 0,(vert_scale/2)*.9 TO 0,vert_scale/2
2660 CLS#2
2670 draw_solid2 t_f_v2%
2680 LINE #2, 0,(vert_scale/2)*.9 TO 0,vert_scale/2
2690 INK#2,4
2700 IF NOT(spin_flag) :stereo_menu
2710 END  DEFine show_pair
2720 :
2730 DEFine PROCedure draw_solid2(f_v_array)
2740 REMark find_face_centers2 : sort_face_zs2
2750  FOR i = 2 TO DIMN(f_v_array)
2760   ii  = zsort2(i) : INK #2, nks(ii) : FILL #2, 1
2770   POINT #2, t_v2(f_v_array(ii,3),1) ,t_v2(f_v_array(ii,3),2)
2780   FOR j = 1 TO 3
2790    LINE #2 TO t_v2(f_v_array(ii,j),1) ,t_v2(f_v_array(ii,j),2)
2800   END  FOR j
2810   FILL  #2, 0
2820  END FOR i
2830 INK  #2,4
2840 END  DEFine draw_solid2
2850 :
2860 DEFine  PROCedure  stereo_menu
2870 CLS#0
2880 PRINT  #0,,,"["&CHR$(188)&' '&CHR$(189)&"]Center to center = "&c_to_c;" mm",
2890 PRINT  #0,"["&CHR$(190)&CHR$(191)&"]Disparity = ";disp_ang&CHR$(186)
2900 PRINT#0\,"[S]pin pair",,"[T]ransforms menu",,"[Q]uit"
2910  ans$=INKEY$(-1)
2920  ans_num  = CODE(ans$)MOD 32
2930  SELect ON ans_num
2940  = 0 : gap_change = 1 : Change_Space
2950  = 8 : gap_change = -1 : Change_Space
2960  = 16 : disp_change =  1 : Change_Disparity
2970  = 24 : disp_change = -1 : Change_Disparity
2980  = 17 :  quit_menu
2990  = 19 :  spin_menu
3000  = 20 :  transform_menu
3010  = REMAINDER : CLS#0 : stereo_menu
3020 END  SELect
3030 END  DEFine  stereo_menu
3040 :
3050 DEFine  PROCedure Change_Disparity
3060 default_disparity = default_disparity + disp_change
3070 rotate t_v2, 2, disp_change
3080 disp_ang  = default_disparity
3090 show_pair
3100 END DEFine Change_Disparity
3110 :
3120 DEFine PROCedure Change_Space
3130 CLS#0:
3140 default_c_c = default_c_c + gap_change
3150 x_offset =  default_c_c - horz_width
```

```
3160 SCALE vert_scale, (-horz_width/2)-x_offset/2, -vert_scale/2
3170  SCALE #2, vert_scale, (-horz_width/2)+x_offset/2, -vert_scale/2
3180 c_to_c = default_c_c
3190 show_pair
3200 END DEFine Change_Space
3210 :
3220 REMark quit menu
3230 DEFine  PROCedure quit_menu
3240 CLS  #0 : STOP
3250 END  DEFine  quit_menu
3260 :
3270 DEFine PROCedure find_face_centers2
3280  FOR i =  1 TO 4
3290   DIM sumxyz(3)
3300   FOR j = 1 TO 3
3310    FOR k = 1 TO 3
3320     sumxyz(k) = sumxyz(k)  + t_v2(t_f_v2%(i,j),k)
3330    END  FOR k
3340   END  FOR  j
3350 REMark then take the average of the three values
3360   FOR k = 1 TO 3
3370    t_f_c2(i,k)=sumxyz(k)/3
3380   END FOR k
3390  END FOR i
3400 END DEFine find_face_centers2
3410 :
3420 DEFine PROCedure sort_face_zs2
3430 DIM zsort2(4) : FOR i = 1 TO 4 :  zsort2(i) = i : END FOR i
3440 counter = 0
3450  REPeat short_sort
3460   swaps = 0
3470   FOR i = 2 TO 4
3480    IF t_f_c2(zsort2(i-1),3) < t_f_c2(zsort2(i),3) THEN
3490     swaps  =  swaps  +  1
3500     swap zsort2(i-1) , zsort2(i)
3510    END IF
3520   END FOR i
3530   IF NOT(swaps) : EXIT short_sort
3540  END REPeat short_sort
3550 END DEFine sort_face_zs2
3560 :
3570 DEFine PROCedure spin_menu
3580  CLS#0
3590 motion$ = "Spin around "
3600  get_axis
3610  INPUT#0;"ENTER Degrees per frame of animation ?",frame_shift
3620  CLS #0
3630  PRINT #0; frame_shift; CHR$(186);
3640  PRINT #0;" per frame around the ";ans$;"  axis"
3650  PRINT#0;"tap [space bar] to toggle Stop/Start motion"\"hold down [ESC] for stereo
menu"
3660  spin_pair
3670 END DEFine spin_menu
3680 :
3690 DEFine PROCedure spin_pair
3700 spin_flag = 1
3710  REPeat spin_loop
3720   IF INKEY$(10) == CHR$(27) : EXIT spin_loop
3730   rotate t_v,axis_num,frame_shift
3740   rotate t_v2, axis_num, frame_shift
3750   Stereo_pair
3760   IF INKEY$(2) == CHR$(32) : PAUSE
3770  END REPeat spin_loop
3780 spin_flag = 0
3790 stereo_menu
3800 END DEFine spin_pair
3810 :
3820 REMark end of listing Tetrahedra_Stereo_bas
```

The listing of GG#10 is meant to be merged with the listing of GG#8.

Next time? Persian Recursions, or oriental carpets on the QL.

CHEAPER
BUT
BETTER

With the recent release of STYLE-CHECK 3, the upgrading of the JUST WORDS! program range is complete.

All programs are now available in pointer driven versions and are fully mouse and high resolution screen compatible.

They are not only better, but now also cheaper.

SINGLE PROGRAMS:

£10 OR €15

THE COMPLETE SET:

£25 OR €37

Payment by sterling cheque or by giro transfer in euros to Netherlands Postbank number 4111942 (G.T. Wicks)

Also available from QBRANCH.

Geoff Wicks, 28 Ravensdale, Basildon, Essex SS16 5HU, United Kingdom. Tel: +44 (0)1268 - 281826

May I interrupt you...?

Jochen Merz

This text is aimed mainly at programmers; whether they program just for themself or for others.

You probably heard about external event features in SMSQ/E - they are not much used yet but during the last journey to England, Bernd and myself thought of some very useful features which could be implemented into many existing programs ... but first we have to agree on the usage of the event bits.

SMSQ/E does not provide any means of forcing programs to finish themself in a "nice" way. It is also not easily possible to tell programs "from outside" to abort very long, time-consuming actions.

What I mean when I say "nice finish" is, that a program shuts down itself if and when it finds the time. RJOB force-removes a job, which may result in data-loss (you can force-remove it even while saving, so you may lose the entire file), you may not have saved the current text or file, but the program cannot do anything about it, it just gets removed from the system. "from outside" means, that you can tell a different program to shut itself down properly, e.g. save the current file if it hasn't been saved, ask for overwrite if it has to, and THEN terminate itself. "outside" can be a menu which allows you to choose a program, or a BASIC command, or in assembler or any other language.

It should be easy to tell all currently running programs: shut down and save whatever has been changed (for example, if you want to turn your machine off).

Have you ever selected a full tree in QPAC2, which took ages because you did it in the root directoy of your harddisk - especially if you set QPAC2 to tricky sorting? No way to interrupt this? Or have you ever started a force-assembly of all files of a large assembler project by mistake? No way to interrupt it, unless you force-remove the job (which is not a nice way, the output file it currently works on will most likely be rubbish). A fiddly solution would be to set the priority of the job to 0, check that there are no open files and then force-remove it. Annoying, because you have to start the program once again, find the right parameters etc. etc.

Wouldn't it be nice to tell the program to stop its current action and go back into the main menu?

We think we have found a nice, easy-to-implement solution. Please think about it and reply if you find drawbacks, but with better solutions, please!
It is obvious that programs cannot always react tothis kind of request, only in certain situations (depending on the program), e.g. a MAKE program can check for interruption between every assembler run. The kind of signalling should therefore be an external "job" event, which is stored and can be checked by the program from time to time.

Just think about how easy it would be to implement small pointer-driven "Abort" windows - all they had to do is to send a single event to their parent job ... if you know how much work the "Abort" job in QD was then you will know what I mean. With an event-driven solution, adding "Abort" windows to many other programs would be quite easy because it would take next to no effort to do it.

SMSQ/E offers external events for some time. The events can be used by every job, even from BASIC. There are only eight bits available to signal events, so we have to find a way of dealing with them - some programmers may like to use the events also for other purposes.

Our first idea: we use bit 7 to flag the new job-events. If bit 7 is clear (easy test for negative) then the other bits can be used for user-defined purposes. If bit 7 is set, two of the other bits signal the two events mentioned above. However, this approach would require some tricky programming: some bits could be set to signal user-events, then bit 7 plus one of the events would be set to signal a job-event. Now the user-events pending would suddenly become job-events. Maybe an order of interpretation could be arranged, but it would be quite tricky, I think. I wonder if it is possible to define a means of interpretation which covers all possible bit combinations completely unambigously.

Next idea: we all agree on two of the 8 bits to be used for these events (terminate and abort current action). So which bits do we pick, which bits have been used by which software (to avoid those) because we do not want to create confusion if we send a "terminate yourself" to all jobs currently running in the system.
If we agreen on two bits, then the jobs should check the job-events from time to time and react accordingly.

How all this works, especially in BASIC, will be explained in the next issue - but we are waiting for your response first. We have ideas for our own programs and the events will quickly be implemented, provided we come to an agreement. Maybe you have another idea for another job event which could be useful to all of us.

So please think about it, write to us - a quick email to **SMSQ-event@j-m-s.com** will not take too much time, will it?

You and your Software - Just good Friends? - Part 2 "Menus"

Geoff Wicks

Do you remember buying your first QL and your first attempts at using Quill? Would it have been easier if, instead of having to remember L for Load, S for Save and P for Print, you had had to remember F5 for Load, F10 for save and Shift + F7 for print? In total 40 combinations of the function keys with Shift, Control and Alt? This is what you would have had to learn if, instead of a QL, you had bought a PC and WordPerfect. Quill came free with the QL, but WordPerfect would have charged you hundreds of pounds for the privilege of using what must be one of the most user unfriendly commercial programs ever written.
In this article we are concerned with menus in all their aspects including their length, content and placing. All these affect how easy your program is to use.
First length. The longer your menu, the more difficult people will find your program. Sub-menus improve things a little, but some users also find these complicated. It is a good practice to look at your menu items one by one and ask if each is essential. Let me give a few examples.
My program Solvit-Plus 2 has two dictionary commands, one for loading a dictionary in Solvit-Plus format and another for importing a plain text file. In

practice only one command is necessary. The later version Solvit-PLus 3 first checks the format of a file, and if is not in the Solvit-Plus format warns the user and asks whether he wishes to attempt an import.
Similarly, most word processors have separate load and import commands, but Perfection does not. In contrast Text87 has a complicated menu structure which makes it a slower and more difficult program to use. There are separate load and import commands, and then an import submenu with a choice between ASCII and Quill formats. Whenever I use Text87, I am continually wondering if the menu structure of the program could have been much simpler.
Programming is a logical activity and sometimes when we think logically we miss the simple and obvious. When I was designing Style-Check, logic said I would need a load command followed by a submenu to choose between Quill, Perfection, Text87 and ASCII formats. It was only when I was writing the program that I realised the different file formats were easy to detect automatically. I needed just one load command.

Friendly Software Rule: Keep your menus as simple as possible.
When you have decided what to have on your menu, go through the items one by one and assess how often that item will be used. In a word processor loading, saving and printing are among the most used commands. In most word processors these commands are the first on the menu bar. In contrast the help screens may only be used when the user is learning the program. Similarly configuration commands are likely to be used only occasionally, usually when the program is new. Both of these commands have relatively low priority in the menu list.
If your program has a print routine, you will need menu commands for printing, setting the baud rate and the printer output device, but do not place these together. Print will be used regularly, but the other two perhaps only once in the life time of the program. Place them by the configuration command.
Press F3 in Perfection or Text87 for good examples of setting priorities on menu lists, and in Quill for a bad example.

Friendly Software Rule: Place regularly used items at the start of the menu.
The next thing you have to decide is the placing and form of your menu. The most important thing is that your menu items should stand out from other items on the screen. They can be a different colour, a different size, a different font or in a special menu box or window. Good examples of this are Quill and to a lesser extent Perfection. An appallingly bad example is Text87. Poorly planned, barely legible menus are one of the commonest faults in QL software.
My recommendation for pointer programs is that each menu item is placed in its own box. This makes it easier for the user to get the pointer in the correct place.
In my program QL-Thesaurus I went a stage further and put the main three search commands in a red stippled box to draw attention to them.

Friendly Software Rule: Make your menu items stand out on the screen.
Finally your menu items should be clear and unambiguous. Most programs no longer use the function keys, but a mnemonic. Where possible the mnemonic should be the first letter of the menu item. Emphasise this letter by using a capital letter, a different colour or in pointer programs underlining.
If you have two menu items with the same first letter, try to find an alternative name for one of them. In some of my pointer programs Dictionary has been changed to word List because I needed the D for page Down. Be careful, however. One reviewer rightly criticised me for using "Name of dictionary" in a program instead of "Load dictionary". When I was developing the program I needed the L for another command, which I later replaced with a different routine. I forgot to change "Name" back to "Load".
If you have to use a letter other than a first letter for the mnemonic, do not use any old letter, but think carefully which is the most suitable. In the print submenu of Text87 there are three commands beginning with P, Print, Preview and Page. Obviously print itself must have the P. The other two are preView and paGe, which in both cases is a good choice of alternative letter as these letters are emphasised when the words are spoken. In comparison pReview or pAge would have been inappropriate.

Friendly Software Rule: Make the keypress used to implement a menu item clear and unambiguous.
The more daring programmers now use icons as menu items. There are pros and cons in doing this, which I shall discuss in a later article.
Next time: System compatibilities.

TF Services

superHermes

A major hardware upgrade for the QL

- All Hermes features (see below for list) PLUS full 19200 throughput on ser1/ser2 not affected by sound
- IBM AT keyboard interface (plus foreign drivers)
- HIGH SPEED RS232 industry standard two-way serial port. 4800cps throughput (supergoldcard - qtpi - zmodem) at 57600bps
- THREE low speed RS232 inputs (1200 to 30bps) Driver for SERIAL MOUSE supplied. Other uses include RTTY/ graphics tablet etc
- THREE spare I/O lines (logic) with GND/+5V
- Capslock/scrollock LED connector
- Turbo/keylock connectors
- 1.5k user data permanently storeable in EEPROM

All this on a professional board about twice the size of the 8049 co-processor it replaces

Cost (including manual/software)	**£90** (£92/£87/£90)
IBM AT UK layout Keyboard	**£22** (£24/£23/£27)
Serial mouse	**£11** (£13/£12/£14)
Capslock/scrollock LED	**£1** (£1.50/£1/£1.50)
Keyboard or mouse lead.............	**£3** (£3.50/£3/£3.50)
High speed serial (ser3) lead	**£4** (£4.50/£4/£4.50)

Hermes available for £25 (£26/£24/£27) **Working ser1/2 and independent input, debounced keyboard & keyclick.**

superHermes LITE

All Hermes features (see above) + an IBM AT keyboard interface only. Entry level superHermes.
Cost (incl keyboard lead)...**£53** (£55.50/£51/£53.50)

Minerva

MINERVA RTC (MKII) + battery for 256 bytes ram. CRASHPROOF clock & I^{2}C bus for interfacing. Can autoboot from battery backed ram. Quick start-up.

The ORIGINAL system operating system upgrade

OTHER FEATURES COMMON TO ALL VERSIONS

DEBUGGED operating system/ autoboot on reset of power failure/ Multiple Basic/ faster scheduler- graphics (within 10% of lightning) - string handling/ WHEN ERROR/ 2nd screen/ TRACE/ non-English keyboard drivers/ "warm" fast reset. V1.97 with split OUTPUT baud rates (+ Hermes) & built in Multibasic.

First upgrade free. Otherwise send £3 (+£5 for manual if requd). Send disk plus SAE or two IRCs

MKI...£40 (£41/£40/£43) **MKII...£65** (£66/£63/£67)

QL REPAIRS (UK only)

Fixed price for unmodified QLs, excl microdrives. QLs tested with Thorn-EMI rig and ROM software.

£27 including 6 month guarantee

QL RomDisq

Up to 8 mbyte of flash memory for the QL

A small plug in circuit for the QL's ROM port (or Aurora) giving 2, 4 or 8 mbytes of permanent storage - it can be thought of as a portable hard disk on a card, and reads at some 2 mbytes per second.
Think of it - you could fully boot an expanded QL, including all drivers/SMSQ etc off **RomDisq** at hard disk speed with only a memory expansion needed.

2 mbytes RomDisq............	**£39** (£41£37/£40)
4mbytes RomDisq..............	**£65**(£66/£63/£67)
8 mbytes RomDisq..........	**£98** (£100/£95/£99)
Aurora adaptor......................	**£3** (£3.50/£3/£4)

MPLANE

A low profile powered backplane with ROM port

A three expansion backplane with ROM port included for RomDisq etc. Aurora can be fitted in notebook case and powered off single 5V rail - contact QBranch for details. Two boards (eg Aurora and Gold Card/Super Gold Card/Goldfire fixed to base. Suitable for Aurora (ROM accessible from outside) & QL motherboard in tower case. Specify ROM facing IN towards boards, or OUT towards back of case.

Cost............................£34 (£36/£33/£35)

I2C INTERFACES

Connects to Minerva MKII and any Philips I^{2}C bus

Power Driver Interface **16 I/O lines with 12 of these used to control 8 current carrying outputs (source and sink capable)**

2 amp (for 8 relays, small motors)	**£40** (£43/£38/£44)
4 amp total (for motors etc).............	**£45** (£48/£43/£50)

Relays (8 3a 12v 2-way mains relays (needs 2a power driver).. **£25** (£28/£23/£27)

Parallel Interface Gives 16 input/output lines. Can be used wherever logic signals are required.. **£25** (£28/£23/£27)

Analogue Interface **Gives eight 8 bit analogue to digital inputs (ADC) and two 8 bit digital to analogue outputs (DAC). Used for temperature measurements, sound sampling (to 5 KHz), x/y plotting** ... **£30** (£31.50/£29/£30)

Temp probe (-40°C to +125°C).........	**£10** (£10.50/£10/£11)
Connector for four temp probes	**£10** (£10.50/£10/£11)
Data sheets	**£2** (£2.50/£2/£3)
Control software & manual (for all I/F)..	**£2** (£2.50/£2/£3)

QL SPARES

Keyboard membrane	**£12** (£12.50/£12/£13.50)
1377 PAL..	**£3** (£3.50/£3/£4)
Circuit diagrams...	**£3** (£3.50/£3/£4)
68008 cpu or 8049 IPC	**£8** (£8.50/£7.50/£9)
8301/8302 or JM ROM or serial lead .	**£10** (£11.50/£10/£11)
Power supply (sea mail overseas)...........	**£12** (£17/£16/£21)

Other components (sockets etc) also available

Prices include postage and packing (Airmail where applicable). Prices are: UK (EC/Europe outside EC/Rest of world). Payment by cheque drawn on bank with UK address, debit card/Mastercard/Access/Eurocard/postal order or CASH! (No Eurocheques). SAE or IRC for full list and details

19 Dec 98

VISA

29 Longfield Road, TRING, Herts, HP23 4DG
Tel: 01442-828254 Fax/BBS: 01442-828255
tony@firshman.demon.co.uk http://www.firshman.demon.co.uk

MasterCard

BYTS OF WOOD

SAW POINTS OFFCUTS AND SNIPPETS

Ramblings of an Unreasonable Man

First of all a big thank you to those people who attended that Hove show at the end of February and an even bigger thank you to those who took the time and effort to give a talk. Mark Knight did a wonderful exposition on fractals, Geoff Wicks demonstrated the new version of his Style Check program, Keith Mitchell gave a demonstration of the MinisQL construction and Rich Mellor stepped in at the last moment (when Jonathan Hudson dropped out) to give a quick talk on the progress of RWAP software. All in all an enjoyable show for me, and I hope for all who attended.

Thanks again to all who helped me to organise it.

Reasons to be Cheerful - Again.

There are many reasons why I prefer to have most of my files on QL based hardware. At the Hove show a customer brought along a nice 300 MHz laptop with a 1024 x 768 pixel display and wanted me to install QPC on it for him. 'No problem', I said because I have installed many copies of QPC and had no problems before. This one was different.

It seems he had a virus on his system which, as soon as I tried to restart the computer, deleted the COMMAND.COM file. Those of you who have PCs will know that this file has all of the basic DOS commands in it and, because there is no ROM on a PC, this means that you can do absolutely nothing. You cannot delete, copy, run a directory or anything. This laptop occupied about 4 hours of Steve Hall's time at the show and, in the end, we had to re-install Windoze.

OK, you may think, that will solve it. Nope. Windoze 95 does not have a screen driver which will handle 1024 x 768 on an LCD screen and the display would not go beyond 640 x 480 and looked dreadful. We wound up having to take it back to the shop and install Windoze 98 after a full hard disk format and a half hour on the internet downloading the necessary drivers from the maker's website.

I am very glad that my Aurora systems have solid, un-infectable operating systems on ROM on the main board. Even restoring my system from the hard disk crash mentioned in the last issue did not take as long as it did to fix this one laptop.

Reasons to Engage Brain Occasionally

I mentioned, in the last issue, a couple of customers who had problems with their systems but managed to solve most of them themselves. We all do something stupid sometimes and we often compound the problem by asking other people questions which, either as we say them or as we slip the letter into the post box, we suddenly realise are dumb. Some of the PC users that come into my shop abuse that privilege.

A recent case was a man who bought an Olivetti Inkjet printer from me. First he told me that it would not print so I had to tell him to install the driver. This solved the problem and he was quite happy for about three months. Then he came back. 'It has stopped printing again', he said. I started to go through the usual problems when he said that his dad took the top off the computer and it stopped working so he bought another one. I asked him if he had installed the driver on the new machine and he assured me he had. PCs have an amazing facility which allows you to play around with the BIOS and do useful things such as switching off the parallel ports etc but I did not want him in the BIOS so I said bring the printer and computer in and I would look at it.

It was only when he brought it in that I thought to ask if there were any lights on on the printer. This was actually to see if the power was getting through to it but it brought the response,

'Oh yes, these two flash on and off'

'What does that mean ?', I asked.

'I don't know', he said.

'Well look it up in the manual'

Guess what? It had run out of ink!

Reasons to Listen

I received two communications recently and both of these had a bearing on the way in which the everyday user sees the software that he has to use. I did discuss this with Stuart Honeyball a while ago and, in principle, I agree with him that software should be fairly transparent to the user and the idea that he should have to wade through reams of paper to be able to use it is not a good thing. Of course this does depend, to a degree, on the complexity of the actions that the program is going to take.

The first of these letters came from a subscriber to this magazine who decided not to re-subscribe because he could not understand how to use the QL and had moved to a PC whose programs had contextual help files which pop up at many stages in the process and explain things. In many ways this is a good way of doing things and I would like to see more of this in our programs. Jochen's latest menu_rext used in conjunction with SMSQ/E has the ability to pop up small explanations along these lines and both ProWesS and PD3/S utilise a system whereby each command menu had a help file attached to it so you could go directly to the help you needed. This is something we should see more of.

Reasons To be Humble

The second communication was by email. Another person who did not want to continue with QL accused me of rejecting his programs. I did this because, in my opinion, they were unfinished - lacking a front end. This is a matter I have raised with other people, mostly PD or Shareware writers. These people do all the clever stuff. They write or port from other systems, great applications many of which are very useful but, in order to use them, you have to type in a list of switches and file names in a set format.

When asked to provide a user-friendly front end to these they reply, 'Well the information is there, you can write it yourself'.

Ok maybe I can (and I have often done this) but some people cannot and unless these people are helped out by programmers like Thierry Godefroy they are barred from using all of these utilities. Without Thierry's excellent Archivers Control Panel a large amount of users are effectively disbarred from using programs like Zip or Unzip. This is not because the actual use of the programs is hard but the manuals provided are all but un-inteligable and the real information is buried under reams of programmer-speak.

The 'Well I can do it and if you can't then I cannot be bothered with you' attitude is a supreme kind of arrogance and intellectual snobbery and we can do without it.

Reasons to BBS

While I am on the subject of PD writers I have noticed that there has been a remarkable lack of programs arriving on the QL BBS (Bulletin Board System).

This is not, I think, because the programs are not being

SBASIC / SuperBASIC REFERENCE MANUAL

If you have ever felt the need to program the QL in SuperBASIC, this is the book for you. With detailed descriptions of all of the keywords available to the programmer, including notes on each of the different implementations and warnings as to where incompatibilities may lie, coupled with 100s of examples, any discerning programmer cannot do without this book. Detailed appendices deal with hints on programming, using device drivers and networks, getting the most out of the QL's colours and display and even using multiple copies of Basic.

A detailed index program is supplied, which allows you to look up what you are trying to do in SuperBASIC and then tells you which parts of the Reference Manual cover this topic. A disk full of examples from the book is supplied, as well as a disk with all of the public domain toolkits covered in the Manual.

Release 1 of this top selling QL book is now available. See the review in Volume 3 Issue 6 for brief details.

Be a Feeder rather than a Chicken!!

Price: £40 plus post & packing

UPDATE SERVICE

Existing owners of the Reference Manual (purchased before 28/2/99) need to update now!!

Get the Release 1 Update for £6 or pre-order the Release 2 update at the same time for £10 in total.

You need to keep up to date to get the most out of your computer!!

HOW TO ORDER:
Contact: Rich Mellor, 26 Ashenhurst Road, Russells Hall, Dudley, West Midlands DY1 2HH
Tel: 01384-350043
Credit Card orders can be accepted via Qbranch. Otherwise payment must be in cheque Sterling payable to *R. Mellor*

written but because many of the authors have their own websites on which they post the programs. This is Ok if you assume that all the people who want the programs have internet access and they can work out that there are new programs available there.

I got my first modem a few years ago and I was a regular visitor to the BBS trawling for new and unusual items. Even though I am now a regular user of the internet I still use the BBS to upload my adverts to both Tony Firshman and Jochen. It seems annoying to me to do the ad on my QL and then transfer it to the PC in order to send it when there is a perfectly good QL way of doing it.

QL Today offered PD and Shareware writers a space to announce their new programs but so few of them took it up that we never ran it as a regular feature. It seems absurd to me that people who write QDOS/SMSQ programs put it out on a media not accessible by QL hardware (and often with instructions in a format unreadable by QL users).

This was illustrated by the ql-users mail group on the internet recently. Arnould Nazarian popped up to talk about the Palm Pilot hand held computer. Someone else mentioned that Joanthan Hudson had written/ported some programs which allowed QDOS to communicate with the Palm Pilot O/S and loads of people popped up to say 'What? Where can I get it? What does it do?' etc.

Come on guys, I know the BBS cost a bit more to access but put your stuff up there and let QL users with no internet access have a look at it.

On a QL BBS, when you log on, you get a list of new files. On the internet you have to go to each site in turn and look to see what is new. Thierry Godefroy (this is fast turning into the Thierry Godefroy fanzine here) does at least put everything he can find on his site but most of the other PD writers don't. Don't let us turn into a little fragmented group who only post stuff on our own sites - publish and be downloaded !

Reasons to be Clean and Shiny

A while ago Dilwyn suffered from the same problem that a few other people have had - namely, when they switch on they get the mystic words 'QL ROM Not Recognised - Contact Miracle Systems'. This is doubly annoying because if you do contact Miracle Systems they can't tell what the problem is without having your system back for a complete test (actually Stuart just passes these problems on to me so bypass the middle man and tell me first).

One thing to check before pouring more cash into the Telecoms industry is the ROM socket. When you switch the machine on it goes through a set sequence of actions checking the RAM, the extension slot and the ROM slot before finally firing up the system. If the ROM slot is occupied then it will read the ROM and put any header in its code on screen.

If, however, the ROM slot does not read properly or the code in the ROM will not read into the system it will stop right there and not go any further. Since reading the ROM slot is one of the first things that the system tries to do you will get the message above. It does not mean it cannot read the QL ROM itself but the effect is the same.

If you get this message try removing any ROM you have plugged in and cleaning the contacts with a pencil eraser. If the problem persists remove the Gold/Super Gold Card and GENTLY clean the pins of the connector on the QL end. These little things could solve the problem and put you back on the road again.

Reasons to go TRA la la

There are many ways to get confused in the wide world of computing. Ask someone about something complicated that he knows a lot about and prepare to be assaulted with a mass of information, a lot of which will emulate Concorde and whizz over your head at a vast rate of knots. You can also ask an expert about something that he thinks is simple. The resulting reply may fulfil three criteria:

a) short,

b) delivered with an unspoken implication of 'You mean you don't know about that ?' and

c) unenlightening.

Years ago when I started to use QSpread I found that the pound sign did not come out as a '£'. Jochen said I was using the wrong translation table but I had never used any translation table at all and had no idea what I should do. SMSQ/E made things simpler because 'TRA' became a new KEYWORD but I still did not understand it.

When Jochen and Bernd Reinhardt put their heads together to produce the new, and vastly improved, QSpread 99 (big ad here) they emailed me copies of the beta versions to test. Jochen wrote a new printer filter/driver and it still did not print the required pound signs. I asked Jochen about this and finally the light dawned. If you want to get programs like QD and QSpread to print the pound sign add the line 'TRA 3' to your BOOT file. This is quite simple and '£' signs appear miraculously.

The next problem is that programs like Text 87 which always produced the required printout now produced a wrong character. If you change the printer port in these programs to 'PARd' everything returns to normal because they then ignore the translation table.

All this is probably 'old hat' to many of you but I had somehow missed the vital piece of information which I needed to make it work and I suspect I am not alone.

Reasons to Hang out the Bunting

The big highlight of March, for me at least, was receiving the first two SMSQ/E ROMs for my Q 40. Tony Firshman had been very pessimistic about progress for a few weeks since a lot of the code he had received had not worked. He sent the ROMs to me and, having plugged them into my Q 40 (one of the first production boards) up came SMSQ/E! Tony had a faulty component on his board which was no problem when firing up QDOS Classic but caused a big problem to SMSQ/E.

It is not the full shilling yet but there is not much change left to juggle before it is - Whoopee here come the fast QLs at last. At the time of writing I have a basic QPAC 2 system up and running with Cueshell, QD, FileInfo 2 and a lot of other stuff running and, by the time you read this they will be available at a QL show near you.

My joy at getting this project off the ground is not just a financial one since the Q 40 itself is not a big money spinner. The real reason that the Q 40 is such a big event is that it is the first total QL replacement since the Thor. The extra things that the Q 40 offers will allow programmers to do a lot more than any other piece of QL hardware and they thus hold out the carrot of better programs. After years of hard work by Peter and Claus Graf and another effort by Tony Tebby, Mark Swift, Tony Firshman and Richard Zidlicky (who has ported LINUX to the Q40) the thing is finally a reality and we have a real alternative, advanced, system - and not a wisp of vapourware.

Of course each new piece of hardware adds a few little problems to the general melting pot and there are going to be programs which will not run and others which don't do what they are supposed to do. As a lighting engineer friend of mine once said 'There are no problems - only new opportunities'. The old 'One small step for man one giant leap for QDOSkind' misquote.

And Finally...... Reasons for Shady Mice

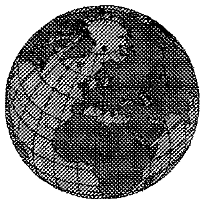
I had a brain teaser in the shop recently. After a discussion with Steve Hall about mice and the relative merits of spending an extra three or four pounds to get one which said 'Logitech' on it I found the the cheapish ICE mouse which I had been using would only make the cursor go up and down and not from side to side. I immediately suspected some merry prankster had been fooling with the device but could find no evidence of this.

I took it apart and cleaned it. No better. This was very frustrating because I was trying to do the end of the day's accounts on QSpread and a mouse is pretty essential for that. I gave up and went home. The next day the mouse was fine again. Two days later it happened again. I took the mouse apart and examined the wiring. I took the MinisQL apart and examined that too. Nope - no answer there.

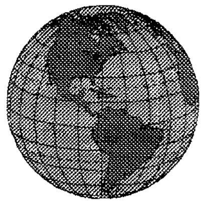
After a period of functionality the mouse reverted to its directionalist tendency (maybe, I thought, this is some sort of political mouse statement. Centreist leaning if there is such a concept.). Finally, last week, it happened earlier in the afternoon and I decided to have a cup of tea and contemplate the problem. I had been through all of the usual ideas about components overheating after being on all day. I had switched the computer off for a while, I had changed the serial cables. I put the cup down on the desk and moved the mouse. It went sideways - for a while.

Then I saw the light! Or rather the mouse had seen the light earlier on. Late afternoons, if the sun is out, it shines directly on the desk where the mouse sits. It is an optical mouse. The sun shines through the mouse buttons in the front of the mouse and it is blind in one eye the reason it worked when I put the cup down was because the cup shaded the mouse. Simple really but very hard to diagnose.

■



The QL Show Agenda



Regular QL Meeting - (NL) Eindhoven

Saturday, 26th of June 1999!

Held at its usual Venue: St. Joris College.

The meeting starts at 10am and ends at 4pm.

QBranch with the Q40 and Jochen Merz Software with QPC 2 will be present!

QL Shows - (F) Paris

Sat., 18th of September

The date is fairly definite but the place was not known at the time QL Today went to the printer. More details in the next issue - watch this space!

Quanta workshop - (GB) Byfleet

Sun., 3rd of October 1999

As usual in Autumn, the Byfleet Quanta workshop.
There is still plenty of time, so details will follow in the next issue.

QL Meeting - (A) Heidenreichstein

Sat./Sun., 9./10.th of October 1999!

Same venue as last time: Gasthof Nöbauer.

Definitely a very nice, social event too!

Austrian meetings are always more than "just" a QL meeting. This time, a visit to a narrow gauge railway and a model train exhibition is planned. More details in the next issue of QL Today