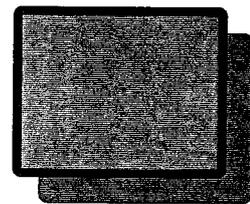
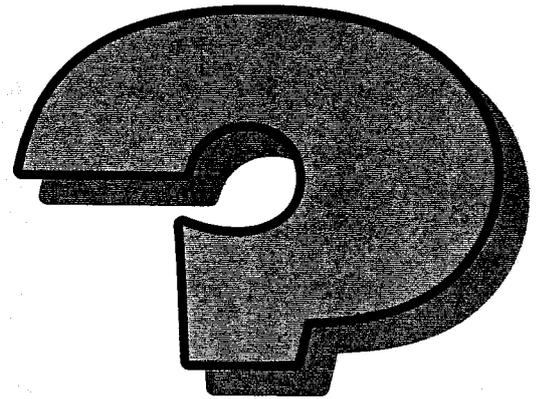


QL Today

Volume 15
Issue 2
Dec. - Jan.
2010/11

ISSN 1432-5454

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



The
Future
of
QL
Today?

www.QLToday.com

Contents

3	Editorial	
4	News	
7	25 Years - Part 5	Tony Tebby
16	QL Queues	George Gwilt
17	Goodbye Paper? - Part 2	Geoff Wicks
19	XCHANGE	David Denham
23	Optical Illusions	Stephen Poole
24	QL Today, The PDF Version	Norman Dunbar
27	The quest for another lost treasure	Urs König
28	A Surprising Thing	George Gwilt
34	RS232 Monitor Project	Ian Burkinshaw
37	Letter-Box	
38	Last-Minute News from Dilwyn	

**The deadline for
the next issue
is the 12th of
February 2011**

Advertisers

in alphabetical order

Jochen Merz Software	33
Quanta	15
QuoVadis Design	29

QL Today

ISSN 1432-5454

German office & Publisher:

Jochen Merz Software Tel. +49 203 502011
Kaiser-Wilhelm-Str. 302 Fax +49 203 502012
47169 Duisburg email: smsq@j-m-s.com
Germany email: QLToday@j-m-s.com

Editor:

Geoff Wicks Tel. +44 1332 271366
Flat 5b email: gtwicks@btinternet.com
Wordsworth Avenue email: QLToday@j-m-s.com
Derby DE24 9HQ
United Kingdom

Co-Editor & UK Office:

Bruce Nicholls Tel +44 20 71930539
38 Derham Gardens Fax +44 870 0568755
Upminster email: qltoday@q-v-d.demon.co.uk
Essex RM14 3HA email: QLToday@j-m-s.com
United Kingdom

QL Today is published four times a year, our volume begins on beginning of June. Please contact the German or English office for current subscription rates or visit our homepage www.QLTODAY.com.

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. We prefer ASCII, Quill or text87 format. Pictures may be in _SCR format, we can also handle GIF or TIF or JPG. 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.

QL Today reserves the right to publish or not publish any material submitted. Under no circumstances will **QL Today** be held liable for any direct, indirect or consequential damage or loss arising out of the use and/or inability to use any of the material published in **QL Today**. The opinions expressed herein are those of the authors and are not necessarily those of the publisher.

This magazine and all material within is Copyright 2009 Jochen Merz Software unless otherwise stated. Written permission is required from the publisher before the reproduction and distribution of any/all material published herein. All copyrights and trademarks are hereby acknowledged.

If you need more information about the UNZIP program which is used by our BOOT program to unpack the files, we suggest that you visit Dilwyn Jones' web site where you find more information about lots of interesting QDOS software and INFOZIP at <http://www.dilwyn.uk6.net/arch/index.html>

Editorial

by Geoff Wicks

The last issue of QL Today caused a stir that we had not expected. Our one page explanation of why it was not possible to have an electronic edition of the magazine led to a lengthy discussion on the QL Users email group and experiments to solve the problems. The experiments have reinforced what we wrote.

The good news is that Urs König has produced an electronic edition of an early issue of QL Today. The bad news is that it has a file size of 50Mb, which makes it suitable for archiving purposes, but not for the distribution of the current edition.

Norman Dunbar has also been working hard to solve the problems and he describes some of his findings in this issue of QL Today. We first provided him with a sample Calamus file and later with sample articles in the form that Jochen receives them. In a private email he wrote,

"Laying out a magazine from people's contributions is a nightmare! I've had a quick attempt at converting the files you sent me into a PDF magazine. (Expletive) it's hard work!"

Articles to QL Today arrive in many formats, and not always conforming to the guidelines we print on page 2. We frequently receive articles in Microsoft Word format with illustrations and diagrams embedded in the text. I have software to extract the pictures but have to use screen capture for some diagrams, which, for quality reasons, we prefer not to do. The articles have to be sub-edited, spellchecked and often reformatted. Both Dilwyn and I have written programs for automatic reformatting, but these are not suitable for articles containing assembler code where the reformatting has to be done manually line by line.

All this is simple in comparison with the layout work. If you have no experience of producing a magazine it is hard to imagine how difficult laying out can be. Indeed I would probably use a stronger expletive than Norman to describe it.

It is not simply a matter of putting one article after another. You have to get the paging right and there are what are known as "orphans" and "widows". This is where a paragraph is too long to fit in a column, and one line remains at the bottom of the column or flows into the next column. You then have to adjust spacing or the size of images to create a teeny bit more space. The reverse of this problem is a small white space at the end of column that has to be filled.

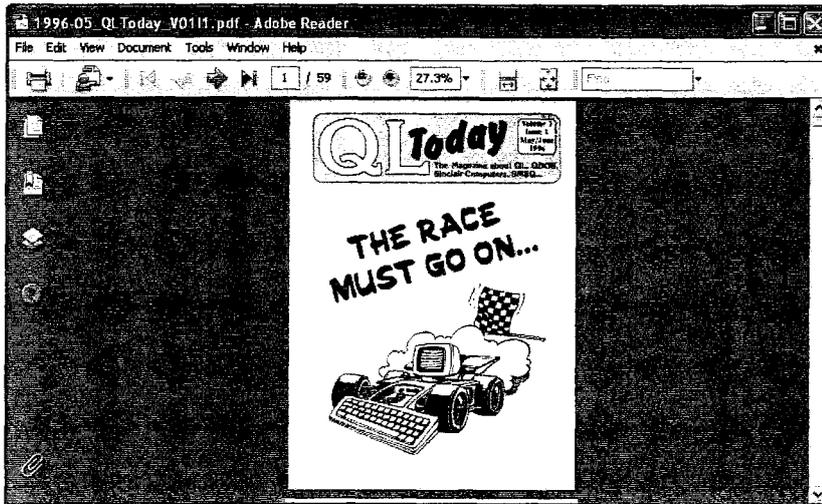
Nor is it just a problem of orphans and widows. Similar things happen with subheadings or with illustrations where the solutions are not so easy, especially when correct placing is essential for the reader to follow and understand the text. Sometimes Jochen has to change the order of news stories to solve problems like these. Another, but fortunately rare problem, is when a news item becomes outdated while the magazine is in production. Then it has to be updated and rewritten with the same number of words so that it still fits in the allotted space.

It is precise, time consuming and painstaking work, and to complicate matters you are working with a computer screen. You continually have to flip between a full page image where you cannot read the text or a part image to see details. It soon becomes very tiring work.

The production of the QL Today is a complex process. We are not just being lazy or difficult when we say we cannot produce an electronic edition. There are many problems still to be solved.

First Electronic QL Today

| column arrangement on some pages - and then use the text to build a new PDF file that is all of the above."



Norman did some tests on a file provided by **Jochen Merz**, but a few days later had to report disappointing results:

"I have downloaded a trial version of the Calamus program for Windows and I've run it with a small test file provided by Jochen.

- I also installed a number of PDF "printers" and each time I try to create a PDF the 15 pages start printing and then the CPU hits 100% for a couple of hours. This is totally unacceptable in my opinion.

QL Today found itself making the news in the last few months. Shortly after the last issue, detailing the problems of publishing the magazine in electronic format, reached the readers, a lengthy discussion took place on the QL-users email list. Several suggestions to solve some of the problems were made, further experiments and tests for electronic publication were done and an electronic version of the first ever QL Today published.

In the last issue the editor of QL Today outlined the main problems with going electronic. QL Today is produced using Calamus, an emulated Atari program that does not use PC fonts. Each page has to be produced as a bit map image leading to PDF file sizes that are too large for electronic publication.

Norman Dunbar commented:

"I realise that there are PDFs and there are PDFs and Geoff's comments about each page having to be a bitmap will indeed make the issues massive, but I'd be pretty sure that there's bound to be a way of getting actual text into the PDF"

He also laid down the conditions for a PDF file:

"To me, a PDF file containing magazine text, or book text etc, should be:

- Searchable - bitmap pages are not.
- Copy & paste'able. Again, bitmap pages are not.
- Small!

So, we only need the scanner to get the pages as images for an OCR to extract the text - I'm not sure how an OCR would treat the two

- Printing one page to PDF works, but takes far too long as well.
- The PDF created is a massive file, for one page it is about 4.5 Mb which implies a certain amount of bitmapping is going on. This too is unacceptable.
- The full 15 pages eventually filled my disc!
- There do not appear to be any programs that can extract the contents of a Calamus data file (*.CDK) although the versions of Scribus greater than 1.3.5 are supposed to be able to import Calamus Vector Graphics files, version 1.3.7 doesn't appear to be able to display them or print them to PDF. Sigh!"

However Norman started tests to see if he could produce a PDF version of QL Today using other programs and reports some of his results elsewhere in this issue.

Wolfgang Lernerz suggested a different approach of scanning the paper copy of each issue that he estimated would take about 6 minutes. Just over a week later **Urs König** tried just that and produced the first electronic edition of QL Today. With permission an electronic version of the first issue of QL Today can be downloaded from:

http://www.cowo.ch/downloads/1996-05_QLToday_V0111.pdf

Although the PDF is searchable the file size is just over 50Mb, which is too large for distribution to readers. However some readers have indicated they would like the magazine to remain on paper, but with an electronic facility for archiving. Urs estimated that a complete archive of QL Today would fit into a DVD compared with bookshelf space of 26cm x 21cm x 20cm for the paper copies.

QL Today has still been unable to find a way of producing a PDF edition of a manageable size, and in the discussion on the QL users group it became clear that several people have difficulty in understanding the reason for this. Although Calamus is an Atari program the version that QL Today uses is an emulated PC version. However the emulation means that although it is run on a PC it does not use PC fonts. For this reason each page is produced as a bit mapped image. For QL Today to be able to produce a manageable PDF file some means must be found to either extract both text and images, or to import the complete Calamus file into another program that recognises both text and image as separate entities.

Norman Dunbar thought that Calamus files could be imported into Scribus but discovered that it could import Calamus Vector Graphics files, but not Calamus text. **Tobias Fröschle** suggested an alternative would be a plug in module for Calamus, Bridge 6 pro. Unfortunately there is no demonstration version of this program and QL Today does not know if it can produce PDF although this is thought likely. Should it be able to do so there remains uncertainty over the quality and nature of the PDF files. Can the module handle Calamus fonts to produce searchable text and manageable file sizes or will it still produce large bit map images?

QL Today is not in a position to fund the purchase of an expensive module of which there is so little known, and any help from readers would be appreciated.

NEW Q-emuLator

Daniele Terdina has released two new versions of Q-emuLator, one for Windows and the other for Mac OS. He writes:

"Version 3.0 of Q-emuLator for Windows is now available (<http://www.terdina.net/ql/Enter.html>).

This is a paid update (except for users that registered in the last six months - they will receive a free update). The unregistered version (with limited features) can still be used for free.

New features in version 3:

- The main window can be resized to make the QL screen bigger.
- Compatibility improvements.
- Precise QL speed emulation.
- Dot matrix printer emulation.
- Mount .ZIP files as read-only disks.
- Use .QLPAK single-file QL software archives.
- Smart full screen upscaling.
- Access microdrive images and floppy disk images.
- QL Sampled Sound System.
- Improved display emulation when running on Windows 7 (and full screen mode fixed).

Q-emuLator 1.0 for Mac OS X is available at <http://www.terdina.net/ql/EnterOSX.html>.

It only works after getting a registration code, but I can send temporary registration codes to people interested in trying it (email danieleterdina@hotmail.com and please include your full name).

Gwilt Updates

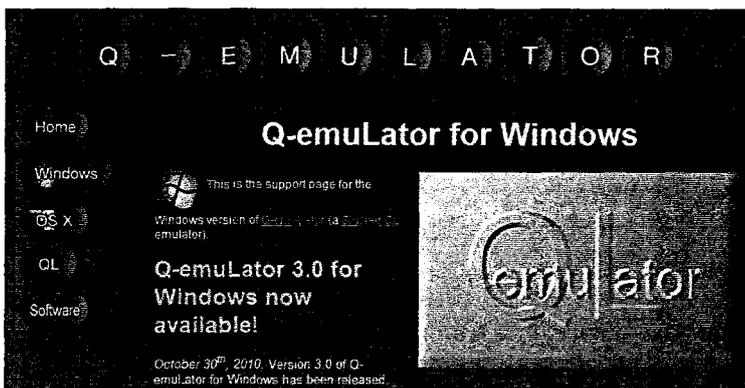
George Gwilt has announced updates to two programs:

"SETW , which produces window definitions for programs using TurboPTR, C and Assembler, has been upgraded.

1. The maximum space used for storing sprites, blobs and patterns has been trebled.
2. The option to request clearance or not of main windows and sub windows when they are drawn or redrawn has been added.
3. The option to request for each main window and for each application sub window that the arrow keys be disabled from moving the pointer has been added.

Tptr_bas, which is the crucial part of TurboPTR, has been altered in consequence of 1 above."

The altered programs are available at <http://web.ukonline.co.uk/george.gwilt/>



Retro Show?

Urs König has made it possible to follow the presentations at last year's Lucerne show by posting the entire proceedings on YouTube. In his own words: "Finally the video footage has been catalogued, ordered, brushed up (video noise, audio volume), edited and became a final directors cut. Then the 24 videos had to be uploaded to my YouTube channel. All

this took much more human and processing power than expected. The session videos are online for public viewing. Just browse down the events webpage until you'll see the fire and enjoy the easy navigation through the video playlist. There's more than 5 hours of video footage for you to enjoy."

<http://tinyurl.com/ql-mac-show>

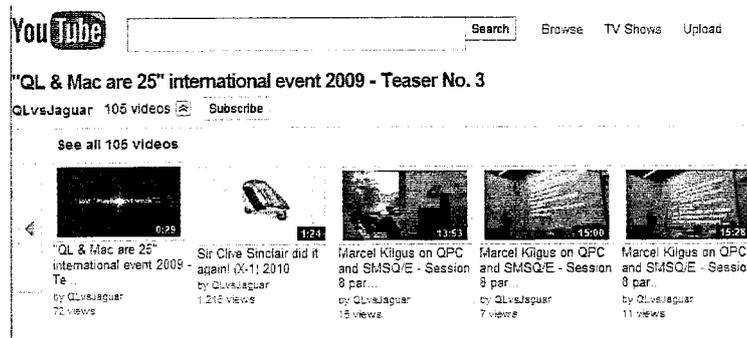
Alternatively try the link:

<http://www.youtube.com/watch?v=s58U-gdvh4>

Urs has posted several other videos:

"This October I released six more videos on YouTube; Two covering the ATARI JAGUAR and FLASHBACK (Mini 7800), four covering the Sinclair QL. One of the QL videos is Dilwyn's "History of the QL" presentation which was shown at the shows exhibition."

<http://www.youtube.com/user/QLvsJaguar#g/u>



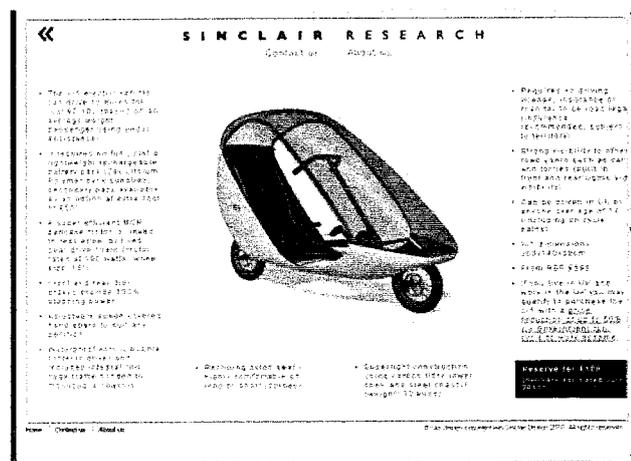
"Two famous ex QL'ers who made a difference in the IT world met at LinuxCon in Sao Paulo, Brazil recently. On a spare minute at an excursion to the Sao Paulo zoo they had a chat about their experiences with the Sinclair QL. This has been filmed by Jeremy and the video is on YouTube now. Watch it!"

<http://tinyurl.com/ql-videos>

(third video under Favorites)

Also on YouTube is a video of Sinclair Research's latest offering for anyone with £999 to spare. Details to be found at:

<http://www.sinclair-research.co.uk>

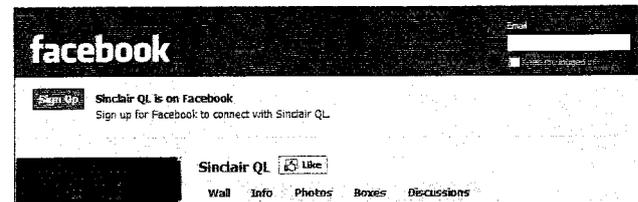


Also on Facebook

The QL also has a page on Facebook administered by Rich Mellor:

<http://www.facebook.com/pages/Sinclair-QL/32408594350>

He has uploaded photos and screenshots of hardware and software and has invited people to have a go at identifying them.



HxC Floppy Disk Drive Emulator

Rich Mellor writes:

"We have now successfully tested the HxC Floppy Disk drive emulator with the Sinclair QL home computer, meaning that you can now copy raw images of Sinclair QL disks to an SD card, and then connect this floppy disk drive emulator to your QL's disk interface and read/write to an SD card instead of floppy disks.

The latest v3.0 of Q-emuLator supports raw disk images, so when that is officially released, you should be able to use that to get software into the raw disk format

required. Then all you need to do, is to convert it using the supplied HxC software, copy it to the SD card and then hey presto, the QL sees it as a floppy disk! At the moment, I can only get Q-emuLator to read from raw disk images, although Daniele is working on raw disk image support.

You can copy any number of floppy disk images to your SD card, and then use the buttons on the front of the control panel to set up which disk image is to be seen by the Sinclair QL as FLP1 and which is to be seen as FLP2.

Currently, it only appears to support DD disks, although I dare say it will work with other disk types if I had some raw disk images of QL HD and QL ED disks to try (I cannot make them on my PC).

Ideally, we need to find some power supplies which can be used to power these units - at the moment, we are using an old external floppy disk case. Also, for some reason, both the Gold Card and the MicroP disk interface we have here needed the cable the other way around to the QL norm!

This means that the HxC Floppy Disk Drive Emulator available from SellMyRetro.com

<http://www.sellmyretro.com/search/naturalSearch?keyword=hxc>

has now been shown to successfully work with all of the following equipment as a complete replacement for the good old floppy disk drive:

- Any computer/piece of equipment that uses PC formatted floppy disks (3.5", 5.25" or even 8" drives)
- Atari ST/STF/Falcon
- Amstrad CPC6128
- Commodore Amiga (currently write only)
- Dragon 32 / 64 (VDK or JVC disk format, which should also therefore work with the Tandy CoCo)
- Emax and Emax II Sampler
- Ensoniq Mirage Sampler
- Korg DSS-1 Synthesizer
- MSX2
- Oberheim DPX1 Sampler
- Oric Computer (with MicroDisk)
- PC
- PC88
- SAM Coupe *.MGT and *.SAD formats
- Sinclair QL raw disk images
- Sinclair ZX Spectrum +3 or Sinclair ZX Spectrum with PlusD disk Interface
- Super Wildcard DS-SWC3201
- Thomson TO8D
- TI99/4A
- x68000

Boots On?

Although QL Today is posted to all non-German readers from Austria on the same day, some readers have to wait longer than others to receive the magazine. In fact it is delivered in Canada earlier than any other land than Austria itself, 3 days after posting. Next is the UK at 9 days and then next door Switzerland at 13 days. France takes 14 days and Belgium and Norway 15 days.

Dilwyn Jones was merciless:

"Well, there we have it. The ultimate cost-saver for Jochen. He would be able to walk all over Europe to drop off copies knowing that in most cases he'd get there quicker than the postal services could manage. You're WALKING - a few good pair of shoes is all you need."

Clearly Dilwyn (Dylwin? Dillwyn?) has not forgiven QL Today for spelling his name in three different ways three issues ago.

New QL Forum

Peter Scott sent a last-second news item:

www.qlforum.co.uk

It looks great!



News

25 Years - Part 5

by Tony Tebby

2005ish – The beginning of the end for two paradigms

Objects lose out to defined data structures

2005 marks the year that Microsoft's Office Open XML was proposed as a standard document format. This was published in December 2006 as standard ECMA-376. What has that to do with objects? Nothing! That is the point. A fundamental part of the object concept is that the structure of the data is hidden and cannot be accessed directly. A collection of algorithms (properties and methods) is provided to set, retrieve and manipulate data without applications needing to know about the structure of the data.

One of the long standing computer science paradigms is that objects, with their hidden data structures, would displace "raw" data structures as the data carrier for data processing, storage and transfer.

Objects embedded in documents

The major advantage of representing embedded elements of documents (charts, formulae, etc.) as objects is that the data structures for these elements do not need to be pre-defined as they are manipulated only by the methods and properties exposed by the object.

Embedding objects in documents was the primary objective of Microsoft's OLE (Object Linking and Embedding). This was a nasty little patch that, according to the blurb, enabled you to embed, for example, a pie chart in a Word document. The principle was that the embedded object could be manipulated by its associated code which could, effectively, be executed within the application container. The practice did not live up to the principle. Documents with embedded objects proved to be neither very portable nor easily maintainable.

Microsoft's Office Open XML introduced the idea of nested defined data structures. Thus a DOCX text document file (a defined data structure) can include an XLSX file (another defined data structure) with a chart eliminating the need for OLE in most cases. Unfortunately, given Microsoft's attachment to OLE, this does not seem to happen by default.

The timing of the announcements for Office Open XML appear to have been forced by an attempt by OASIS (a consortium of losers in the office workstation market) to hijack the process by getting a similar but incompatible standard (ODF) based on Sun Microsystems's proprietary office document format published by ISO/IEC just seven days before Office Open XML was approved by ECMA International.

Office Open XML, like ODF, has an archaic expression of the data (XML) and a document concept based directly on the nearly 30 year old separation of word processor, spreadsheet and slide show rather than a more easily processed binary format with separation of the content (in a common format) from the presentation (in a coherent set of formats). Office Open XML does, however, mark a major step back towards sanity: in most cases OLE can be replaced by embedded data structures.

Objects for distributing information and processing

The advantage of using objects for distributed processing is that, since the data inside the object cannot be accessed directly by an applications program, it does not make any real difference whether the data is on the same machine or on the other side of the world: the operations can all be done passing messages. There is one little difference: using an object within a program is stunningly inefficient, whereas using an object on another machine is mind-bogglingly inefficient.

Distributed Objects Everywhere (DOE) was Sun Microsystems's attempt to allow data to be processed remotely by using object methods called from one machine to process object data on another. It was based on the Common Object Requesting Broker Architecture (CORBA) which allowed clients to retrieve objects (data and the code to process it) over networks. Five years in development, it was released as NEO in 1995 and withdrawn in 1996. It is not at all clear whether CORBA is dead as well.

Sun replaced DOE by Java applets for clients and servlets and Javabeans for servers. More than 10 years later, PHP (not really object) seems to have displaced object oriented server side Java for all but a hard core of true believers. As far as transferring data to net clients, Java seems to have been reduced to an animation niche. The bulk of data distribution over the net is not in objects but in defined data structures: starting with html, through standard image formats, jpg, gif and png, to swf, pdf, avi, etc.

Objects in programming

The intention of object oriented programming was to reduce development costs and improve quality by simplifying the process of creating software, improve the maintainability and re-usability of software, going one stage beyond conventional modularity. From a naïve point of view, object oriented programming has two advantages: applications can manipulate different types of object using common methods, without knowing how the methods work on each particular object, and the internal data structures can be modified or extended without any impact at all on application code.

As a consequence, since the operation of the methods depend on both the data structures (which are not "exposed" and liable to change between versions) and classes of object (which are not necessarily known in advance) methods and their side effects can only be defined in the very vaguest of terms. Some defenders of object oriented programming claim that methods and their side effects can be well defined but, in general, this can only be true if the methods apply only to a known class and if the data structures are fixed. However, if the class is known in advance and the internal data structures are fixed, the only differences between using well-defined methods and well defined, old-fashioned procedures with well defined data structures and functions are the syntax, the efficiency and the flexibility.

Object oriented programming is extremely inefficient, reducing the performance of software while increasing its size, greatly contributing to the bloatware phenomenon. According to computer science dogma this is unimportant as users can always get a more powerful computer: the inefficiency is "justified" as the approach reduces development costs and improves reliability.

Does it? Over the past 25 years object oriented programming has taken over from other software development methods while software development costs have ballooned and quality has declined.

There is a small minority view (see Box 12) that puts the peculiar flexibility of object oriented programming as one of the causes of this decline, while the majority view seems to be that things would have been even worse if object oriented programming had not been adopted. This majority view that object oriented programming has saved the world has the advantage that, since it can neither be proven nor disproven, it can be taken on faith: anyone who believes otherwise is clearly demented.

Box 12 - The flexible animal object

Programming using defined data structures and object oriented programming provide radically different notions of flexibility. With defined data structures, an application programmer can easily extend the range of functions and procedures beyond the "standard" library functions for that data structure, but he also has the flexibility to destroy the integrity of the data. With object oriented programming, an applications programmer is strictly limited to the standard methods as only programmers with inside knowledge can extend the functionality. Object oriented programming provides, however, the flexibility to use a given method to manipulate radically different objects without needing to know how the method works. Is this a good idea?

Searching the net for any sign of a rational justification for object oriented programming I came across the "animal" object which was used in a computer science course as an example of how the flexibility of object oriented programming simplifies software development, reducing costs and improving quality.

The animal object has a method goFaster. For some animals goFaster makes them run, for others it makes them fly. There is no "fly" method because not all animals can fly, and there is no "run" method because not all animals can run. goFaster is, therefore, an abstraction for either. A programmer does not need to know how an animal responds to goFaster, it just works for all animals. Magic!

Imagine, for a moment, that the animal object is used in a simulation with birds. When a bird pecking at seed on the ground reaches a stream, we can get it across the stream with the goFaster method. This will work fine until the system is used with flightless birds, when, except in a few remarkable cases, the flightless bird will drown.

You might say that the problem was caused by the programmer who did not know that there are flightless birds. That is missing the point entirely: in object oriented programming, programmers are not supposed to know what goes on inside an object, so they can have no idea of the limitations on the applicability of any particular method.

You might say that the problem was caused by the programmer using the wrong method. That is missing the point entirely: the number of methods "exposed" by an object is limited and programmers have to use the nearest match unless write their own methods, which would negate the whole purpose of using object oriented programming as they would have to know about the object's internals, which they are not supposed to.

Why would anyone think that forcing ignorance on software developers is a good idea? Beats me.

Multiprocessing hits a brick wall

Although symmetric multiprocessing was fundamental to the 1960s systems theories that have dominated systems design from the time MULTICS was designed and although symmetric multiprocessing versions of Unix and Windows had been around for more than a decade, by 2005 symmetric multiprocessing was still a very restricted sector. In May 2005 AMD introduced its first multicore workstation processors, the Opteron and the Athlon 64 X2, and Intel introduced the Pentium D. Over the next year, experience of multiprocessing increased rapidly.

From the 1960s the IT establishment had viewed symmetric multiprocessing as the natural choice for performance. The driving force behind Windows NT was to produce a shared memory symmetric multiprocessing competitor to UNIX. In the guide to Windows 2000 (NT5), Microsoft stated its position, which corresponded roughly with the establishment view.

If you wait long enough, perhaps your performance problems will just go away with the next generation of computer chips! Another proven technique is multiprocessing, building computers with two, four, or more microprocessors, all capable of executing the same workload in parallel. Instead of waiting another 18 months for processor speed to double again, you might be able to take advantage of multiprocessing technology to double or quadruple your performance today.

The widespread adoption of multi-core processors in desktop PCs has changed the perception from simple scalability towards doubtful benefit. Very few would now be bold enough to describe shared memory symmetric multiprocessing as a "proven technique ... capable of executing the same workload in parallel ... to double or quadruple your performance (two or four microprocessors)". This is a

goal (the unachievable "Holy Grail" of computer science) that has kept tens (or is it hundreds?) of thousands of computer scientists off the streets for nearly half a century.

Multiprocessing problems

Microsoft's "Multiprocessor Considerations for Kernel-Mode Drivers"²⁰ (October 2004) states by way of introduction "future technologies mean that all new machines will eventually support more than one processor". The document concerns shared memory symmetric multiprocessing as implemented in Windows NT. There is no attempt to justify shared memory symmetric multiprocessing rather than asymmetric architectures and there is no suggestion that single processor architectures could continue to provide a cost effective solution for a large part or even the majority of systems. In the 1960s, multiprocessing was seen as a means to an end, for the last 25 years it has been an end in itself.

In shared memory multiprocessing systems, the problems of memory contention are both far more performance critical and far more delicate than for other architectures. Whereas Microsoft was bullish about their proven multiprocessing technology in the blurb for Windows NT5, 4 years later in this document they became less categorical about the reliability of their system.

However, in a few situations, you must prevent or control reordering. The volatile keyword in C and the Windows synchronization mechanisms can also enforce program order of execution in nearly all situations.

So, according to Microsoft, the memory contention mechanisms used by Windows device drivers for multiprocessor systems will work in "nearly all situations". What happens in other situations? This is hardly "proven technology" of the year 2000.

Multi core performance benchmarking

The performance benefits of multiprocessing depend very strongly on the type of applications being run. There is a class of "embarrassingly parallel" problems that comprise a very large number of similar, independent calculations (such as calculating fractals, image generation, finite element analysis, playing chess, etc) or similar, independent operations (such as indexing, data mining, spiders, etc). Although they are all easily implemented as parallel processes, they are not necessarily suitable for shared memory multiprocessing: if the algorithms are more data intensive than calculation intensive, then memory bandwidth or disk bandwidth will be the limiting factor and using computer arrays or farms will be a far better approach than using multi-core or multiprocessor systems. Furthermore, these problems are rarely true workstation applications in that they yield results over timescales from minutes to days.

In the enthusiastic rush by the press to print articles extolling the virtues of multi-core processors in personal workstations, there was little objectivity. A typical example is the comprehensive AnandTech report²¹ summarising the performance on a range of standard benchmarks (See Box 13). Not only did the results show that dual core processors nearly always had a worse price / performance ratio than similar single core processors, they also showed that, on a number of important tests, the performance of dual core processors could be improved simply by disabling one of the cores.

Three types of usage gave distinct results.

For ordinary "one thing at a time" usage, where a single operation dominated even though system maintenance tasks would be working away in the background, the second core was either useless or degraded the performance. This type of usage had two groups of applications: some potentially embarrassingly parallel applications which would become parallel by 2009, and "office" applications where a single core system is likely to be best choice for the foreseeable future as the most important performance criterion is not the "throughput" measured in these tests but the response time – the time to carry out a single action in response to a single event (keystroke, mouse click, etc.) – and the predictability.

20 http://download.microsoft.com/download/d/e/b/a/eba1050f-a31d-436b-9281-92cdfcae4b45/MP_issues.doc

21 <http://www.anandtech.com/printarticle.aspx?i=2410>

Box 13 - Multi-core processor benchmarks

A good example of independent benchmarking in 2005 of the new dual core processors was published by AnandTech. The set of benchmark results included two AMD processors with identical technology and clock speed (Athlon 64 4000+, single core and Athlon 64 x2 4800+, dual core) but twice as much high speed cache for the dual core processor. The benchmarks were carried out for three different types of workload.

1 One thing at a time workstation usage

- For 8 straight office applications (Business Winstone, PC Worldbench and WinRAR) the dual core was 0.88 to 1.00 times as fast as the single core processor, i.e. the dual core processor would have performed better if one of the cores had been disabled.
- For the single-threaded computer generated image test (with two different sets of data), the dual core was 1.01 and 1.03 times as fast as the single core processor.
- For 6 graphics intensive games, the dual core was 1.01 to 1.04 times as fast as the single core processor.

For all of these, an equivalent cost single core processor would clearly be a better choice for performance.

2 Embarrassing parallel multi-threaded applications

- For 8 straight media encoding tests, there were three groups of results. Three of the tests (image and sound processing) gave no significant difference between dual and single core despite the fact that two of the three programs were explicitly multi-threaded (more recent tests have shown some "improvement" in these applications). For three of the tests (video encoding) the dual core was 1.16 to 1.38 times as fast as the single core processor. For the other two tests (DivX and WMV9 HD video encoding) the dual core was 1.73 and 1.91 times as fast as the single core processor.
- For the single 3D image generation test (repeated five times with different data) the dual core was an average of 1.82 times faster than the single core processor.

The median speed advantage of the dual core processor was 1.24. However, the test conditions were biased against the single core processor by using the same number of threads for the single core processor as was used for the dual core processor introducing unnecessary overheads on the single processor system.

3 Multitasking user

The third group of tests simulated users carrying out background tasks while working with a principle application in the foreground. Aggregate speed is a very poor measure of perceived performance.

- For 3 office applications where the user was performing several tasks simultaneously (Sysmark 2004) the aggregate speed for the dual core was 0.95 to 1.27 times the single core processor speed.
- For 5 media creation activities with the user generating or encoding images (1) or videos (4) while carrying out other operations the aggregate speed for the dual core was 1.40 to 1.55 times the single core processor speed.

There were a number of tests showing the difference in background task speeds, but only in one case were both speeds given. This showed that the Windows XP scheduler was fairly effective in that the execution speed of the foreground task was very similar for single and dual core, but that, with a single core processor, Windows NT sacrificed the speed of the background task(s) to maintain the foreground task speed.

The results were very different for the embarrassingly parallel applications. Multi-threaded image generation and video processing should give the most favourable results for multi-core processors. But, even if as many as 5% of workstations are used for these applications an average of 10% of the time, they would currently represent less than 1% of workstation usage. Despite this they figured in 16 out of the 36 straight benchmark results, which indicates the level of bias in this and other reports. Even so, an equivalent cost single core processor with correctly configured applications would have given as good or better performance on 4 or 5 out of the 9 tests dedicated to embarrassingly parallel applications. The other tests gave genuinely better results for the dual core processor but they fell very far short of the "expected" doubling of performance. The principal reason is that the standard "desktop" workstation configuration is very ill-suited to these types of applications: using two separate computers, each with a single core processor and half the main memory could have provided a much higher performance, at little extra cost, than the dual core processor tested.

The third type of usage was the "multitasking user". These tests rarely measure anything meaningful. Where a user is working with an application while, for example, fetching e-mails in the background, the

speed of the application in the foreground is important, but the speed of the background task is not. Measuring aggregate speed (or even worse, just the speed of the background tasks as in most of these benchmarks) gives results that are very favourable to multiprocessor architectures, but rarely applicable to the real world. The only test that measured the foreground task speed showed no advantage for the dual core processor.

All these benchmarks taken together indicate that, although there are certain cases where the dual core processor performed better, the dual core processor would certainly underperform a single core processor with similar technology, similar total cache and similar cost (and, therefore, higher clock speed and more "cache per processor") under typical workstation conditions. "Power users" might find that when they are running a number of tasks simultaneously the dual core might give a higher performance, and this higher performance might offset the lower performance of the dual core processor on a more mundane workload – but it is not certain and much will depend on the effectiveness of the operating system's scheduling algorithm and the prioritisation of the foreground and background tasks.

Multicore processors 4 years on

In 2009 AnandTech published another benchmark report²² featuring multi-core processors. As the report set out to compare quad core processors, ordinary workstation use was excluded. Even so, the performance improvement in embarrassingly parallel applications was only about a factor 2 in the four years since the previous benchmark cited above, very much below the previous rate of a factor of 2 every 18 months for applications in general. Furthermore, even under the very favourable benchmark conditions, the report pointed out that the migration from dual core processors to quad core processors only increased the performance by about 30%: the largest contribution to performance improvement was the new cache architecture in both Intel and AMD processors.

Hardware for parallel processing in workstations

One of the notable features of benchmark reports from 2005 onwards was the absence of critical comments pointing out that, even for embarrassingly parallel problems where the work can easily be divided into a number of independent tasks, the new generation clearly failed to approach the "proven" 2 or 4 times increase in processor power using 2 or 4 processors.

Fundamentally, multi-core processing is a loser technology for workstations (see Box 14), as was the earlier RISC architecture. One of the common features of the applications where the benchmarks gave multi-core processors a significant advantage was that they carried out intensive processing on relatively small datasets.

Intensive processing on relatively small datasets is ideal for "computer farms" where an "intelligent" controller "farms" out the work to not very intelligent, but very fast, calculating units, ideally with "calculator" instructions (fixed point arithmetic, table interpolation, etc.) and graphical data handling (pixel masking, anti-aliasing, etc. as in GPUs). A calculating unit could be packaged as a modest quantity of fast RAM tightly coupled to a processor that occupied much less chip space than one core of an equivalent technology x86 processor while delivering several times the calculating power. Moreover, for the type of applications concerned, the processing speed would be almost proportional to the number of calculating units.

The Transputer, the first single chip computer specifically designed for farms and similar architectures, was released 25 years ago in 1984. It was an unconditional failure. Intensive calculation is the only application for which parallel processing is of clear interest, but the technological limitations of the time, coupled with the firm belief in the dogma that symmetric multiprocessing was the only true way for all computing, meant that the Transputer was not well targeted for intensive calculation and too expensive for anything else. Asymmetric (one controller for many calculators) computer farms have since become fairly commonplace for a variety of seriously intensive calculations. Why not in a workstation?

The simple answer is that intensive calculation is a tiny minority interest (I am in that tiny minority) and apparently cannot justify the development costs. But if it is only a tiny minority interest, why does this type of computing dominate current workstation benchmarks and why were multi-core processors

22 <http://www.anandtech.com/printarticle.aspx?i=3492>

Box 14 - Multi-core, a loser technology

While it should be fairly obvious that on dominantly single task workloads, multi-core processors will not provide a better performance than an equivalent cost single core processor, why can disabling cores on a multi-core processor improve the performance and why is the speed increase for ideally parallel processing very much less than the number of cores? The answer to both questions lies in the main memory bandwidth and caching.

1 Caching

At the limit, a computer's performance will be limited by the main memory access time: the speed at which the processor can move data in and out of memory. This is masked, to a certain extent, by using processor caches to hold data to improve the speed of repeated accesses to the same data items and hide the write back time from the processor.

The largest contribution to benchmark performance increases since 2005 has been improved caching with the introduction of three level caching. The 2009 AnandTech article cited gave claimed, measured and estimated access times for the three levels of caches in Intel and AMD processors.

The fast (multi-ported, prefetched) L1 caches nearest the execution unit had access times (latencies) of 3-4 cycles, the L2 caches had access times of 11-15 cycles and the large shared L3 caches had access times of 40-50 cycles. External RAM accesses took 200-250 cycles. The timings are fairly balanced with about a four times increase in access time at each level. An increase in the overall cache miss rate of 1% could increase the average data access times by about 50%.

With a multi-core processor the largest cache is usually shared between the cores. When all cores are executing, the largest cache will see continuous accesses by all the cores, with each of the tasks executing concurrently "stealing" cache continuously from the other tasks.

For a dominantly single task workload on a single core processor, this cache stealing still occurs but is limited in effect as the processor only switches tasks at well spaced intervals. With a multi-core processor, the continuous cache stealing by the background tasks can significantly increase the number of cache misses by the dominant task, reducing its performance by more than the small advantage gained by executing the background tasks on other cores. This can be seen in the 2005 benchmarks.

For an ideally parallel workload on a multi-core processor, the cache stealing can be far more serious as it not only directly increases the cache miss rate, it also increases the risk that the processor falls near or into "cache thrashing" where the miss rate increases dramatically and the performance drops. While cache thrashing can occur with just one core, the more cores that are accessing a shared cache, the more likely it becomes. Under strain, therefore, the performance of a multi core processor may degrade more quickly than an equivalent single core processor. Because this is an "occasional" phenomenon, it will not show up very clearly in benchmarks that test only average speeds, but it should rule out the use of multi-core processors in response critical systems.

2 Main memory bandwidth

25 years ago, memory bandwidth was the brick wall limiting processor performance. The introduction of caches has cushioned this performance barrier but not removed it. There will always be instructions that cannot be executed entirely from cache.

But, if there are two cores, one core can be executing from cache while the other core is waiting for the external memory. Can this provide a doubling of performance under an ideal parallel workload? The simple answer is no. Not only is it very unlikely that the cache misses (which are randomish) will interleave nicely, cache stealing will increase the cache miss rate and so the performance of each core will be severely degraded. For each additional core, there will be less memory bandwidth available for any of the cores, reducing their performance and there will be more cache stealing, further reducing their performance. With each additional core, the performance gain is less and the performance loss greater.

Even for ideal parallel workloads, unless you can guarantee that nearly all the data required for the execution of all tasks on all cores can be held in private caches or in a multi-ported cache shared between the cores, there is a limit to the number of cores that can be used before the overall performance actually drops. For ordinary workstation benchmark tests in 2005 (Box 13), this limit was one.

The dream of massively multi-core processors (64, 128 etc.) is just a nightmare.

developed for workstations when their only advantage over single core processors is for this tiny minority interest?

The emergence of multi-core processors in workstations has nothing to do with performance, it is just the pursuit of a 40 year old dogma.

More than 20 years ago, the designers of Plan 9 (and Unix) based their whole concept on the eagerly anticipated "coming wave of shared-memory multiprocessors".

More than 20 years ago, Windows NT was designed to support shared memory symmetric multiprocessing which was, at the time, a 20 year old, past-sell-by-date concept based on a 1960s misreading of the future of computer hardware in the 1970s. As a result, it was astoundingly slow, complex and over-size: the size did not matter because RAM prices were dropping and the speed did not matter because you could always use **"two or four microprocessors .. to double or quadruple your performance"**!

Software for multiprocessing

One remarkable feature of the arrival of multi-core processors for ordinary workstations that seems to have escaped comment is that existing software actually worked on these new platforms. The reason is very simple: designing all software specifically for symmetric multiprocessing has been a central computer science dogma for very long time. This could be viewed in two ways.

The conventional view is that this vindicates the 1960s dogmas of symmetry and transparency for parallel applications and the amazing foresight of the academics that developed the theories underlying these dogmas and the even more amazing foresight of the industry in developing suitable software well in advance of the arrival of mass-market, shared memory multiprocessing systems.

The minority view is that this is result of an astounding collective madness that, for 40 years, has compromised the performance and quality of software that should have been written for the single processor systems that were actually in use rather than for a hypothetical computer architecture which was to become close to a reality some time in the distant future.

The contention between processors in a symmetric multiprocessing system creates problems that need to be handled in software. The conventional methods, principally synchronisation, used to deal with the problems of shared memory symmetric multiprocessing reduce systems' performance and increase their complexity. Dealing with the increased complexity further reduces the performance while reducing the quality and increasing both the size and development costs. Is this really a sane approach for single processor or asymmetric systems where symmetric multiprocessing problems cannot occur?

On the other hand, can a system such as Domesdos or Stella, designed specifically for a single processor, work on a multi-core or multiprocessor shared memory computer? Yes it can. But can it work more efficiently than a system specifically designed for shared memory symmetric multiprocessing? Yes it can. It is a question of scalability.

An ideally scalable system would have constant overheads per processor (core), per active task and per task. Domesdos and Stella were not ideally scalable. In particular, the basic operating system overheads had a tiny scalable component and a potentially larger (N-1) component where N is the number of symmetric processors or cores sharing the same main memory. This second component is zero for a single processor and jumps as soon as there is more than one processor, leading to the accusation that these types of systems cannot be used for shared memory symmetric multiprocessors.

The reality is different. The (N-1) component for Stella is so much lower than the constant overhead of conventional shared memory symmetric multiprocessing systems (two to three orders of magnitude) that, for a modest number of processors or cores (less than 100?), Stella would maintain its advantage. Furthermore, it is finally being accepted that conventional shared memory symmetric multiprocessing systems based on synchronisation are themselves far from ideally scalable as the cost of waits and context switches that are forced by the synchronisation mechanisms increases rapidly with the number of tasks that are executing concurrently.

The question is not whether Stella would outperform BSD, Linux or NT on a symmetric 16 core processor system but whether prolonging the life of an archaic systems architecture that is totally irrelevant to current and foreseeable future requirements would be morally justifiable.

In the next issue, Tony looks at 2009 and will be "Gazing into the future"

QUANTA

Independent QL Users Group

World-wide Membership is by subscription only,

Offering the following benefits:

Bimonthly Magazine - up to 52 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 Full Membership

PayPal (see QUANTA Web Site),

Cash, Cheques and Postal Orders Accepted

Now in our Twenty Seventh Year

Further details from the Membership Secretary

**John Gilpin, 181, Urmston Lane,
Stretford, Manchester, M32 9EH (UK).**

Tel. +44 (0) 161 865 2872

Email: membership@quanta.org.uk

Visit the QUANTA Web Site

<http://www.quanta.org.uk>

Next QUANTA Sponsored Event

Annual General Meeting 2011 and Workshop

Date: Saturday/Sunday 16th/17th April 2011

Workshop from 12.00 Noon (Doors open 10 am for setting up) to 5.00 pm Saturday

And 9.00 am to 1.30 pm Sunday

Annual General Meeting 2.00 pm Prompt Sunday.

**Venue: 3rd Davyhulme Scout Headquarters "The Endeavour", Conway Road, off
Lostock Road, Davyhulme, Manchester. M41 0TF. Near M60 J9.**

Full details from
Chairman@quanta.org.uk

In Andrew Pennel's book *Assembly Programming on the QL* he dismisses the two instructions LINK and UNLK with the remark "The use of these commands is quite advanced, and will not be covered here." I took that to mean that QL programmers would not be expected to use such commands. I was surprised, therefore, to learn that C68 made extensive use of them in its compilations. Indeed I use the instructions myself in GWASS when dealing with macros.

Recently, when looking through Norman Dunbar's useful QL Wiki, I noticed that the five vectors dealing with QL queues had not been defined. I had not used these vectors myself so I decided to find out what they did and how they worked so that I could update the Wiki. In my mind these vectors are rather advanced like Pennel's LINK/UNLK. In fact the vectors seem to be used mainly, or even entirely, as part of device drivers for such things as pipes or devices using the keyboard.

Since there may be among you some who would like to delve into the murky waters of QL queues I thought that, having learnt about them, I might try and explain them here.

The QL operating system has its own special method of dealing with queues. Programmers are able to access this by means of five vectors. These vectors enable you to set up a queue and manage it.

A queue is a device which allows buffering of input or output, for example for a CON channel or a pipe. Bytes are added to the end of the queue and extracted from the front. One way of operating such a system would be to have the next byte to be extracted located at the beginning of the queue area and to have a pointer to the next available space for input. When a byte is extracted the whole queue would be moved backwards. This is a rather cumbersome method and a better one, the one actually adopted, is to have two pointers, one to the first byte to be extracted and the other to the first free position for input. These pointers would point to the front and back of the queue. This has one alarming consequence. As bytes are added and extracted from the queue, the pointers point further along the area devoted to the queue. This area is, clearly, limited. So what should one do when the input pointer comes to the end of the area? I suppose it is obvious that the pointer then points to the start of the area.

This system has one more snag to be sorted out. It is obvious that you should not be allowed to add bytes to the queue if it is full. Nor should you be able to extract bytes from an empty queue. It is therefore necessary to be able to calculate the number of bytes in the queue at any time. And this number has to be derivable from the two pointers and, possibly, the size of the queue area. When the queue starts up the two pointers are equal so the difference between them, zero, gives the current size of queue. If a byte is added to the queue, once again subtracting the output pointer from the input pointer gives the current size. If we add bytes until the queue area is filled up, the input pointer is now pointing to the start of the queue area. But so is the output pointer. This is troublesome because the pointers are indicating that the queue is both full and empty. As a final tweak to the system, then, it is decreed that the maximum number of bytes which the queue can hold at any time is equal to the size of the queue area less one. In that case equality of pointers correctly indicates an empty queue. The size of queue is found from:

`input pointer - output pointer MOD size of queue area`

Thus if the output pointer points to just one byte beyond the place to which the input pointer points, the queue is full.

Rules

The rules of operation are:

1. When a byte is put into or taken out of the queue the appropriate pointer is advanced by one. The new value of the pointer is given by:

`(Old Value + 1) MOD (Size of Queue Area)`

2. No bytes can be extracted from an empty queue. Nor can bytes be added to a full one.

3. The pointers are held in a header whose format is as follows:

Position	Item	Length	Purpose
\$00	q_eoff	Bit	EOF flag
\$00	q_nextq	Long	Not used by vectors
\$04	q_end	Long	Pointer to end of queue area
\$08	q_nextin	Long	Next location for input
\$0C	q_nxtout	Long	Next location for output
\$10	q_queue	-	Start of queue

Vectors

The five vectors operating the queue are:

Name	Purpose
IO_QSET	Sets the header given its address and the queue area's length.
IO_TEST	Tests whether the queue is empty, and if it is, whether EOF. Also gives the amount of free space in the queue.
IO_QIN	Puts a byte into a queue unless it is full or EOF.
IO_QOUT	Takes a byte out of the queue unless it is empty.
IO_QEOF	Sets the EOF flag, q_eoff.

The operation of EOF is curious. First of all only bit 31 of the first long word of the header is used by the vectors. If that bit is set it prevents any further bytes being added to the queue but allows bytes to be taken out until the queue is empty. At that stage the queue is dead. The rest of the first long word can be used as a pointer to other queues by the user.

Details of how to use these vectors can now be found on Norman's Wiki.

Goodbye Paper? - Part 2

by Geoff Wicks

At QL Today we frequently complain about the lack of feedback from readers, but the 650 words I wrote in Volume 15 Issue 1 about the difficulties of an electronic edition of the magazine produced an unexpected reaction. They sparked off a lengthy discussion on the QL Users email group, and some practical attempts to solve the problems raised.

This time I want to look at the QL use of the internet in more general terms and, unlike last time, I am writing personally and not as editor of QL Today.

Members of Quanta will remember that two years ago Duncan Neithercut made some interesting proposals for Quanta that I reported in QL Today (V13 I3 P6). He had some criticisms to make of the electronic Quanta Magazine:

"The email distributed .pdf magazine is a dopelganger for the paper magazine! None of the opportunities of electronic publishing have been used, not even colour! No doubt there seem to be good reasons for this, such as it makes it easier to publish the paper copy, and there is an upper limit to the acceptable size of email attachments."

Quanta has interpreted this as a plea for a "posher" electronic magazine than the paper one, and missed some more radical proposals:

"Abolish the magazine in its present format and move to an internet based, simpler and cheaper way of keeping members informed."

Duncan's suggested format was a blog, which if Google adverts were permitted would raise some funds for Quanta. Individuals could be authorised via passwords to create new content or upload content from other members. The content could be submitted listings, links to the library contents and other sites and general comment. He also suggested some form of paper printout for those members not on the internet.

As it happens I do not think Duncan's proposal is suitable for Quanta, and I wrote something to that effect in the Quanta Magazine, but it is a vision that started me thinking. It may not be suitable for Quanta, but it could be highly appropriate elsewhere within the QL community. Perhaps the time has come to stop thinking in terms of paper publications.

In fact this may be forced upon us. I fully expect Quanta to be wound up within the next two years and, realistically, we have to think how long a future QL Today can have.

Now consider the QL community. It is an international community and if Quanta closes there will no longer be any national QL interest groups. We shall become much more a group of individuals scattered throughout the world. The internet is an ideal way for us to communicate with one another quickly and efficiently. We also know from the QL user email discussion earlier this year that the vast majority of UK QL-ers are not members of Quanta, nor readers of QL Today, nor subscribers to the QL user group. We need to reach out to this group.

There is ample evidence that the way to do this is via the internet. As a trader Rich Mellor has shown that trading on the internet has brought him contacts and customers that other traders had missed. During the QL's quarter centenary celebration Urs König's celebratory website page had 6,341 unique hits with 1,303 downloads of the PowerPoint presentation and 464 downloads of the QPC demo version. About 2,000 visitors had come to the site via Linus Torvald's blog. Earlier this year Dilwyn Jones reported that his website attracts interest from "retro" users and that his advice pages get a respectable number of hits. Although much less visited than Dilwyn's site, the help and advice page on my own site is the most visited section of the site.

In recent years, in an attempt to broaden knowledge of the QL, Rich Mellor and Norman Dunbar have set up QL wikis. Unfortunately they do not always come high up on search engine results. Type "QL Wiki" into Google and items 5 and 6 are Wikipedia and Norman's wiki respectively, but there is no immediate reference to Rich's wiki. Type in "Sinclair QL Wiki" and items 1 and 2 are Wikipedia and Rich's wiki respectively, but there is no mention of Norman's wiki. However take a recent initiative and type one of the two terms into the dedicated search engine on Dilwyn's site and all references are to the QL. It is a good reason for paying regular visits to Dilwyn's site.

What we can learn from these experiences is that QL sites are visited and used by a wide range of QL-users and not just Quanta members, QL Today readers and QL-user group subscribers. However we need the publicity that Urs managed to create and tools such as Dilwyn has devised to help users find relevant sites.

Would the QL benefit from a public magazine style site? Instead of thinking in terms of what

Duncan Neithercut describes as a "doppelganger" for the Quanta Magazine or for QL Today, should we instead aim for a site much like the websites of national newspapers?

A website gives many advantages over a printed magazine, although there are also serious disadvantages that I shall come to later.

A major part of such a site would be an interactive rolling news page. It would have the advantage over paper of having immediacy, colour, live links and even film clips. It would, however, need dedicated news editors. Both the Quanta Magazine and QL Today are facing a problem of a decline in QL news. There are still news stories around but they are rarely there for the plucking and both Dilwyn and I have to go actively in search of them. Quanta has made some steps towards an interactive news page, but it is buried deep in their website where few non members would look and, as I write this, the news was over 3 months old.

The internet would have similar advantages over paper as far as articles are concerned. Not only would colour, live links and film clips be possible, but also, as the recent short lived editor of the Quanta Magazine, suggested circuit boards could be shown in high resolution images. There would be fewer problems with illustrations and long listings and for non-English speaking writers the possibility of reproducing their contribution in both their native language and English translation. But like the present paper magazines there would be a need for a regular team of writers as well as occasional contributions from others.

The site could also be a repository for help and advice and data documentation. This could either be on the site itself or in the form of central co-ordination providing links to help and advice on other sites. Once again this would be a team job.

There are disadvantages to web publication, particularly if the intention was for it to be a public and not just a subscribers' site. Compared with a printed magazine the very public nature of the internet would give some additional legal complications over such matters as privacy, copyright and freedom of expression. There is also the powerful argument that people value things more when they have paid for them, and that, because of this, a free site could not be guaranteed a long term future. A further complication with a rolling as distinct from a fixed publication site is how to keep people returning to the site.

All this suggests that setting up and maintaining a magazine type website would not be an

easy task and would certainly be too much for a single individual. A dedicated and enthusiastic team of workers would be necessary. It is doubtful whether the QL now has sufficient resources to run both paper and web publications, but I am

not a pessimist who believes the end of paper publications will herald the end of the QL. We should prepare ourselves for a situation where paper publications will no longer be viable.

XCHANGE

by David Denham

Plenty of QL users are still using the original Psion programs supplied with the QL back in the 1980s. There's nothing wrong with using Quill to knock up a quick document, or creating a small spreadsheet in Abacus, for example.

But there does exist in the form of Xchange an improved version of these programs which are more stable, feature additional facilities and are better integrated.

Xchange was originally produced for the CST Thor computer in the 1980s. The original Thor was based on a QL circuit board, so was just like a QL with expanded facilities such as floppy drives, more memory, better case and keyboard and so on. Xchange was then unofficially adapted for a standard QL with disc drives and expanded memory and several versions followed until the software settled at version 3.90L. Most of the work on these early QL versions was done by people like Erling Jacobsen and Gunther Strube in Denmark. Some minor revisions and patches have occurred since, mainly to improve operation on a modern system with the colour drivers. These more recent updates were written (or more correctly, adapted) by Marcel Kilgus.

Erling Jacobsen in Denmark was a driving force behind Xchange in those days (his website still exists at <http://linuxcub.adsl.dk/> and you can download Xchange itself and some documents from here).

We can get the current version and the update patches from places such as Dilwyn Jones's website and presumably other sources of public domain and freeware software. I would point out that technically Xchange is still subject to its original copyright, although we are still free to copy it for QL users.

Download Xchange 3.90L from:

<http://www.dilwyn.me.uk/psions/index.html>

Scroll down to the section entitled "Xchange For The Sinclair QL" and click on the link which says "Download Xchange V3.90L" and also the link under it which says "Download Xchange Documentation".

Having downloaded the two zip files - it downloads as files called xchange.zip (the program itself) and xchdoc.zip (which contains the documentation files for Xchange) - we can then unzip these files to where we want them on the QL.

Installing

Once downloaded, you need to decide where to unzip these files to. If you have a hard disc, you may wish to use a directory name like win1_xchange_. Create the directory there with a command like MAKE_DIR WIN1_XCHANGE_ and unzip the file there:

```
EX dddn_UNZIP; "-dWIN1_XCHANGE_
  dddn_xchange_zip"
```

Replace dddn_ with the name of the drive/directory where you stored the xchange_zip and the Unzip program.

Next, do the same for the documentation files - you don't necessarily have to put them in the same directory.

If you are using a system with floppy disc drives and no hard drive, you will need to put both zip files on the same disc - they are both around 300 kilobytes in size, so both can fit on a single DD disc for unzipping purposes. Make sure there's a copy of unzip available - there should be enough room to add a copy of the unzip program to the same disc as the one containing the zip files.

Put the disc containing the zip files and unzip into FLP2_ and a blank, formatted disc in FLP1_. The package is unzipped with these commands:

```
EX FLP2_UNZIP; '-df1p1_ f1p2_xchange_zip'
```

You should then have a list of files like this on the disk. Alongside the list I've printed a brief description of what each file is:

abba_hob	Abacus help file.
archex_asm	Archive extensions assembler source.
ARCHEX_pmc	Archive extension code.
archv_hob	Archive help file.
BOOT	A boot program to start Xchange.
CHOOSEPRINTER_BAS	Printer driver selection program.
config	QJump level 1 configuration program.
config_bas	System Configuration program for Xchange.
config_obj	Compiled version.
DEMO_zip	Some zipped up demonstration files.
DRV_zip	Some zipped up printer driver files.
DUMP_zip	Sample Easel and graphics dumps.
epsonibm_asm	Easel gprint assembler code.
gprint_prt	Easel printing code.
graf_hob	Easel help file.
hob1_bas	See hobutils.txt.
hob2_bas	See hobutils.txt.
hob2asm_bas	See hobutils.txt.
hobutils.txt	Instructions for help file utilities.
INTRO_doc	Introductory document.
pedit	Compiled printer editor.
pedit_bas	Basic printer editor.
pedit_dat	Printer driver data.
printerset_dat	Printer driver data.
QDUMP	Screen dump utility.
qdump_install_bas	Installation program for Qdump.
quil_hob	Quill help file.
quill_gls	A sample Quill glossary file.
ramdisc_cde	Machine code which installs a ramdisc.
readme_doc	Introductory document, with updates list.
README1_DOC	Document about editing and choosing printer drivers and qdump screen dumps.
TSL_doc	Document about Task Sequencing Language.
TSL_zip	Sample Task Sequencing Language files.
unzip_bas	A basic program to help you unzip files.
viewdat	Printer_dat decoder program (compiled).
viewdat_bas	Printer_dat decoder program (basic)
xchange	The Xchange program itself
xchange_dat	Xchange printer driver file.
xchg_hob	Xchange help files.

Xchange needs a ramdisc to work. The boot program loads a short piece of code which installs a ramdisc if you haven't got one installed on your system. Most modern QL systems and emulators provide one built in, in which case you can change lines 160 to 210 of the boot program to prevent it loading and formatting the ramdisc (just place a REM statement after the line numbers in those lines in the boot program).

It is worth reading through the boot program - it does provide some handy snippets of information, such as how Xchange determines the amount of working memory, explained in REMark statements in lines 280 to 310.

The first document you should read is INTRO_DOC. This provides a brief history of the package and explains the startup process. Reading the other DOC files before you start using the program will also be beneficial - README_DOC, README1_DOC and TSL_DOC.

Basic Principles

Xchange is a single program which includes Abacus, Archive, Easel and Quill in a single program. Xchange has a "front-end" screen which lets you control the other programs, start up one or more than one Quill or whichever program you need, transfer data between the programs (hence the name Xchange), print files in the background,

list files, set default drives and run something called Task Sequencing Files (which I won't go into here, but they allow a degree of automation of the program).

When you run the boot program, you get the Xchange front end screen. This is shown in figure 1.

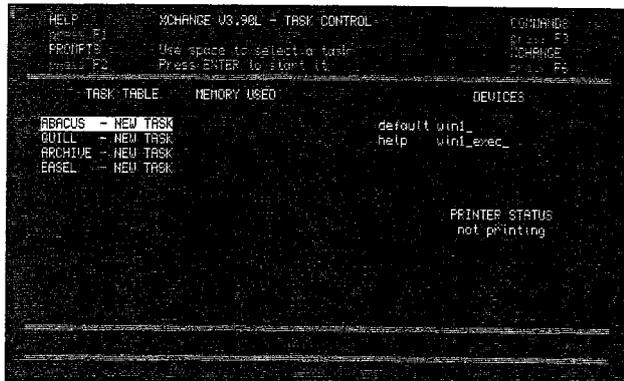


Figure 1 - Xchange front end screen

This looks fairly like the displays used in the original programs and some of the layout and facilities are similar. For example, you can get help by pressing F1. You can press F2 to toggle the prompts area at the top off and on. Pressing F3 shows the list of available commands in the usual place in the top centre of the display. Whether you are in Quill, Abacus, Easel or Archive, you can usually press F6 to return to this "control centre" from Quill, Archive, Easel and Abacus to start new tasks and so on - if you press F6 in one of the programs, it doesn't stop that program, just jumps back to the "control centre" so that you can do something else such as start a copy of another program without having to first close down the first program.

When you first start Xchange you get a list of the available four programs.

```

ABACUS - NEW TASK
QUILL - NEW TASK
ARCHIVE - NEW TASK
EASEL - NEW TASK
  
```

Selecting one of these (press a key to move the highlight down the list, it will wrap down past the end back up to the top) will allow you to start a new task for one of these programs. For example, press the cursor down key to move the highlight down to QUILL - NEW TASK then press ENTER, type in a short name such as "quill1" (without the quotes) and the familiar Quill screen

appears, then you can start typing a document. The names you give these "programs" are mainly used to identify which is which, for example, if you have more than one copy of Quill running you can call them quill1, quill2, and so on, or you can give them more meaningful names like club, work, home etc.

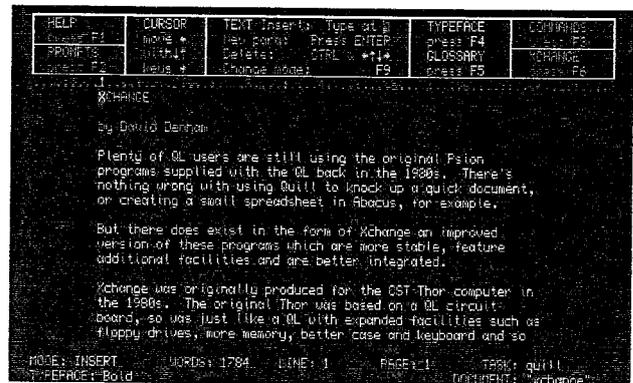


Figure 2 - Xchange Quill

Once you have finished the document and finished using Quill, just quit in the usual way (press F3, Q for Quit) and you will be returned to the Xchange front end.

The same applies to the other three programs, of course.

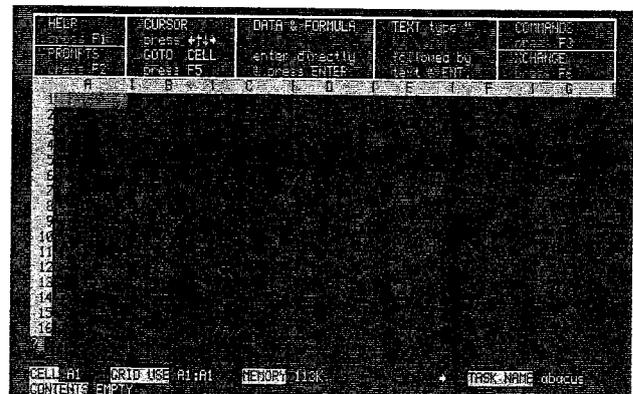


Figure 3 - Xchange Abacus

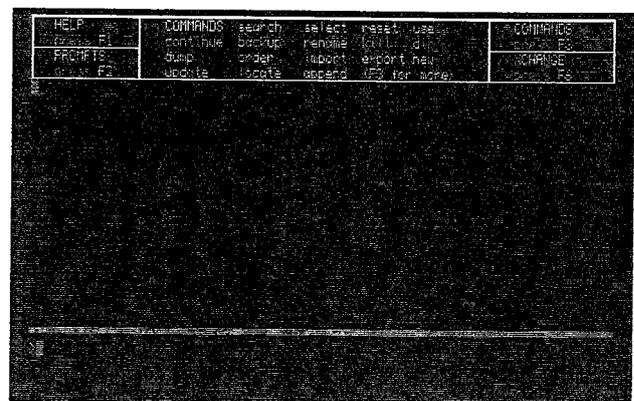


Figure 4 - Xchange Archive

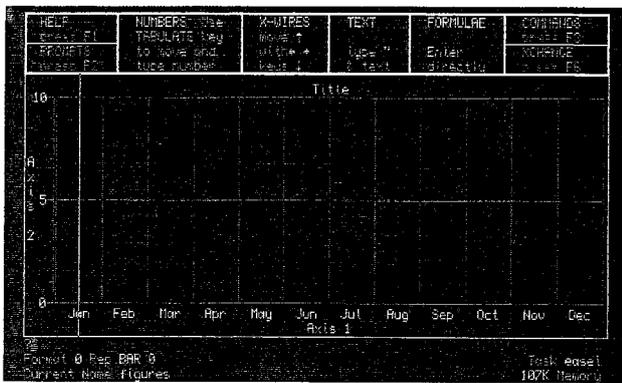


Figure 5 - Xchange Easel

When you press F6 from one of the four programs to get back to the Xchange control centre, you get a screen like the one shown in figure 6, which not only lists the options to start new copies of Quill, Abacus, Archive and Easel, it also shows which ones are running at the moment. This makes it easy to get back to the right one if you left it in the middle of doing something else. You'll see in figure 6 that I had one copy of each of the four programs running. Suppose I had been using Quill and pressed F6 to get back to the control centre and wanted to go Easel next. So all I had to do was move the highlight down to "Easel - easel" and press ENTER. Xchange now takes me back to where I was in Easel, and I can do the same thing later to get back to my running copy of Quill (or one of the others if I wished of course).

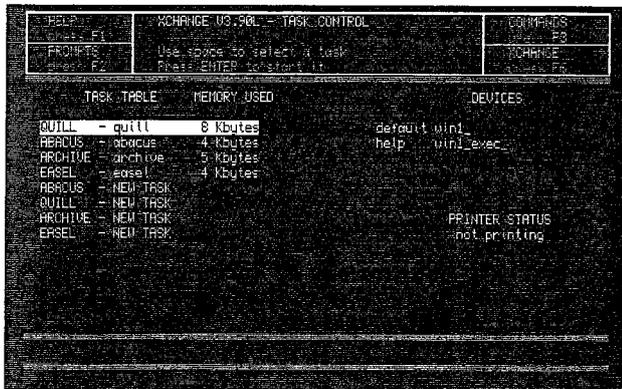


Figure 6 - Open programs shown in Xchange list

Note - I'm not limited to one copy of any of the programs. I can have two copies of Quill open if I wanted to. Also, note the names in the second column, after the hyphen. When I started each of the four programs, I could give them a name. See how the names can identify the programs in this list. If I was using two copies of Quill, I could call them Quill1 and Quill2 for example. It also lets us tell the difference between an already running copy and the options to start a NEW TASK for example.

Once you have finished using Xchange, make sure you are at the front end screen, then press F3, then press Q for Quit.

Unlike the old Quill, for example, Xchange does not take over all available free memory and you can safely CTRL-C in and out of the program. By default, Xchange grabs 310 kilobytes of space, but you can control this by specifying the number of kilobytes as a parameter in an EX command used to start Xchange:

```
EX FLP1_XCHANGE;"100"
```

This will allow Xchange to have 100 kilobytes for data. The minimum data area you can assign is 64K.

Xchange will need to set up temporary files in RAM1_, so you should ensure that a ramdisc is available, whether you use the one supplied or the one built into your QL system.

Important Note: Xchange has a config block, which you can configure with the same config program used to configure pointer driven programs. Much of what's in the config block won't make much sense at this stage, but the first item might well be important to your setup. Xchange 3.90L should run OK on a 512x256 QL display normally, but if you are getting errors which imply that Xchange has too big a display to run on a standard QL screen, the reason might be that someone has configured a border for the Xchange display. According to Erling Jacobsen's notes, adding a border is done outside the Xchange display, so it might grow too big for the display (it makes the Xchange screen bigger than 512x256). If you get this problem, configure the border to OFF. The border is a bit pointless on a standard QL, but if you are using an emulator or Aurora or Q40 with a bigger screen display, the border can be useful to show the limits of Xchange on a high resolution screen. It is intended to makes the divide between overlapping programs on the screen much clearer).

The Improvements

Once you have unzipped the documentation files disc, I suggest you first look at the README_DOC on that disk. This contains some notes from Erling Jacobsen and Gunther Strube about the improvements made to Xchange. It also lists what's what amid the collection of files on the documentation disc. Then I suggest you read the xhistory_doc. This is mostly historical information

about Erling Jacobsen's struggles with the first versions of QL Xchange, but there's some useful reading in there which gives you an idea of some of the things the new version can achieve.

Loading these files into Xchange is pretty much the same as with the original Quill. Start a copy of Quill as described above, then press F3 in Xchange Quill and type in the filename. Note that some of these files won't load into the old Quill - some of the filenames are longer than the old Quill can handle. Xchange is not limited to 8_3 style filenames like abcdefgh_doc - it can handle longer filenames than the original Psion programs can, and it can also handle directory names if you are using a system with level 2 directories. So, if you'd like to store your documents in a directory called WIN1_docs_ you can save files from Xchange Quill using a name like win1_docs_longfilename_doc.

Conclusion

That's enough to take in for now! In a forthcoming issue, I'll start to look at the new facilities. For now, try to read as many of the doc files as you can and have a little play with the program.

Be aware that some commands have changed very slightly because of the new facilities added. For example, in Quill, the Search command now asks if you'd like to start searching from the cursor point in the text, or from the top of the text, instead of always just searching from the top in the old versions of Quill.

The Files menu (F3, O for Other, F for Files) in Quill has a few new facilities as well, such as Export, Transfer and Mail. The Help function (which still works by pressing F1) contains some information on how to use these new facilities.

Have fun trying to find out what these new facilities do!

Optical Illusions

by Stephen Poole

In my 'THINK_bas' article I discussed animal intelligence and perception in some detail, and mentioned how animal brains construct images of what they sense using their normal points of reference. If those references change, their brains cannot react correctly, such as when a fish gets surrounded in a net, to which it has no evolutionarily program, and so from which it has no reflex to escape, as it assumes there is no danger. In the same way, the human brain is programmed to react to set references, outside of which illusions occur, such as those which trained magicians trap us into thinking we see. Psychologists and artists have experimented for years to see how the brain deceives the eye when confronted by certain exotic images.

I am going to show you just one illusion, where the brain interprets straight lines as curves,

(discovered by Akiyoshi Kitaoka). By fitting a disk of smaller squares inside larger ones, the brain assumes the inner circle is a dome, and therefore 'curves' the overlying lines as if they were on a protruberance, even though a ruler proves that they are dead straight. (Don't sit too close to the screen). This reveals categorically that we do not see with our eyes, but reconstruct an image with our brains, which interpret incoming signals according to the way it is evolutionarily programmed via its assumed reference points. Our brains build images using thousands of millions of 'pixels', seeing in both distance and stereo!

It will take a very great deal of research to even begin to get this sort of performance from computers. Never assume that you cannot be fooled....

```
100 ::
110 REMark illusion_bas, by S.Poole. v22nov08
120 REMark for QL Today.Beta-test by B.Coativy
130 :
140 CLEAR: RESTORE : OPEN#1,con_16
145 WINDOW 512,256,0,0: BORDER: CLS
150 WINDOW 256,206,256,0: PAPER 2
160 m1=1.85: m2=m1*2: m3=m2/8: m11=m1-.5
170 SCALE 80,-8,-16: st=8: ss=2+7*st
180 s=ss-2: r=ss+2: t=90
190 :
200 DATA 0,0,0,0,0,0,0
210 DATA 0,0,3,5,1,0,0
220 DATA 0,3,3,5,1,1,0
230 DATA 3,3,3,5,1,1,1
240 DATA 1,1,1,2,3,3,3
250 DATA 0,1,1,2,3,3,0
260 DATA 0,0,1,2,3,0,0
270 DATA 0,0,0,0,0,0,0
280 :
290 DATA 0,0,0,3,1,0,0,0
300 DATA 0,0,3,3,1,1,0,0
310 DATA 0,3,3,3,1,1,1,0
320 DATA 0,6,6,6,4,4,4,0
330 DATA 0,1,1,1,3,3,3,0
340 DATA 0,0,1,1,3,3,0,0
350 DATA 0,0,0,1,3,0,0,0
360 :
370 DATA 0,0,0,0,0,0,0,0
380 DATA 0,0,3,3,1,1,0,0
```

```

390 DATA 0,3,3,3,1,1,1,0
400 DATA 0,3,3,3,1,1,1,0
410 DATA 0,1,1,1,3,3,3,0
420 DATA 0,1,1,1,3,3,3,0
430 DATA 0,0,1,1,3,3,0,0
440 DATA 0,0,0,0,0,0,0,0
450 :
460 DATA 0,0,0,5,0,0,0,0
470 DATA 0,3,3,5,1,1,0,0
480 DATA 0,3,3,5,1,1,0,0
490 DATA 6,6,6,0,4,4,4,4
500 DATA 0,1,1,2,3,3,0,0
510 DATA 0,1,1,2,3,3,0,0
520 DATA 0,0,1,2,3,0,0,0
530 DATA 0,0,0,0,0,0,0,0
540 :
550 FOR up=2 TO ss STEP st
560   FOR ac=6 TO ss STEP st
570     READ sk: square ac,up,7
580 END FOR ac: END FOR up
590 :
600 FOR up=6 TO ss STEP st
610   FOR ac=2 TO ss STEP st
620     READ sk: square ac,up,7
630 END FOR ac: END FOR up
640 :
650 FOR up=2 TO ss STEP st
660   FOR ac=2 TO ss STEP st
670     READ sk: square ac,up,0
680 END FOR ac: END FOR up
690 :
700 FOR up=6 TO ss STEP st
710   FOR ac=6 TO ss STEP st
720     READ sk: square ac,up,0

```

```

730 END FOR ac: END FOR up: i$=INKEY$(#1,-1)
740 ::
750 DEFine PROCEDURE square(x,y,i)
760 LINE x,y: TURNT0 0: MOVE m1: INK i
770 PENDOWN: FILL 1: TURN t: MOVE m1
780 FOR j=1 TO 3: TURN t: MOVE m2
790 TURN t: MOVE m1: FILL 0: PENUP: sq sk
800 END DEFine
810 :
820 DEFine PROCEDURE sq(qr)
830 IF i=7: INK 0: ELSE INK 7
840 SElect ON qr
850   =1: dot 0 : dot 180
860   =2: dot 180: dot 270
870   =3: dot 90 : dot 270
880   =4: dot 90 : dot 180
890   =5: dot 90 : dot 0
900   =6: dot 0 : dot 270
910   =REMAINDER
920 END SElect
930 END DEFine
940 :
950 DEFine PROCEDURE dot(dt)
960 LINE x,y: TURNT0 dt: MOVE m11: TURN t:
    MOVE m11: squ
970 END DEFine
980 :
990 DEFine PROCEDURE squ
1000 PENDOWN: FILL 1: TURN t: MOVE m3
1010 FOR j=1 TO 3: TURN t: MOVE m3
1020 TURN t: MOVE m3: FILL 0: PENUP
1030 END DEFine
1040 ::

```

QL Today, The PDF Version

by Norman Dunbar

After Geoff's comments in the last issue regarding the difficulties involved in creating a PDF version of QL Today, the news group "took off" with all sorts of comments and requests for, would you believe it, a PDF version of QL Today!

Geoff had pointed out the problems of:

- File size
- Quality of print
- Time constraints

And these are the bugbears in creating a usable PDF file. In addition, Jochen added that whatever was necessary to create a decent PDF file should not involve any further work on his part, additional software to be learned and so on. Jochen does enough hard work producing the magazine and has little time for any more work on it.

Looking into the situation and based on comments on the list, I found a number of problem areas:

- The DTP system used is Calamus on Windows.
- It cannot produce a PDF file directly.
- Using a Windows "printer driver" to create a PDF file from a printed document works, but, takes many hours of "printing" and eventually fills the hard drive with a massive data file for a 50 page magazine.
- Using a Calamus module to create a PDF works much much faster, but still created a bitmap PDF file where the text cannot be selected and copied out (unless you run Okular on Linux - but that's another story!) - however, the minimum resolution required to get a decent quality is 300 DPI (Dots Per Inch) and printing at this resolution creates the problem of a huge file again - 6 Mb for a sample of 10 pages.

So, it looks remarkably like Geoff & Jochen's comments that it is far too difficult to do stand.

I produce large numbers of documents in PDF format using Docbook which is an XML system, in fact, all my articles in the Assembler Series are produced in this way, but converting all submitted QL Today articles to XML is a non-starter. The good news is that all the PDF files produced are text based and not bitmapped.

One of my documents is currently 228 pages long, with screen dumps etc, and the file size is approximately 800 KB which is a massive difference from the 6 MB file with 10 pages I mentioned above. What to do?

Obviously we don't want Jochen & Geoff to have to purchase software and learn new skills and Jochen has already intimated that he doesn't have the time to learn a new system when the one he uses is perfectly good for his needs producing a paper magazine. However, I'm wondering if there is an easy way. OpenOffice.org.

OpenOffice.org is a freely available and excellent Office suite that is compatible with Microsoft Office however, it can produce PDF files instantly with the simply click of a button. PDF creation is built into the system and easy to use. So, how hard can it be to create a PDF copy of the magazine using OpenOffice.org?

While I've used OOo (as it shall be known from now on!) for many years and run my business using it, I've never really progressed beyond the standard document layouts. So I decided to investigate setting up a document that allows single columns, tables of contents, two column layouts and so on. How hard can it be?

Well, eventually, I produced a "magazine" with about three articles in it, in PDF format with a mix of single and twin column layouts. It's was a small file to boot and looked right up our street.

Jochen doesn't really get on with OOo. He has had difficulties in the past with it - admittedly importing a Microsoft Office document for a customer, not creating a magazine from scratch - so the idea fell by the wayside straight away. To be honest, I had a few problems getting things working correctly myself and I wouldn't really like to use OOo to create a magazine - with or without a PDF version. It's simply just not suitable. It's a word processor (well, the part I used is!) and not a DTP system. What next?

I have, for some time, been quite interested in a program named Scribus which has been receiving lots of good reviews in the Computer Press for it's quality and ease of use. Scribus is a DTP

system similar to Calamus which Jochen uses. Could be worth investigating I thought.

As it turned out, I already had it installed on my laptop (under Linux) and fired it up. I have no experience of DTP layouts and I had never used Scribus at that point either, so I was a total newbie to the processes.

No sooner had I started looking into the program than Urs König announced on the news group that he had produced a QL Today magazine (Issue 1, Volume 1) in PDF format and had made it available for download (with Jochen's permission) at

http://www.cowo.ch/downloads/1996-05_QLToday_V0111.pdf

The story he related was about scanning each page on a high quality scanner and creating a 50.2 Mb download (so beware if you are on a slow link!) for 56 pages.

I downloaded the issue from the above and the quality is indeed excellent, and you can search for text and copy and paste it elsewhere, but, once again, we have an extremely large size.

Given my experiences with text based files, my 228 page PDF should be around 204.3 Mb in size - quite a difference from the actual file at 800 Kb. Something has to be done!

Using Linux, my PDF reader of choice is a utility named Okular. With this I can draw a rectangle around some text - even in a bitmapped page - and then get the option to copy to the clipboard as text (doing OCR on the fly!) or as an image. This is useful and with a good quality PDF like the one from Urs, I was able to extract the first 9 pages - including the cover as a bitmap - and create a text based PDF using the extracted text. The result is to be found at

<http://qdosmsq.dunbar-it.co.uk/downloads/QLToday.pdf>

and is only 1.1 Mb in size which includes the cover as an image plus the fonts required to display the file exactly as I created it.

There is a problem in that the cover page has a page number - but I can easily get rid of that as I have now discovered something called Master Pages - and these are something that Jochen uses regularly in QL Today - which allow me to define standard boilerplate text to appear on each page of a given type - cover, inner cover, two column, single column etc - and by using these master pages, I can apply a cover page that has no footer and page number.

However, as I said, I had never used a DTP system before and so, I was unaware of these helpful features. I'm a bit better prepared now. I didn't have too much time to spare to carry out the above exercise, however, I have to say that it took me about one hour to create the 9 page test document and, I see from my notes, I took longer to choose the correct fonts than I did creating the layout and the content. My current problems with Scribus are few - I've even used it to create the "Dunbar Family Christmas Newsletter" that my wife likes to send out at Christmas time - but creating table's of contents seems to be a problem. A text frame can only have one TOC entry, even if there are multiple "subject" within that frame that you want in the TOC. You simply have to use more frames to resolve the issue.

Now, as I mentioned, I had to OCR the text from the various places to get content for my 9 pages. This did require me to make some corrections here and there as OCR is never going to be perfect, even with a starting PDF as good as the one from Urs. Jochen and Geoff do things slightly differently. They start with a text file containing the author's article, possibly a few screen shots, and then work out a layout to suit the content (or force the content into the desired layout!) - so the various spelling mistakes and punctuation problems etc. that appear in my document will not appear in theirs.

Extrapolating the 9 pages into a full 56 pages like the original should work out around 6.8 Mb which is a good size for a magazine and works out at around 7 and a bit of my magazines per one copy of Urs' version. No disrespect to Urs, as I've mentioned he has done an excellent job -

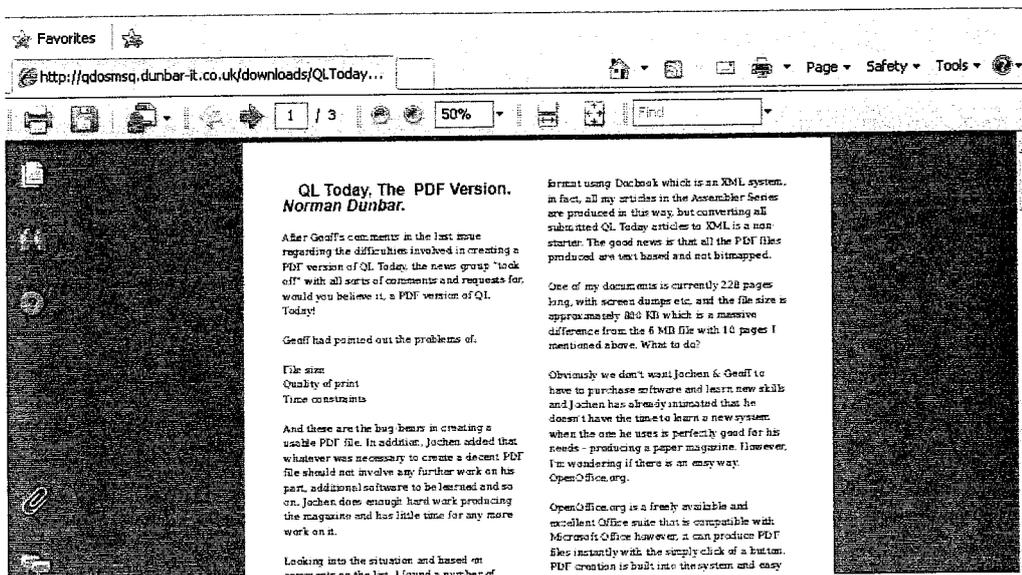
it's just the bitmapped pages that are to blame.

Now, again, I'm probably way off base here because Jochen did mention that he was not willing to spend time learning a new system as he has little time to devote to doing so. I'm taking a risk in saying that I suspect that if he was to use Scribus to create QL Today, the skills he has learned over the years with Calamus should stand him in good stead. However, that is a decision that Jochen needs to make. (Scribus is available for use on Windows too by the way)

Moves may be afoot on the Calamus front as well however. A module named "The Bridge" (which sounds more like an Iain M Banks novel to me) has been touted as a possible help in this matter. I did take a look at this product while I was playing with a trial edition of Calamus on a Windows XP Virtual machine on my laptop (if Windows decides to mess up, it only messes up the VM and not my system - been there, got stung by Windows before, plus, I have no intentions of leaving it around - I don't use Windows often enough to justify its continued existence.)

What I found was the fact that the description and instructions are written in German - which I'm rather ashamed to admit, I don't read or understand - and that no trial version is available. So, I'm unfortunately unable to say whether or not "The Bridge" is the solution to our needs (or wants!).

So there you have it. I hope the above has proved useful and that you understand now why it is quite difficult to produce a QL Today in PDF format without investing more time and/or money in getting it done.



This article is available in PDF format in three forms. One is a PDF created in single column mode by OOo, a second OOo document in two column format and thirdly, a Scribus PDF in the magazines normal two column layout. Nothing fancy, just plain text.

By the way, I liked Scribus so much, I bought the book - Scribus, The Manual - from Amazon. Not surprisingly, that book itself was created using Scribus. Talk about recursion. :-)

Links Urs Koenig's QL Today:
http://www.cowo.ch/downloads/1996-05_QLToday_V0111.pdf

Norman's 9 page "sampler":
<http://qdosmsq.dunbar-it.co.uk/downloads/QLToday.pdf>

This document OOo One Column:
http://qdosmsq.dunbar-it.co.uk/downloads/QLTodayOnPDF_oneColumn.pdf

This document OOo Two Columns:
http://qdosmsq.dunbar-it.co.uk/downloads/QLTodayOnPDF_twoColumn.pdf

This document, Scribus:
<http://qdosmsq.dunbar-it.co.uk/downloads/QLTodayOnPDF.pdf>

The quest for another lost treasure

by Urs König

Before I start

This is a follow up to my article "The lost treasures in-depth" which was published in QL Today V14I3. In this article I use the expression "dream machine" for a computer which was advertised in 1986 but never made it to market. Some said it never existed and was pure vaporware. Others said it was (almost) ready for production but due to legal and financial problems it was killed before birth. I'll tell you my story on the quest for another lost treasure.

Things went a different way

Christmas 1987 was the time when it became clear that I could never buy the dream machine of that time. Over the years other systems came, stayed a while and left my desk. As both the Macintosh and NeXT stories did not develop in the way I was hoping for, I went the cheap and easy way. So it came that in 1992 Windows took over my daily computing tasks at work and in private life. While the development tools and applications were far superior to anything I knew from my QL years, I had to wait until 1995 until Windows NT4 and the Socket 7 Pentium MMX (P55C,) made me relatively happy on the OS and CPU side. 11 years after the first QLs shipped I had a similar multitasking experience to what I was used to in my QL days. Well, the PC was clocked 16 times faster and had 128x more memory than the good old QL.

Years later

A few days after Christmas 2001 a message mentioning the dream machine was posted in the internet by someone living in the US. I tried to learn more but got no replies to my email query. Another 7 years passed until early 2009. While doing some research on the internet preparing my "QLs launch 25th anniversary" message I

came across an announcement that the prototype of the dream machine was on display at a retro computing exhibition on Nov 30th 2008. I tried to contact the guy who ran the event but alas got no response.

Remarkable coincidence

In September 2010 we were looking for a nice place to spend some days on autumn holiday. My wife Sandra finally found a nice hotel in northern Italy at the Lake Lugano. I have to say the Mediterranean climate and life style is just a two hours drive from our home. Some family friends planned their autumn holidays nearby, so we arranged a dinner at a lakeside Pizzeria for Wednesday Oct 6th. Everything sounded perfect! While investigating the area in Google Maps for must see places and hot spots, I realised that the above mentioned 2008 retro computing exhibition was in the same province and just some 18 kilometres from where we intended to stay. The day before our trip I posted "Anybody near..." to the Italian newsgroups where I came across the show announcement. In addition I sent another personal email. I had no great hope for a reply, but while driving through the Gotthard road tunnel (16.9km in length) I've got a reply from Bruno. Bruno? Guess who? He is the guy who ran the above mentioned retro computing exhibition. A few emails went back and forth and on Wednesday Oct 6th at around 6 p.m. - while my family met our friends for the dinner - I met Bruno at his home in Varese. We had never met before but as we were about the same age and experienced a similar computing path since our youth we had an easy talk and time was flying. I have to say that Bruno has a very wide Sinclair knowledge but he's more a Spectrum than a QL expert.

QUO VADIS
DESIGN

Independent Information
Technology Services

www.ql-qvd.com

QUO VADIS DESIGN Independent Information Technology Services

QL/QDOS/SMSQ/E Software

Home Products Support Company Contact

Welcome

Quo Vadis Design sells software for the Sinclair Quantum 16-bit computer (Q1) and its state-of-the-art 128 called SMSQ/E.

The QL is a computer in its 20th year Anniversary.

THE Q1 QL is a computer in its 20th year Anniversary.



And you can run it on the QL, or you can run it on a PC/Mac with Windows/Linux or Mac Operating systems.

News

- * QVD QL News Blog - keep up to date [News Blog](#) a programming
- * Quo Vadis Design Website Launched in January

FEATURED PRODUCT



BUY NOW

Copyright © 2010 Quo Vadis Design. All Rights Reserved.

Home Products Support Company Contact

Bruce@ql-qvd.com

Quo Vadis Design
38 Derham Gardens
Upminster
RM14 3HA
UK

Tel: +44 (0)20 71930539
Fax: +44 (0)870 0568755

Check the QL News Blog on
our website for updates.

www.ql-qvd.com/blog

QL Today The QL magazine
for all QDOS, QL,
SMSQ ... users!

Subscriptions taken online

these versions I used to use EX NET_PEEK expecting PROGDS\$ to be set to the directory containing the program. One day, I was experimenting with C68, for which I had set PROGDS\$ to the directory WIN1_C68_. It annoyed me when I typed EX NET_PEEK and got the infuriating message "Not found". I had to laboriously type EX WIN1_SYS_NET_PEEK to get the program to load. I then remembered that another program, much used by me, could be loaded by using EXEP. Typing EXEP QD always produced another version of QD whatever the setting of PROGDS\$. I determined there and then to make NET_PEEK an executable Thing, just like QD.

Two problems had to be solved first though. I wanted to be able to call NET_PEEK either as an ordinary executable program by EX or as an executable Thing by EXEP. To set NET_PEEK as a Thing I needed to LRESPR it. The first problem was how to program NET_PEEK so that it could distinguish between being LRESPRd to become a Thing and being loaded by EX. The second problem was how to make NET_PEEK into a Thing.

Problem 1 – LRESPR or EX

The first problem is easy to solve. If NET_PEEK has been LRESPRd the program ID will be that of either master BASIC or that of a daughter BASIC, if SMSQ/E is in operation. If the program ID is zero then NET_PEEK has been LRESPRd via master BASIC. If the long word just before the position to which A6 points is "SBAS" then we are in a daughter BASIC. In all other cases it is reasonable to assume that the program has been started by EX.

As it happens, I do not allow an LRESPR by a daughter BASIC. This is because if the daughter BASIC is removed, then so will the NET_PEEK Thing. To give NET_PEEK immortality we need master BASIC to do the LRESPRing.

Problem 2 – How to Make NET_PEEK a Thing

The manuals tell us that we need a linkage block defining NET_PEEK as a Thing and that we also need a Thing Header.

Linkage Block

The linkage block consists of 42 bytes of information followed by a string containing the name of the Thing. The long word at position \$10 is a pointer to the Thing itself. The manual states that the "linkage block must be in the common heap. ." This block is linked into the Thing list by sms.lthg, the description of which also states that the linkage block "must be allocated in the common heap . ."

Thing Header

The format of an executable Thing's header is:

Item	Address	Length	Value
THH_FLAG	\$00	4 bytes	"THG%"
THH_TYPE	\$04	long	1 (for executable code)
THH_HDRS	\$08	long	offset to start of header
THH_HDRL	\$0C	long	size of header
THH_DATA	\$10	long	dataspace
THH_STRT	\$14	long	offset to start of code or 0

Notes

1. The offsets are calculated from the start of the Thing Header.
2. If a program does not alter its code it is said to be re-entrant. In this case it is possible to have several instances of the program in force at one time with all of them using the same code. Each of the programs will have its own entry in the Program Table and its own Header followed by the few bytes at the start of the program containing its name. After this comes the dataspace. The alternative is to have multiple copies of the whole program. The Thing Header supports both these alternatives. For the first, THH_DATA contains the length of program to just after its name and THH_START points to the single copy of the whole program. In the second case, THH_DATA contains the length of the whole program and THH_START is set to zero.

Bearing in mind that the Linkage Block had to be in the "common heap" I decided to put it at the end of my program NET_PEEK just after the Thing Header, since presumably the program would be put in the common heap by LRESPR.

End of Slight Digression

If you remember, I was just looking into LRESPR. Well, I decided to try out LRESPR on NET_PEEK to make it a Thing. This had worked well on Q40, Q60 and QPC2 to name a few platforms. Without any problems UQLX accepted the LRESPR command. However, when I typed EXEP NET_PEEK I got an error message. I think it was "Not found" but I'm not sure. My mind had just then become completely blank.

Luckily I was able to load NET_PEEK by using EX. I used that version to trace through all the Things to make sure that NET_PEEK was indeed linked into the Thing list. Since it was in the list I couldn't understand why EXEP did not work. In an attempt to find out I looked more closely at NET_PEEK's linkage block and at its Thing Header. After a bit I noticed that the item THH_HDRL, which should have been a relatively small number was incredibly large! I noticed in fact that it was the absolute address of the Linkage Block. What was this doing in the middle of NET_PEEK's Thing Header?

The code at the end of my program was as follows.

```
; This is the NET_PEEK Thing
N_THING  DC.L      "THG%",1
         DC.L      HEADZ-N_THING    ; Pointer to header
         DC.L      PRS-HEADZ        ; Size of header
         DC.L      D_SPACE          ; Amount of dataspace
         DC.L      STARTA-N_THING   ; Pointer to start of code

; This is the linkage block
TLINK    DCB.W     19,0
         DC.L      "1.00"
         HED1      <"NET_PEEK">, TLINK1
```

I found that although the size of header calculated as PRS-HEADZ above was correctly set before the Linkage Block was linked into the Thing list, it had been altered by the time linkage was complete. I traced the offending code to a subroutine called th_newth in the SMSQ/E source code. It resides in the file util_thg_usage_asm. This code quite definitely sets the address of the Linkage block into a long word six bytes earlier than the start of that block. As you can see, in my program that is just where the size of header is set in the Thing Header.

Explanation

From the point of view of an application programmer all this might seem extremely odd. But a systems programmer will see it differently. The application programmer should know that when a program is removed any of its channels still open are closed and any space allocated to it is returned to the heap. These useful facts allow application programmers to save code relying on the operating system to clean up after them. They do not need to know how the cleaning up is done.

The systems programmer has to be aware of the details of the cleanup. After all he has to do the coding of this. So how is it done? When a job is removed the operating system must go through the channel table closing all those channels owned by that job. It must also be the case that all allocations from the heap are examined and those owned by the job being removed returned to the heap. Of course, if the allocated area happens to contain a Thing linkage block it will be pretty disastrous if the area is returned to heap without previously being unlinked.

Each area allocated from the common heap has a 16-byte header. I can only find one reference to this header in any of the relevant manuals. That is the Appendix R on page 336 of Dicken's QL Advanced User Guide. However, keys_chp in the SMSQ/E source code perhaps shows more clearly what this header contains. There are two types of block, those which are owned by a job and those which are free space. The contents of the four long words of the header are slightly different in the two cases and appear to be as shown here.

Free Space

<u>Address</u>	<u>Value</u>
\$00	Length of area
\$04	Relative pointer to next free space
\$08	-1 (SMSQ/E), 0 (QDOS)
\$0C	0

Allocated space

<u>Address</u>	<u>Value</u>
\$00	Length of area
\$04	Pointer to driver linkage or 0
\$08	ID of owner job
\$0C	Address of flag byte set when space released or 0

All device drivers have a long word containing the link to the next driver and, three long words later, the address of the "close" routine. It seems clear that if space allocated to a driver's linkage block is to be returned to the heap it should first be unlinked. This can be done by the "close" routine. Indeed, when a job is removed the code returning allocated space owned by the program first calls a "close" routine if there is one.

The Thing linkage block is treated just like a device driver. Thus, when a linkage block is linked into the Thing list, the address of the linkage block is placed six bytes before the start of the block, which is what I was complaining about earlier, and also the address of a "close" routine is set three long words on from the start of the linkage block. It is this which ensures that the Thing linkage block is unlinked before the space in which it has been living, is discarded.

But all this is on the assumption that the linkage block is the first, or only, item in the allocated space. It is a pity that the manuals do not mention this important fact. I suggest that the wording in the manual should have been that the linkage block "must be the first or only item in space allocated from the common heap . . .".

Final Comments

First

I did not want to alter NET_PEEK so that it would create a Thing linkage block in its very own space. Instead I moved the linkage block down a bit by adding the line

```
PSEUDO_H DC.L 0,0,0,0 ; Pseudo header for Thing linkage
```

just before the Thing linkage block.

This allowed the linking of the linkage block to occur without damaging NET_PEEK's Thing header.

Since I did not allow NET_PEEK to be LRESPRd from any BASIC but the master, I knew that the owner of the linkage block would never be removed so that the unlinking would never need to occur by space being returned to the heap.

Second

Having altered NET_PEEK in the way I have mentioned I loaded it into UQLX and found that it both LRESPRd to a Thing and also came to life by EXEP NET_PEEK. I assume that the very large space erroneously assigned to the header by the Thing linking had made it impossible to set NET_PEEK as a program started from the executable Thing called NET_PEEK.

Last

This all started because of the cry "LRESPR doesn't work". Well, so far that has not been explained. I conjecture that the use of an early ROM in the particular UQLX meant that LRESPR failed because some jobs had already been loaded which means that the resident space is no longer available. Of course with SMSQ/E we do not have that problem.

Kaiser-Wilh.-Str. 302 D-47169 Duisburg
http://SMSQ.J-M-S.com SMSQ@J-M-S.com

QMENU Version 8! XMAS-OFFER!

It has taken a long time ... but here it is: **QMENU Version 8** and **The Menu Extension Version 8**

Most Pointer Environment users already know it: the Menu Extension. It is an interface which provides ready-made menus like file-selector boxes, simple-choice-menus or select from a list. QMENU is a guideline how to use it from BASIC, Machine code or maybe other programming languages which allow Machine code interfaces. It explains how to use it with various examples in BASIC and Machine code. You are allowed to use it in your own programs and you may even sell it under license. The Menu Extension also contains the Scrap Extension ("clipboard").

Multi-column menus, file-select with tree and view option, FileInfo II support - just the FileSelect menu on its own is a beautiful extension to your system.

QMENU has not been advertised for quite a while, as the last version 7 manual was not updated in the past few years, while the Menu Extension itself got updated here and there. However, many updates in the Menu Extension and several user inquiries made me think about releasing an updated version of **QMENU**. The manual has been completely revised and reflects all the minor and major changes and add-ons: from the assembler-side, from the BASIC programming side, and also from the user's side. You get a 42-page printed manual, a floppy disk with updates keys, updated help texts for QD Hyperhelp and updated and new examples.

Please note: The Menu Extension from version 7.65 onwards works only under SMSQ/E V2 (e.g. QPC2 or systems with high-colour screen drivers). If you run the "old" QL Pointer Environment, you should stick to your old Menu Extension. English only (a German version of MENU_rext is also on the disc, but no German documentation).

Some of the changes since version 7.04 (the last "officially" documented one) are:

DSEL (Directory Select) allows up to 10 devices

RSTR (Read String) has additional parameters (which force the values entered to be ints, floats, not empty, disables ESC etc.) It can also be used to enter hidden passwords.

Timeout feature has been added to RPER (Report Error) and ITSL (Item Select).

Some menus have got a MOVE facility.

New menu SYSS (System select) provides fast selection of items from the Hotkey buffer history, currently running jobs, Things in your system, Executable Things in your system). Just one call and the System Select procedure collects all the information for you and provides it in a list - very easy selection. Hotkey buffer history now available in the file-select instead of cycling through the "previous" ones.

All this, bug fixes and more - available NOW.

To order, please send letter, fax or E-Mail or place an order through the secure order form on **SMSQ.J-M-S.com** (you will find screenshots on the website too).

Special XMAS offer, valid until 31st of January 2011:

QMENU Update: EUR 15.90 including postage (instead of EUR 19.90).

New product slightly delayed! J-M-S advertised something new to arrive in November in the last Quanta magazine - unfortunately, it seems, it will take a bit longer. More on the homepage when ready!

We accept VISA, MasterCard & Diners Club online and offline!

New payment methods for our customers: Money transfer to "local" account in many countries!

• Deutschland: Jochen Merz, Account 493 50 431, Postbank Essen, BLZ 360 100 43

• Österreich: Jochen Merz, Account 85055317, PSK Wien, BLZ 60000

• Switzerland: Jochen Merz, Account 60-690080-4, PostFinance, Clearing-Nr. 09000

• The Netherlands: Jochen Merz, Gironummer 3258439, Postbank NL Amsterdam

• and from all other countries in EUR with IBAN and BIC to account

Jochen Merz, Deutsche Postbank AG, IBAN: DE21 3601 0043 0611 1004 37 / BIC: PBNKDEFF 360

• UK customers can pay in £ (convert EUR prices above to £ by multiplying with 0.88) to

Jochen Merz, Account 83795395, Citibank UK, Sort code 30-00-45

or send cheques in £ - no fee for UK sterling cheques!

• US customers can pay in US\$ (convert EUR prices above to US\$ by multiplying with 1.38) - no fee for US cheques in US\$!

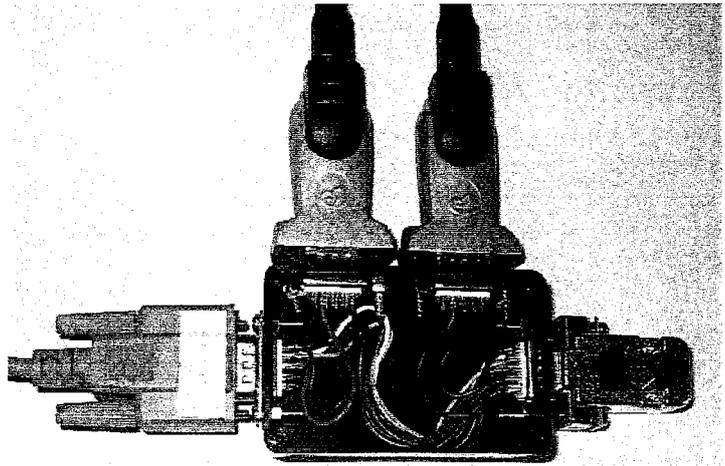
Cheques payable to Jochen Merz only!
Price list and offline exchange rate
valid until 31st of Jan 2011

RS232 Monitor Project

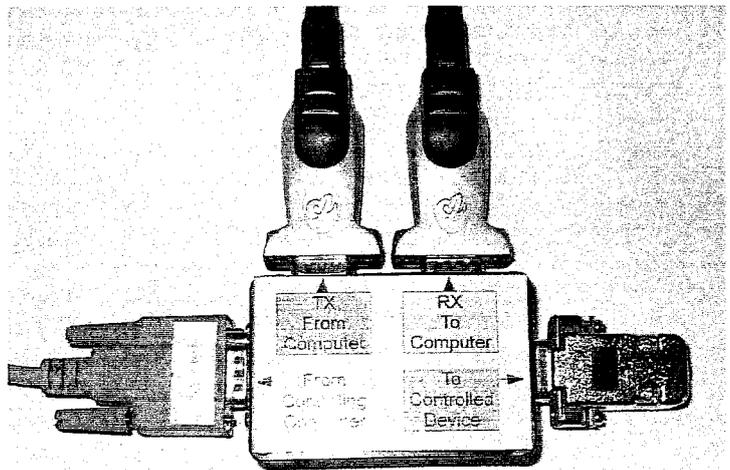
by Ian Burkinshaw

This is a very simple project to monitor the communications across a RS232 link. This could be between two computers or alternatively between a computer and some other RS232 controlled device. It checks the communications both ways at the same time, between the controlling computer and the controlled device, so that you can see the data being sent to the controlled device and the data returned from the controlled device back to the computer. It can also be used for checking or researching the protocol between devices. In fact the reason I developed this project was that I wanted to check the communications between my PC running QPC2 and an AOR7030 communications receiver that does not have the simplest protocol. I could use this to send commands to the receiver and check the returns to control if the receiver had done what I told it to do. It saved a lot of time fault finding my code, since it permitted me to run my development program and at the same time monitor the communication across the RS232 link.

So what is required? If you are using original QL type hardware then you will need two computers, because the black box original QL has only 2 RS232 ports. However if you have a SuperHermes this will be different since you could use the extra RS232 ports this option provides. This could be a new use for the QL sat in the cupboard. When you use two computers, one can be running your application, and second one monitoring the RS232 link. However, if you are, say, using QPC2, then only one computer may be required. This would depend on your software. It is possible to run your application and a RS232 monitoring program at the same time or have the monitoring routines as part of your application. Either way you will require the breakout box, detailed below. Depending on the PC computer hardware you are using, you may need USB to RS232 adapters even if your PC computer has an RS232 port, which is becoming more and more rare. If it has, it will be most likely only be one and you will need to provide more RS232 ports using USB to RS232 adapter(s). To provide the main port for controlling your device, plus two further ports for the monitoring, I run this on both my desk top PC



Finished box open



Finished box closed

and my Asus Eee PC computers with no problem.

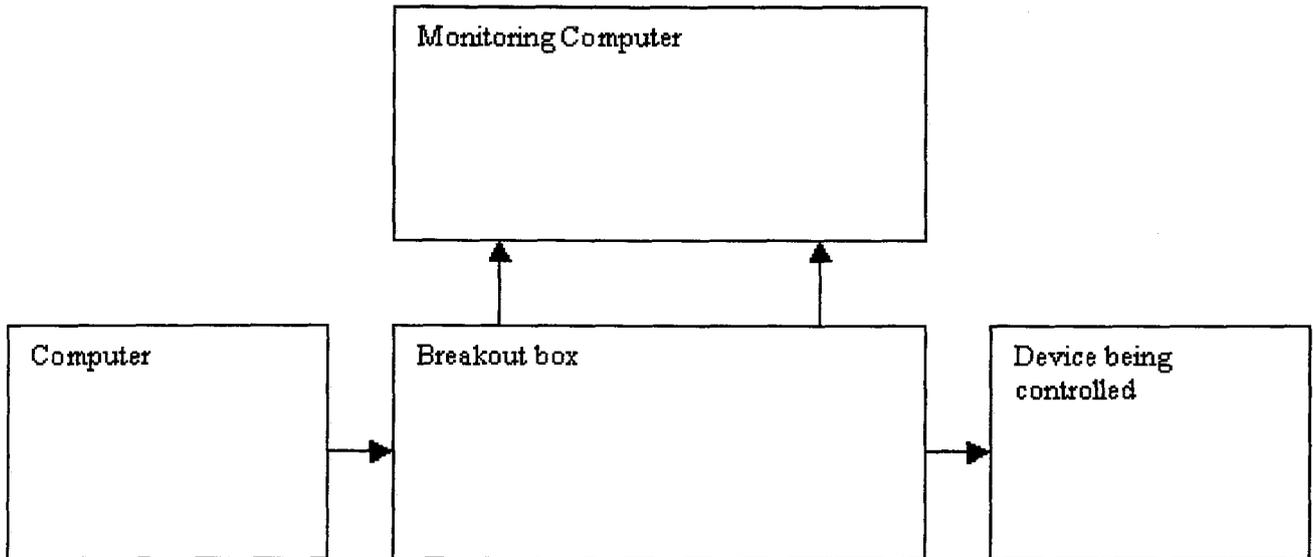
Part required and approximate cost

1 x D-Type 9 way plug, Maplin code	RK60Q	
	£1.39	
3 x D-Type 9 way socket, Maplin code	RK61R	
	£1.39	£4.17
1 x T3 box, Maplin code	KC92A	£2.19
3M Nuts and Bolts		
Hook up wire		
2 x USB to Serial Adapters, Maplin code	ZP43W	
	£14.99	£29.98

Total cost approximately **£37.73**

The above information correct at time of writing.

As can be seen this is as simple as is can get.



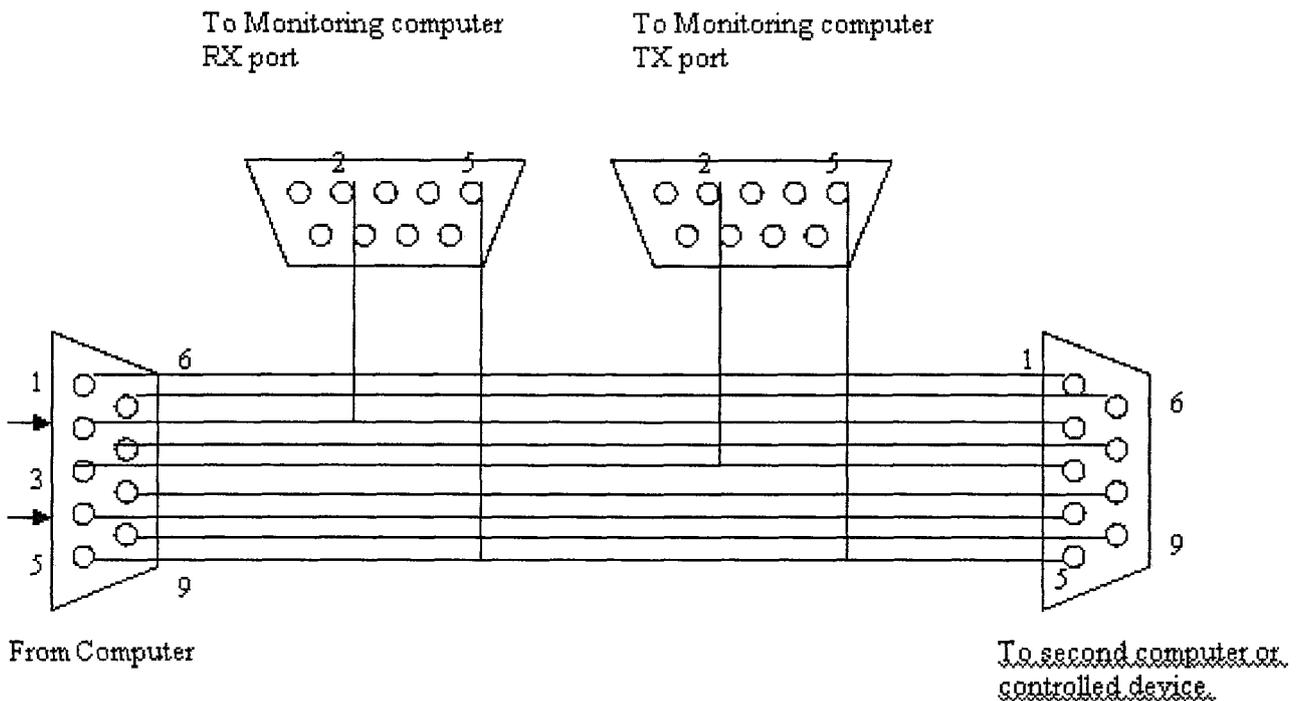
An example of how the box can be connected up is shown above. This example uses separate computers for controlling and monitoring, but this need not be the case if you have more than 3 RS232 ports on your computer.

As can be seen from the circuit diagram, all connections are between the computer and the controlled device, with a spur for the serial data transmit and receive lines. This means if the hardware hand shake is used between the computer and controlled device this is not affected.

So how does this work. Very simple, we monitor pin 2 [RXD (Received Data)] and pin 3 [TXD (Transmitted Data)] of the main path from the controlling computer to the controlled device, on two

RS232 ports, one monitoring the RXD and the other TXD. So we just use the RXD on the two RS232 monitoring connectors.

The break out box has 4 x 9 pin D-Type connectors. 3 female 9 pin D-Types and one male 9 pin D-Type. One male and one of the female connectors are connected together pin for pin. So that is pin one to pin one, pin two to pin two and so on. Pin five is the ground pin. Pin 2 is the receive data pin and pin 3 is the transmit data pin. The two remaining female 9 pin D-Types have ground connected to pin 5, and one connector pin 2 is connect to pin 2 on the main pass connector and the second remaining male 9 pin D-Type pin 2 is connect to pin 3 on the main pass connector. See the diagram below.

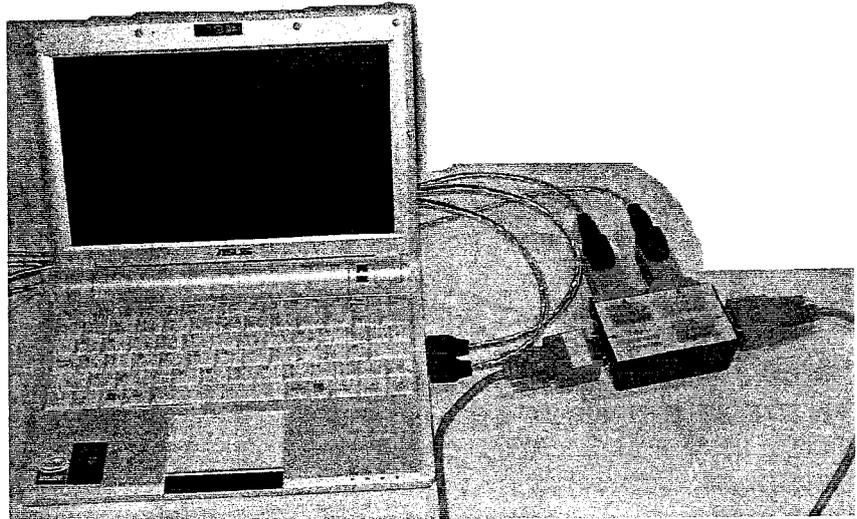


You will see from the basic program shown below, that we open two channels one for each monitoring port. It reads the inputs from these ports and then prints out the data. That is it for a very basic monitoring operation. In some applications you may wish to show the data in Hex or Binary form which is very easy to do.

```

20 OPEN#4;ser1:REMark open serial port, change port number for your system
30 BAUD#4,9600
40 OPEN#5;ser2: REMark open serial port, change port number for your system
50 BAUD#5,9600
60 REPeat loop
70 q$=INKEY$
80 IF q$=="q" THEN EXIT loop:REMark quit loop and program
90 t$=INKEY$(#4)
100 PRINT#1;t$;
110 r$=INKEY$(#5)
120 PRINT#2;r$;
130 END REPeat loop
140 CLOSE#4
150 CLOSE#5
160 STOP

```



Monitoring system sep-up

The following program is a more advanced monitoring routine with two windows displaying both the TXD and RXD data, in decimal and hex form. This program also has some error trapping.

```

10 REMark RS232 monitor Feb 2010
20 setup_serial_ports
30 setup_screen
40 run_monitor
50 CLOSE#3
60 CLOSE#4
70 STOP
1000 DEFine PROCedure setup_screen
1010 xsize=SCR_XLIM
1020 ysize=SCR_YLIM
1030 WINDOW#0;xsize,ysize,0,0
1040 PAPER#0;0:INK#0,7:CLS#0
1050 WINDOW#1;xsize-20,(ysize/2)-20,10,10
1060 PAPER#1;2:INK#1,7:CLS#1:BORDER#1,2,255
1070 WINDOW#2;xsize-20,(ysize/2)-20,10,(ysize/2)+10
1080 PAPER#2;3:INK#2,7:CLS#2:BORDER#2,2,255
1090 AT#0;0,2:PRINT#0;"Data transmitted by the computer to the controlled
device"
1100 AT#0;38,2:PRINT#0;"Data received by the computer from the controlled
device"
1110 END DEFine setup_screen
1200 DEFine PROCedure setup_serial_ports
1210 INPUT#0;"Input serial port number that is monitoring transmitted data
from computer to controlled device :";portt
1220 portt$="ser"&portt
1230 PRINT#0;"Input baud rate for port ";portt;" ,this the rate being
transmitted from the computer :";:INPUT#0;"";porttb
1240 test_baud_rate porttb: IF baud_ok=0 THEN GO TO 1230
1250 INPUT#0;"Input serial port number that is monitoring received data to
computer from the controlled device :";portr

```

```

1260 portr$="ser"&portr
1270 PRINT#0;"Input baud rate for port ";portt;" ,this the rate being received
from the controlled device :";:INPUT#0;";portrb
1280 test_baud_rate portrb: IF baud_ok=0 THEN GO TO 1270
1290 BAUD portt;porttb
1295 OPEN#3;portt$
1300 BAUD portr;portrb
1310 OPEN#4;portr$
1330 END DEFine setup_serial_ports
1400 DEFine PROCedure test_baud_rate (check_baud)
1410 baud_ok=0
1420 IF check_baud=300 OR check_baud=600 OR check_baud=1200 OR check_baud=2400
OR check_baud=4800 OR check_baud=9600 OR check_baud=19200 THEN baud_ok=1
1430 IF baud_ok=0 THEN PRINT#0;"Incorrect baud rate, baud rate must any of the
following supported rates, 300, 600, 1200, 2400, 4800, 9600 or 19200"
1490 END DEFine test_baud_rate
1500 DEFine PROCedure run_monitor
1510 REPEAT main_loop
1520 q$=INKEY$
1530 IF q$=="q" THEN EXIT main_loop
1540 t$=INKEY$(#3)
1550 IF t$<" THEN PRINT#1;t$;
1555 IF t$<" THEN INK#1;0:PAPER#1;4:PRINT#1;"(";CODE
(t$);)";:INK#1;7:PAPER#1;2
1556 IF t$<" THEN INK#1;0:PAPER#1;5:PRINT#1;"[";HEX$((CODE
(t$)),8);"]";:INK#1;7:PAPER#1;2
1560 r$=INKEY$(#4)
1570 IF r$<" THEN PRINT#2;r$;:r=CODE(r$)
1575 IF r$<" THEN
INK#2;0:PAPER#2;4:PRINT#2;"(";CODE(r$);)";:INK#2;7:PAPER#2;3
1576 IF r$<" THEN INK#2;0:PAPER#2;5:PRINT#2;"[";HEX$((CODE
(r$)),8);"]";:INK#2;7:PAPER#2;3
1580 END REPEAT main_loop
1590 END DEFine run_monitor
32000 DEFine PROCedure UPDATE
32010 SAVE win1_RS232Monitor_RS232Monitor_bas
32020 PRINT "Update Complete"
32030 END DEFine UPDATE

```

Next time I will look at how you can get a I2C port running on your QPC2.

Letter-Box

George Gwilt writes: Letter to QL Today re **Norman Dunbar's Article Easy PEasy Part 2**

I am most grateful to Norman in his article Easy PEasy Part 2 for having corrected and improved the code and comments which appear in my example EX0_ASM. However, there are two comments I would make.

The first concerns the statement towards the middle of page 34. "This code will not return unless an action routine sets D0 with an error code or sets D4 with an action number." It is true that the code will return if at least one of D0 and D4 is non-zero. However, the code might return as a result of an event key being pressed and not as a result of an action routine. If an event key is not set to select a loose item then pressing it will cause an event and the code reading the pointer will return with D4 non-zero. If that key does select a loose item then whether or not a

return occurs depends on the action routine for that loose item.

[Norman replies: I agree. My wording is wrong/incomplete. I should have been clearer on the event situation]

As it happens, in EX0 the event keys for both move and resize select their respective loose items. Furthermore neither of the action routines causes an exit from the reading loop. The moving and resizing take place entirely within the action routines. As a result of that in EX0 the code checking for a move or size event is superfluous because you could never arrive there. However, that code would have been needed if it had been deemed necessary for the event to be activated by an event key which did not select the appropriate loose item.

The key F1 causes a help event so pressing it will cause an exit from the reading loop. However,

since that event is not checked in the code at no_err, the program simply branches back to the reading loop. You can perhaps detect this happening by placing the pointer on a loose item then pressing and holding down F1. You should then see the pointer disappear.

The second comment refers to Norman's implied suggestion on page 36 that there might sometime be a supplied routine to perform a resize as there is for move and sleep. There are several reasons why it is not practical to produce a single routine for resizing.

[Norman: What can I say? Now that George has explained the joys of resizing a window, the reason why there is no generic resize routine become obvious. Thanks George]

The first reason concerns the position of the move loose item. In EX0 it is at the top right corner of the window. There are three other corners where it might have been placed. In each of these cases there would have to be a difference in the coding just before and just after "resze" on page 38 if it is the rule that the opposite corner remains stationary. When a resize takes place it is either the right or left vertical edge that moves and it is either the top or bottom horizontal edge that moves. If it is the left vertical edge that

moves, then we find the x-value of the new position of the pointer by subtracting the change in size from the original position. Otherwise the original position is unchanged. Similarly if the top edge is moved we subtract the change in size from the original y-value.

I suppose it is conceivable that someone might want to put the resize icon in the middle of the window and leave the midpoint of the window unchanged on resizing. This would entail subtracting half the change in size from the x- and y-values to find the new pointer position.

The second reason for not having a general resize subroutine is that in some programs, as happens for example in QD, you might want a greater step in changes of size than the minimum I have chosen for EX0. Again this means a change of coding.

A third reason is that you may want more significant alterations in the window depending on its size. One example of that is a window allowing scroll bars to move the contents if the window is too small. For both x and y therefore you will either have scroll bars or not depending on the relative sizes of window and contents. That is something I would most certainly not want to put in a general resize routine.

Last-Minute News from Dilwyn

PUZZLE

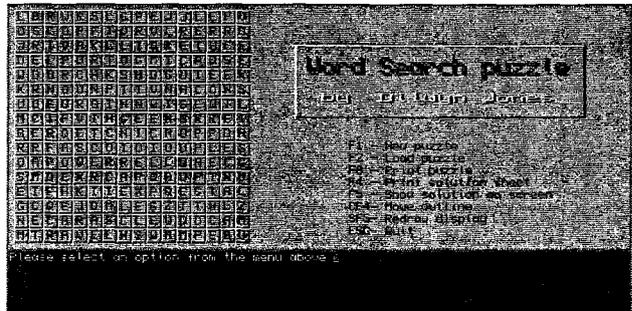
Puzzle is a computer version of the well-known sliding block puzzle we probably all had as a child. The program displays a 3x3, 4x4, 5x5 or 6x6 grid of letters or numbers, with one blank square.



The aim is to get all the characters in order with the blank square at the bottom right. Just click on the square to move and it will slide into the blank square next to it. Keep doing this until you've got them all into order. Not as easy as it sounds! Needs pointer environment and Window Manager 2. Available to download as freeware from <http://www.dilwyn.me.uk/games/index.html>

WORDSEARCH

Another little word puzzle to while away the winter hours. Using its 11,000 word dictionary, Wordsearch creates a grid of hidden words which you have to locate. Can print out the puzzle itself of course, plus a solution sheet telling you where all the words are hidden if you really need the solution. Words can be hidden horizontally,



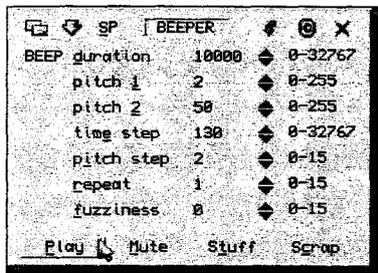
vertically, diagonally and even backward... just like those fiendish word puzzles you get in magazines but without the cost of buying the magazine! Puzzles can be a basic 4x4 grid, or up to 16x16. Be prepared to lose a lot of your time with this fiendish little game. Available to download as freeware from

<http://www.dilwyn.me.uk/games/index.html>

BEEPER

A simple little pointer driven utility to help you experiment with BEEP parameters. While the QL BEEP command does not make for the most advanced sound generator ever, you can still produce some interesting sounds if you are prepared

to play around with the various parameters a little. Use the PLAY button to hear the sound, then once you've got the sound you wanted, the relevant



BEEP command can be copied to the Stuffer Buffer (retrieve from BASIC with an ALT-SPACE keypress) or even to the SCRAP system if you are in the habit of using QD to edit your BASIC programs, for example. BEEPER is pointer driven and needs Window Manager 2. Available to download as freeware from

<http://www.dilwyn.me.uk/sound/index.html>

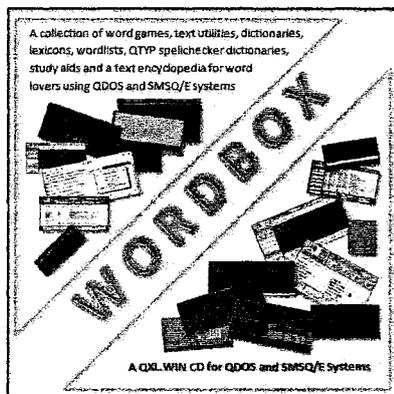
MARQUEE

This is just a simple little text scroller, like those noticeboard displays you get in shops. Written originally just for my own use, I happened to show it to someone and they said I should make it available to others. While notionally just used as an advertising display, it can be used for any application where a scrolling text display is needed. You can configure the window size, text size and scrolling speed to get the display you want. Can be used as a simple information display alongside other programs, since it doesn't need much space on the screen. The text can be stored in a file or passed directly to the program, and it has some simple built in examples, such as scrolling the date/time across the display and a countdown of the number of days to Christmas! Available to download as freeware from

<http://www.dilwyn.me.uk/misc/index.html>

WORDBOX

Brand new CD for the QL, featuring an extensive collection of games, utilities, editors, lexicons, wordlists and spell checker dictionaries together with some educational programs, a dictionary program and even a text based encyclopaedia. The plain text wordlists and QTYP spell-checker dictionaries include most European languages, including German, French, English, Spanish, Italian, Swedish, Danish, Norwegian, Dutch and Maltese. Oh, and



some of the Celtic languages too! A dictionary program on the CD lets you translate words between English and these languages. There's even an English grammar course on the CD and even a guide to learning Latin! Not to mention a style checker, rhyming dictionary, thesaurus, indexing programs, word count utility, crossword aid, anagram aid, Scrabble TM dictionary aid, typing tutor and much more. In other words, everything a word lover and writer could want. The CD should be available from Dilwyn Jones and from the Quo Vadis Design website by Christmas. Further information on my website at

<http://www.dilwyn.me.uk/cdandsoft/index.html>

REPLACEMENT MANUALS

Thanks to Rich Mellor, I have been able to add two further replacement manuals for QL add-ons to the replacement manuals page on my website. As it is so common to buy second-user hardware without getting a manual, this collection of manuals is very useful. The first is a replacement manual for a PCML Q+ Disk Interface, which includes the documentation for the PCML toolkit v1.14 / v1.16, which includes Toolkit 2 style extensions, the facility for direct sector access, ramdisks and hints on getting the Psion QL programs to run from a floppy disk. It also shows how early games able to run on 128K systems may be run without having to unplug the disk interface card. This is only available as a 17 page PDF file. The second is a replacement manual for the CST RAM Plus card, which consists of 512K RAM for the QL, plus up to 4 EPROM sockets, shown as "27256" EPROMs. The manual notes that to use EPROMs, a JS, MG or later version of the QL ROM is required (earlier versions could only handle one expansion ROM unless you had specially written code to help the startup routines). This manual is also only able as a PDF file. Both can be downloaded from the Replacement Manuals page on my website, at

<http://www.dilwyn.me.uk/docs/manuals/index.html>

Urs' Puzzle - the Solution

<http://www.coww.clivl.futura.html>

1. What other name (*) for the QL was in the talks for the ZX83?	Q U A N T A
2. Company who made the Liberator SuperBasic compiler?	L I B E R A T I O N
3. Sign to be used to terminate a QDOS directory device name?	-
4. Family name of founder of GSI?	F E N T O N
5. Rampton based Company of A.J. Tobby serving the QL market?	Q U M P
6. Name of the first available QL successor after the Amstrad sell-out?	T H O R
7. Company who made the first QL emulator board for the ATARI ST range?	F U T U R A
8. First application to use the QL Pointer Environment?	Q R A M
9. First name initials of directors of SANDY U.K.?	A



The big question mark again?

... but not as big as last time because we know there will be a Quanta AGM in April, but we have not heard of any ideas for an international continental show.

And - rather disappointing - no replies from (potential?) visitors to all the questions posed in the last issue, page 45: *"Can we have your feedback please? How is your interest in meetings? Who is prepared to "do" something (find a location)? How do we reach QLers in the future? How do we find out how much interest exists in certain venues?"*

Sjef v.d. Molengraaf has replied, that he thinks it could be possible to have another meeting in Holland again ... that's great news, but will we have any visitors?

Shall we interpret the total lack of replies as no interest in any kind of meetings?

Another (final) call for replies ...?

**THE
QL-TODAY
TEAM
WISHES OUR
READERS
HAPPY NEW YEAR
AND ALL THE BEST
FOR 2011!**

