FROBNICATE

Acorn **# Do you remember me, lost for so long... #** IYONIX pc

OVATION Pro

Hissing Spinach

Spring 2004 Issue 20 €0

Index:

Credits:

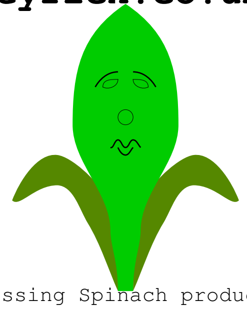
Designed, written, and created by Richard Murray.
The "Iyonix" article was written by Ron Briscoe.

The Frobnicate website – where you can also find the previous 19 issues...

http://www.heyrick.co.uk/frobnicate/

This is the PDF version of Frobnicate. Because it is accessible on-line to those using PCs, and because search engines such as Google support a PDF to HTML conversion, *all* email addresses have been munged with "|-ay-tee-|" for the '@' symbol.

Thank you for your understanding.



A Hissing Spinach production
© 2004 Rick Murray

Keep in touch!

heyrick|-at-tee-|bushinternet.com

Editor' s Note

Well,

my goodness!

I'm not quite sure *where* to begin! It has been nearly ***half a decade*** since the last *Frobnicate*. Seriously! Issue 19 is dated Spring 1999, this one is being released only a few months before springtime 2004. The release date of this issue is Wednesday 28th January 2004 – I'll be 11,000 days old! Also, I can tell you it has been 1732 days between the two issues of *Frobnicate*. [actually, it'll be released on 2004/03/03 with 1762 days between issues, and I'll be 11030 days old]

I am writing this issue for two reasons. The first, it has *always* bugged me that I stopped writing with issue 19, I feel I should have at least made it as far as issue 20. Also, *because I felt like it*. Those of you who were around in the early days (June 1995!) will remember that Ian Kershaw put forth in the ARCHIMEDES echo (yes, Fidonet – remember that?) the idea of an enthusiast-written magazine to be created by the enthusiasts. You might ask "Well, what about Archive?". It's a reasonable question, however archive was a printed 'commercial' magazine. You paid for it. This, instead, was to be written by people with some free time. They share what they know, somebody coordinates it all, and we all benefit. Because I happened to have a lot of free time, and I'm an immensely boring person (hey – I'd *love* to take a running jump off of a mountain with nothing but a parachute attached, but nobody has considered me worth asking!), I decided to step in. Ian was to create the magazine, but apparently he did not receive enough support, which I think is a real shame. That didn't stop me, I 'threw together' the first issue in a matter of hours. And, thus, *Frobnicate* was born.

FROBNICATE... JUNE 1995

Issue 1. The ULTIMATE text-e-magazine for Acorn Enthusiasts.

A BRAND NEW MAGAZINE

Frobnicate Get it right guys!!

In this issue:
- Build yourself a modem to modem link.
- Useful C routines.
- Hacking into protected BASIC programs.
- Hook your house into your computer.

Through the years I put an awful lot into *Frobnicate*. I have quite an imagination, which led me to create such characters as the (demented) hacker. It was almost a necessity, as programming is fun – but you can't stare at code all day long otherwise they'll put you in an incontinence nappy, stuff you full of pills, and leave you to watch the atrocities that pass for children's television these days. I worked two and a bit years as a Care Assistant (in nursing/residential homes), so trust me – I know...

Because of my imagination, the lack of support wasn't as big a problem as it might seem to have been. That's not to say I had no support; even without looking at the back issues, I remember Dizzy Wizard's *Finishing Touch* series, Nava Whiteford's amusing and interesting *Ants*, and Niall's description of a pre-emption system called *Tornado*. Along the way we 'endured' the *Invisible SysOp debate*, an article about how *CDTracker* came to be, Helen Rayner wrote about *Females in computers* (some people took that title a bit too literally and asked "how do they fit, or do you mean mainframes"!), my own mother wrote a thought-provoking article which is as valid today as it was all those years ago (unlike most of the crap that I write!), Glenn (my good friend) wrote about using *Econet* and about how to get some decent sound from the RISC OS machines. If the MDFS hadn't toasted half of my network, I probably still would be using Econet; instead now I'm using 10baseT to my "real" PC – yeah, the same one I had back in issue 11. Nava came back to write about *SWIs* at a low level, I reviewed Argonet (and spent many years in their care). Emlyn Owen (where y'at dude?) provided some lovely rendered front-cover images. And in the most recent issue before this, Paul Roegele contributed an article about using LCD modules with the heart-warming introduction:

> *Right, well, after recently discovering Frobnicate and printing every issue currently available, I decided that I really should contribute.*

You see, that is what it is all about! There's no money in this, I don't pay you for your work, and you don't pay me for *Frobnicate*. Instead, we all share stuff. It's one of my fundamental beliefs, and probably the reason why, at the age of thirty (yaaaah!) my bank account is a mess, still. So maybe I'll die unmarried with no kids to keep the Murray name going... At least, as I rest peacefully in my coffin (spare me the "heaven" rubbish), I can look back and think "I didn't sell out".

Over the years, people have asked me why I chose to use *Ovation* for the magazine, and why if it was to be an electronic magazine, I didn't write it like one. In fact, some people were downright nasty about it.
The answer is a multiple one. Here's the first reason. I *loathe* on-line magazines. You cannot imagine how much I think it sucks to have on-line user-guides. When I'm at the computer, I want to get stuff done. When I'm not, I'm either watching TV or reading. Yesterday I read the 65C02 datasheet, mostly for amusement. If I had to read a PDF on-screen, I'd scream. That is why *Frobnicate* is written as a traditional magazine. You can then do what I do and toss the lot to the printer. Quiet, simple, minimal fuss, and hey – you can read it in bed!
The second reason. I have always been a fan of *Ovation*. Something about it just 'clicked' in my head. Over the years I have tried *every* version of Impression from ImpressionII up to Impression Publisher, and while it was an 'impressive' piece of software, it just didn't feel right (it made an impression on me, ha-ha `<thunk>`). I could see a future with *Ovation*, and, god, who the hell needs rubbish like Word and Publisher when here you have one program that does it all, quickly and easily. *Just the way I like it.*
The other day I discovered a "Power To The Programmers" floppy disc. I popped it into my RiscPC and it crashed. Why? I don't know, nor am I *that* interested to find out. That's the thing with all of these software-based "magazines". Back when I started with *Frobnicate*, RISC OS 2 was still a proposition. People didn't like to see letterboxed TV modes on their VGAs and other people didn't like to see nothing because they only had TV modes. So I decided that, in the first instance, the bottom line would be somebody else's fault and not mine (so, the base spec was whether *Ovation* would run on the machine!), and in the second instance, I didn't want to write a line of code for some stupid on-screen thing. If my entire audience choose to read *Frobnicate* on-screen, it's their decision (most, apparently, do!). I print mine out.
By the way, where is Impression now? I hear some company is trying to make a 32 bit conversion of Impression Publisher for the Iyonix. I wish them luck, but I do wonder if the time hasn't come to consign Impression to the same place we put 1stWord+ all those years ago. Certainly, a 32bit conversion of ArtWorks is useful as we don't have anything quite like it, but Impression? Who needs it! *Ovation Pro* does it better...

Don't look back in anger. That Power To The Programmers "e-zine" doesn't work. *Ovation* is still going strong (come on people, APDL were selling it for about 10 pounds, last I saw). David Pilling, the one who deserves serious respect, has since created *Ovation Pro* which... I'm using it now, and have been for a year, and it still impresses me. But, you know what *really* grabbed my attention? It's the Windows version of it! God-damn! This thing runs at a *usable* speed on my RiscPC co-processor (basic 33MHz 80486SX). I've not had it running full speed on my Pentium as all interfacing is done via VNC which makes my real PC and the emulated one appear to work about the same speed (!), however I get the impression that *Ovation Pro for Windows* runs faster than Windows 95 (OSR 1½ – don't ask) itself does. It is quick, it isn't memory hungry, and the full installation (minus stuff like spell check dictionaries) will fit on a floppy disc. I laugh at the pathetic efforts of other huge corporations with their stupid American mindset (i.e. if we throw enough resources at a problem, we'll overcome it). *Ovation Pro* rocks. Oh yes.
This issue has been written using *Ovation Pro*. You can get a free demo version, you can buy the real thing (if you have not already). If you are a Windows user, right now David is offering *freely downloadable* demo versions of OPW (fully functional, only with a built-in time-out) ... really, there is no reason why you cannot view this issue of *Frobnicate* in some version of *Ovation Pro*.
Okay, there are two reasons – either you're a raving Linux user who never saw the point of 'infecting' a good computer with WINE and the crap it was designed to run; or you're just lazy and can't be bothered to find *Ovation Pro*. For you, my friends, there is the PDF version of this magazine.
I'm sure, these days, people will ask "why didn't you use HTML". It's a fair question, but I was on-line in an era before HTML (gasp! for real!). When Fidonet with its lag was *the* way we all communicated; we all knew each other (something I liked about Argonet), and I've had +£600/quarter phone bills to BBSs (I'm a download junkie).

The internet was a boon for three reasons:

1. I could do many things at once. It wasn't unusual to find me downloading three or four things, uploading something, and writing a message on-line. It's one of the little things that made the 'net better than BBSs.

2. The other reason? Local-rate phone calls. I have spent many a happy hour talking to Dave and DaviD, John Stonier, Steve Pursey, Keith, Chris, and all of the other sysops. Sadly, it was less happy when the phone bill was deposited. British Telecom actually hired a fork-lift truck for that.
3. And, related to the above, the destruction of global boundaries and the speed of things. You can now download the latest version of *OvHTML* directly from me instead of my uploading it to the main BBSs and waiting for others to pick it up.

This isn't to say that BBSing was a bad thing – if it wasn't for the efforts of Dave, DaviD, Steve, John, Daniel Aston, and Graeme Reed; I wouldn't be here today. These people taught me something important, that you don't need to know a person to be friends (a Dave Coleman presents stuff on Radio Four, and I think I recall Dave Coleman from Arcade saying he worked for the BBC – same person? I have no idea). And it was decided that I would be involved in the comms scene.

So there are over *four and a half years* to catch up on. Well, Acorn is dead. The little company that had so much to offer was unable to sustain itself, and part of their parting "puff" was *apparently* to try to blame a newsgroup for supposedly slating a product that hadn't even been released yet – jeez, why doesn't stuff like that ever happen to Microsoft? We can all get together and slag off... well, everything actually. The CEO can sulk, and the company can close. I bet we'd have an easy-to-use Linux within a *week*, and the whole decade of the Microsoft Empire can be called... I dunno, The Dark Ages... Anyway, that was it. No more Acorn. Hasta la vista, baby.

Image from FRU 11

Somebody somewhere had been quite clever. The real baby of Acorn was the ARM processor. Yes, RISC OS is lovely, but the ARM is what wins. It was spun off into a separate company. Now you'll find ARMs all over the place. There is a strong chance that *you* have an ARM processor in your living room – do you have a Sky digibox? Do you, or any of your kids, have a Gameboy Advance? Maybe there's a small Thumb version inside your mobile phone. Recently, Intel (who fabricate it) said that more ARM silicon is shifting than x86 based silicon. Well, waddaya know?!

As some sort of connection (some of the Sky digiboxes are built by Pace), the company Pace bought some of Acorn's remains. One thing included in the package was RISC OS. It is my suspicion that, originally, Pace weren't even really aware of that, and when they were they knew they had something special but didn't know what to do with it. They designed the Bush Box, which was technically quite impressive, but marketed wrongly and (sadly) lacking some of the features that'd make it truly useful. I have had *Ovation Pro* running on the Bush Box, it *can* run stuff in the desktop environment. If only they'd thought of it as a cut-down computer instead of taking such pains to hide the computer element!

Image from Mico website

We were all promised a bright new future post-Acorn. What happened? The Mico. A computer based upon the ARM7500 processor (same as the "Bush Box"). It was good to see because it gave some reassurance that the scene wasn't dead after Acorn died (as was the case with the Amiga – but try getting an Amiga advocate to admit *that*). Then, arrived a new StrongARM machine that could take an XScale as a co-processor. They called it the Omega. Along the way, RISC OS Ltd was formed to release RISC OS 4. I have not upgraded, but I am assured that some of the late work on Phoebe made it's way to reality in the form of RISC OS 4. Then the Select scheme started. I'm still using RISC OS 3.70 as I've not seen that much of a need to upgrade, but that's just me.

Image from FRU 11

Image from FRU 11

One of the interesting spin-offs of the Omega design is the PCI PC card, it features a near-complete PC on a card, you simply plug a processor into it, and plug it into an Omega and it's instant PC time. That's a picture of it on the right ... cute, isn't it?

That's a non-exhaustive listing, but it shows the main players in the RISC OS market.

We welcome the Iyonix with open arms. *Finally* somebody has figured that to get life beyond Acorn, we cannot keep designing systems based around old parts simply because the later processors don't do the 26bit PC+PSR addressing mode. They've *finally* broken through and released a 32bit version of RISC OS based upon the XScale processor. No matter

Image from FRU 11

what happens now (apparently Select and Iyonix use different pedigree of RISC OS, one of the dangers of splitting off like that), ***this*** is what we have been waiting for. ***This*** is the new breath that RISC OS needed.
I'm serious, if I had the money, I'd get an Iyonix today. It is sufficiently far enough advanced as to make it a worthwhile upgrade from my previous systems. And with the amazing *Aemulor*, it is still remarkably compatible.

I wish the team behind the Iyonix *every* success for the future.

One of the benefits of writing my own magazine is that I control the content. This isn't to say I'm a control freak; I value the right to have the freedom to express opinions. Instead, it means that if I want to devote some space to slagging off George Bush and saying what an evil idiot he is, then I can do so. I'm sure some lawyer in the US is reading this and gloating. Dude – your 'wonderful' president (who wasn't even elected by popular vote) waged an illegal war and on 2003/03/20 all but killed the idea of democracy. Maybe you Americans should *get over* your obsession of it being "unAmerican" to disagree with the president. He used legal sleaze to get elected and he used legal sleaze to go to war (conflict, or whatever not-War word you prefer) in Iraq. And then his amazing brainstorm - *If you aren't with us, you are against us*. Truly outstanding! In today's modern democratic society, this dumb Southern prick (yes, I *did* just refer to your president as that) has reduced it to Us and Them while totally missing the fact that some may be completely against the terrorists, but not view the American tactics as anything better. Also, he has started a highly dangerous *war* against "terrorism". It is dangerous as, well, who exactly *are* the terrorists; lest we not forget that America was siding with the Taliban when it was acceptable to shoot "commies". And this, my friend, brings us around full circle. You might disagree with my "unAmerican" views, but... really... go stuff it. The last great "unAmerican" deal was McCarthy in the fifties, and look how *that* turned out. Should I mention Nixon? I think you get my point...

BTW, Feds, spooks, whoever – if you plan to open a file on me, get in touch if you'd like a recent photograph. My passport picture *really* sucks.

I'm tempted to say "*get a friggin' life!*"

Jeez, let's not finish on a depressing note...

So now I must talk about *Amélie*, one of the greatest films ever. In 2002, it quickly rose to number 10 in IMDb's top 250 films (of all time).

Image digitised from *Amélie* DVD (UK version)

This is the gentle story of Amélie Poulain. She grew up in Paris, looked after by over-protective parents (these guys took "over-protective" to a new extreme). During that time, she lived in her own little fantasy world. When it comes time for her to leave home and find herself a place and a job, she becomes a waitress in a little café in Montremart (that is usually pronounced like "mon-marrrh"). Life is okay, but nothing special. One day she discovers an old tin box containing mementos, hidden in her apartment. Amélie decides to track down the boy who owned the box and reunite him with it, and suddenly she finds out what she is supposed to be doing on this Earth ... bringing happiness and love to others. It isn't so easy, though, when Amélie needs a little of her own treatment!

Sometimes you get a movie that people rave about, and when you watch it, it is not really that special. This is *not* one of those movies. *Amélie* <u>is</u> special. Yes, I know it is in French and I know you may need to watch the subtitles. Don't let this put you off. It is a lovely film, and I'm so glad to have it on DVD.
The director, Jean-Pierre Jeunet is the man responsible for *Delicatessen* and *The City Of Lost Children*.
Please, watch this now before Hollywood tries to remake it, as they'll surely lose the magic (like most such conversions).

Amelie
"Quite simply it's perfect. Film of the year"
Total Film
Amelie
a film by jean-pierre jeunet
Amélie DVD and DVD case (UK version), scanned by Rick Murray

And that concludes my Editor's Notes. I couldn't think of a better way to end.

Image from OvationPro user guide

...for Windows!

Prepare to be amazed...

From David Pilling's website, as an *Ovation Pro* user, I downloaded the 'time restricted" development version of *Ovation Pro* for Windows.

Installation was simplicity itself. The program comes as a big self-installing `EXE` file, which, when run, asks you where you'd like to place the *Ovation Pro* application.
By default, this is:

```
    C:\Program Files\David Pilling\
```

so I kept it at that.

Once it has been installed, you simply go to the *Start* menu, *Programs*, *DavidPilling*, and choose *OvnPro*.

Ovation Pro loads, quickly, and a blank window opens. I'm sure a number of Windows users will be disappointed. There's no giant "What would you like to do?" window, no colourful animated 'splash screen', and no 'wizards' to create anything from birthday cards to tax returns.

This is because *Ovation Pro* is a *serious* DTP package for *serious* people. The program is no less capable, it simply isn't loaded with the fluff that you'd find on other comparable packages – I myself have never had a need to use a page design wizard, as I start my desktop publisher knowing fairly well what I want to do...

The first thing I decided to do was to load my CV into *Ovation Pro* (under Windows95 and RISC OS both). This was pretty easy. The only hiccup was because I have to convert the old Ovation document to OvationPro format. As it turns out, this probably wasn't necessary as OPWin has an Ovation1 loader – I just didn't know the correct file extension!

Looking at the document in the Windows (W95 OSR2) version of *Ovation Pro* (on the 33MHz 80486SX co-processor!), it looks just the same as the RISC OS version. I saved it (under Windows) and reloaded it in the RISC OS version. Life isn't always quite that simple as there is issue of font mapping to see to, but in this case OPW used "alias RO Homerton" and the like, so it all went *smoothly*.

Yes, *smoothly*. Not just the font issue, but in use. "*smoothly*" on the old '486 card is hard to achieve, but somehow David Pilling has managed it!

Next, I loaded a DDL file of a teleplay written with the Scripter part of OvHTML – you can find out more about this at:

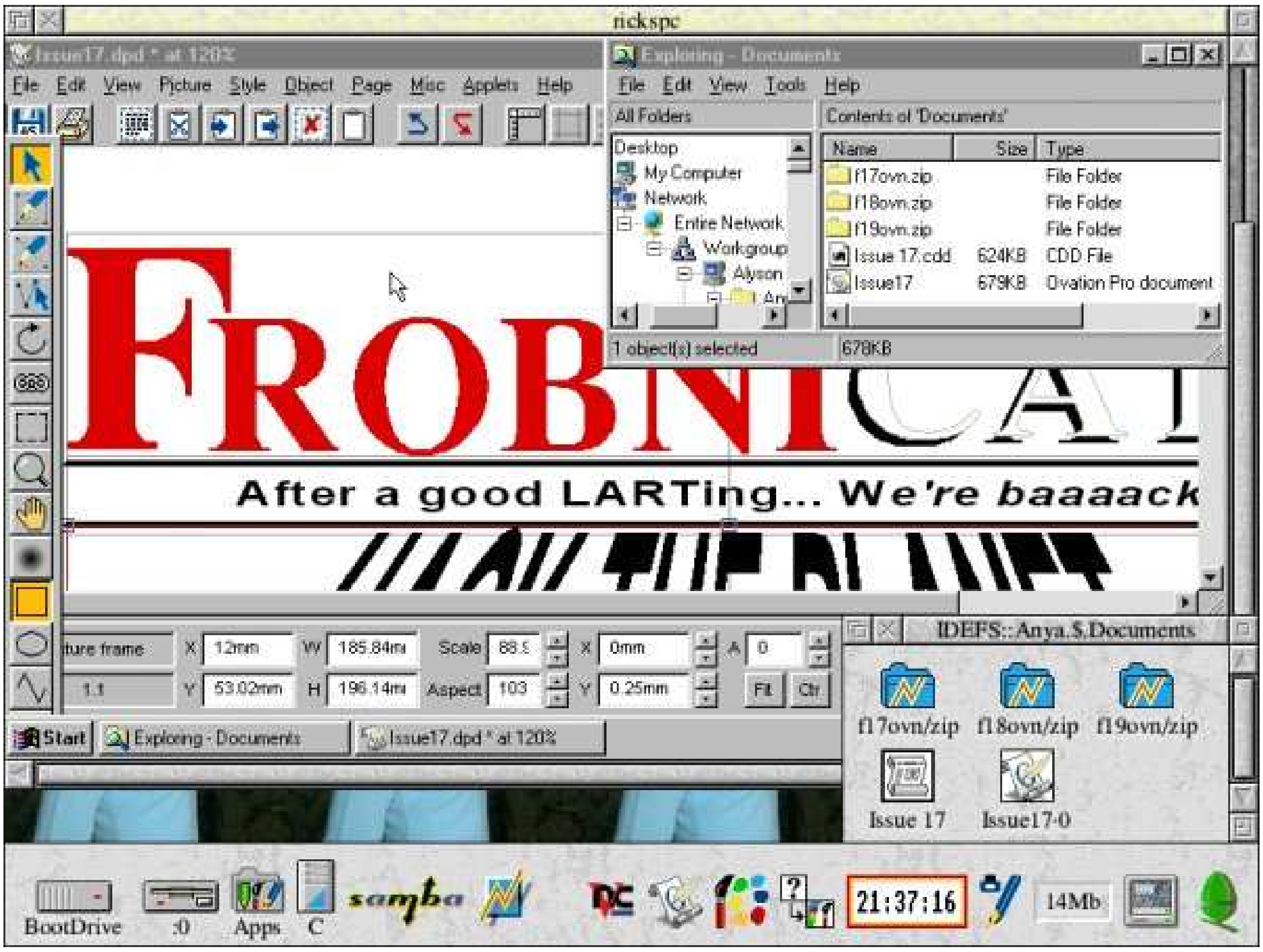
http://www.heyrick.co.uk/software/ovhtml.html

...and then printed the same page to the same printer, an HP DJ540. There were slight formatting differences due to the minuscule differences between Trinity and Times.New.Roman (for example). However I did have to go back and look at the screen to tell which printout was which – they were *so* alike!

People used to *Ovation Pro* will be right at home, everything has a Windowsy look to it, but other then that, things are in the same place. The keypresses appear to be the same. The only real difference is the pop-up menu is along the top of the window – but this is a "Windows" issue.

I must admit that I found a RISC OS-alike menu, complete with ticks and things, and I laughed. Sadly I do not have a picture of this as I forget where it was. Oh well...

The final test that I gave to *Ovation Pro* for Windows was to do something that, honestly, I never thought I would see... Turn the page and *prepare to be shocked...*



This time, *Ovation Pro* was loaded on my 90MHz Pentium (W95 OSR1½). I converted issue 17 to OvationPro format. *Ovation Pro* on Windows loaded this file, via the network, from:

IDEFS::Anya.$.Documents

We're looking at the PC's display (called 'RicksPC') via the VNC system as I only have the one monitor.

And there it was. The previously-impossible:

Frobnicate...

...in *Ovation Pro*...

...in Windows!

Oh... my... God!

It is worth pointing out that using *Ovation Pro* under !PC was about the same speed as using VNC on a 10baseT network. This hardware is not 'cutting edge' by any description, but to give it credit, the software works!

Ovation Pro isn't one of these programs that demands a more recent version of Windows (ME, XP, etc), 128Mb RAM, a few hundred megabytes of disc space, not to mention a 1GHz+ processor. Obviously the more memory and processing power you can throw at Windows, the better, however using *Ovation Pro* on a W95 system, 16Mb RAM, and a 33MHz processor is slow but *not* impossible.

In a future article I intend to try out *Ovation Pro* against a typical DTP program for Windows.
Until then, it goes almost without saying that this software is quite seamlessly integrated with it's brother – the RISC OS and Windows versions share files with minimal fuss.
David Pilling has achieved the impossible. Not only do we see a complete and functional version of *Ovation Pro*, but it 'feels' like its RISC OS progenitor – and anything that feels good under !PC and a 33MHz processor has *got* to be impressive!
As this is development software, it is time-limited. The limit appears to give you a few months of use. Afterwards, you will need to register as a 'proper' user of *Ovation Pro*. Et pourquoi pas?

Responding to the engineer

A friend posted me his Archive volume 15 CD-ROM set as he no longer required them, having since 'upgraded'. It made for interesting reading. Obviously I have *not* read the lot in only a day – just the interesting parts! I am going to email this article to Paul, it'd be interesting to get some feedback from Archive...

I must admit that I fully agree with 'The Engineer''s often excessive rants on dust as I have actually seen a cheap PC blow a hole in it's fragile plastic casing due to an exploding part in the PSU. I'm the kind of person who used to think it was fun to wire a meaty electrolytic capacitor (25V max) up to the mains to watch it blow up and shower tin foil bits across the room (*do not <u>ever</u> try this at home!*) and, well, whatever the PSU cooked due to the build-up of crud, the exploding part put all of my crazy frolics to shame.
So, if you think it is boring to open up Archive and see him talking about dust, *again*, take him seriously. It costs a lot less to pay for a few inches of Archive in which he repeats himself, then it would cost to replace a blown-up part (and surrounding damage).

However, I was *horrified* at his (Ray Maidstone's) suggestion of turning your computer on for a short, organised, session rather than simply leaving it on all the time, in Archive 8.08 [July 1995]:

> *Hard drives wear out in proportion to the time they spin, although, with modern drives being able to spin down, this is no longer a problem.*

and:

> *So, I would advise that if you can organise your computer use to two well-spaced sessions a day rather than just leaving it on, this would reap eventual rewards in longer life of the equipment.*

Let's put it this way. Back in 1994 (or was it 1995?) I had a 1Gb drive for my A5000. It was in the days when it was "impressive" to have a gigabyte. It was a SCSI unit. Originally it ran so hot I couldn't touch it, but after pinching a fan out of an over-cooker extractor (it had three, they wouldn't notice one missing!), I cooled that thing down nicely.

Now, the drive was a fast spindle type. It came with a guarantee that was written in far too many words, but it amounted to 10 years in an always-on situation, or 1 year if the system is turned on at regular intervals (i.e. daily). The drive lasted about two years.
What happened was *thermal stress*. You turn things on, they heat up. You turn them off, they cool down. Over a year, that's some 300 cold/hot/cold cycles (or 600 if you do it twice in a day). Eventually, parts will fail. The solid state is the least likely to fail (but not impossible – the main failure point with the solid state parts is corroded contacts). The most likely parts are your power supplies (the start-up surge my Presario 1410 monitor takes is enough to make the lights flicker!), and your harddisc. I noticed that the technician mentions that modern harddiscs are designed to be spun-down. I believe that doing this is *not* good for them, just because a drive can be spun down doesn't mean that it should be. The assumed exchange rate for harddiscs is high – yesterdays 8Gb behemoth is todays scrap metal.

Back when Ray wrote that (1995), it was probably usual to have a 40Mb drive, better to have a 210Mb, and 'cool' to have 510Mb, and impressive to have anything bigger. In the past few years, thanks no doubt to the amazing inefficiency of a certain other system, harddiscs with 20-80Gb (with a 'G') are not uncommon. In 2001 I went to a local PC World and asked for a price for a 4Gb drive. The guy told me to get real and quoted me a 20Gb unit, the smallest they could get. What the hell would I need a 20Gb drive for? I mean, *really*?
I switch my computer off every night. I know I shouldn't, but I cannot sleep without quiet. During the day, I barely notice the fan and three harddiscs. At night, it's like playing *Linkin Park* full blast. In an ideal world, the computer would not make any noise (I can't contact Ray for details of his fan quietener,

as that won't make my harddisc any quieter). I understand that this is bizarre, especially for somebody who spent five years at a boarding school!

Incidentally, the newer IDE drives cope better with this than did the SCSI drive. I guess modern technology is making the drives more and more reliable... *or is it*? The modern drives have all sorts of smart firmware to mask out bad bits and error correct. By the time your filer reports 'Disc error...", your drive is pretty buggered as *all* of the error correction methods have thus failed. In other words, if your 20Gb monster reports a disc error, abort *everything* and back up your data right here, right now, right here, right now, right here, right now...

What I have done is to fit an additional fan inside my RiscPC (it's a two-slice). I have smoke matches here, from when I was involved with EMC and ISO 900x testings, but I've yet to set one off in my bedroom (can you imagine my mother's reaction if she saw smoke pouring out of my RiscPC?) so I've guessed where the air-flow goes, and added the extra fan to give it an additional push. My harddisc runs warm, but I can keep my hand on it, which is a start, I guess.

This summer, here in sunny France, the temperature topped 41°C. At one time, according to CNN teletext, we were the *third hottest* place on the planet. Okay, not here specifically, it was most of France, into Italy. Hotter was Madrid, and some place near Iraq.
During this time, my harddisc began to fail. It would shut itself down with no warning. Upon a reboot, it may or may not come back. Typically the Simtec start-up report said 'V' for the first time, then 'B' a couple of times, before saying 'MLR' like it normally does. I have no idea what those letters mean, only what should be there, and 'V'/'B' were *not* normal.
Each drive was a master. The drive was warm, but it didn't feel overly hot.

I poked around, to no avail. Eventually I encountered other effects, like corruption in the data written, and the damn thing shutting down in the middle of a CD-R write. Fearing the drive was on the way out, I asked John Williams (he has a place in France not too far from us) if he could pick me up a second hand 1 or 2Gb unit. He brought a 1Gb he found for a fiver. I copied my main software across and took the time to burn the partitions on the old drive to CD-R.
Disaster.
The new drive failed.
John warned me not to put too much trust in a five-pound drive, but hey, this just stopped responding to IDE signals. A side effect was the partition table was screwed up. I've managed to recover that without the hassles of a format and re-install, however please observe:

```
*IDEFS:DeviceMap
Simtec 16 bit IDE interface (slot 2, interface 0)
 0 QUANTUM SIROCCO2550A        2445MB  63  16 ML-
Simtec 16 bit IDE interface (slot 2, interface 1)
 2 FUJITSU M1636TAU            1226MB  63  16 ML-
*
*IDEFS:ListMounts
Name        Size   Dev Part Drv Boot Access Type
===================================================
 [we're not interested in the Quantum partitions :4 to :0]
Anya         408MB  2    0  :1       R/W    RISC OS
Tara         408MB  2    1  :2       R/W    RISC OS
Willow       410MB  2    2  :3       R/W    RISC OS
Raw-2        456KB  2    3           R/W    Other
*
```

You might read that and think I have a thing about *Buffy* (you'd be right). Pay more attention. This one is weirder...

No? Then try adding it up.

`408+408+410 = 1226`, the quoted size of the Fujitsu drive. So, what the hell is that `Raw-2` partition? I'm sorry, but I find it highly suspicious that the numbers run in sequence (456), besides which, the unit isn't a 1682Mb drive!

I've left it, as it is ignored by IDEFS and causes no problems (I don't plan to `*attach` it and look at it with a sector editor either, in case the drive panics), however, you have to admit it is pretty weird.

So I looked at the wiring. The IDE lead was longish and doubled over – just like Ray advises against! – and yet it had worked flawlessly since I fitted the IDE card into my RiscPC, what, two years previous?

I changed the IDE cable.
No difference.

I pointed my speakers at my RiscPC and played *Linkin Park* at it. That didn't work either, and I could hardly threaten it with *Evanescence* as it has already heard it (it likes "*Hello*", which is disturbing...).

As a last ditch attempt, I decided that since it wasn't exactly cool around (it topped 34°C *inside*, and this place has stone walls two feet thick), I dug up that old mains-powered extractor fan and attached it across the top of the machine, in the middle, and left the lid off. I powered up and let it run for a while. The air shifted was so immense that the harddiscs actually got *cooler* with the computer powered up.

And, hey, the whole system worked!

So I fitted the fan inside (overkill? nah, just about enough kill) and shut the lid. The airflow inside the RiscPC was all mucked up, but that wasn't a problem, the air coming out of the side of the case was enough to blow papers around! I only took the fan off two months ago when it started to get cold. I can now hear the harddiscs click, but they're warmer. Maybe I'll clock my PC back down to 75MHz and pinch the processor fan and stick it to blow air over the harddiscs? The PC will be okay, it didn't have a fan until I upclocked it to 90MHz.

The lesson here? I'm sure Ray will agree that a repeated and reproducible failure of a specific part doesn't mean that the part itself is faulty. As the second drive failed, this points to the IDE hardware – however I am inclined to trust the Simtec hardware.

What I'm less inclined to trust is the damned edge-connector on the RiscPC's backplane, who's smart idea was *that*? Either way, something was getting too hot and not liking it.

In another article (I forget which), Ray mentions about PCs going 'boom' as a result of dust. I remember the office PC at work (before I was a Care Assistant) failing because of the processor fan failing. It was dusty, but not jammed. One day, it just popped it's clogs. The processor, a Pentium 233, was so hot you could burn yourself touching it (I know, I did). How did we spot the fault? Well, amazingly it didn't explode or catch fire. It kept working. Only it seriously warmed up the memory above it. This lead to the highly amusing situation of Windows swapping to random locations on the harddisc, and getting stuck in a loop chundering away like crazy. The person who owned the computer was amazingly annoyed, as he wouldn't know a backup if it ran him over. Twice.

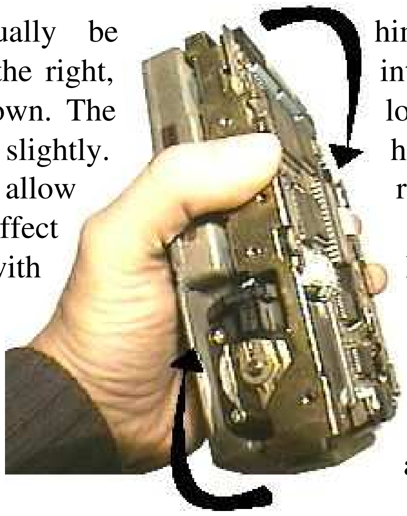
Next, a fault report. *Can you help?* My RiscPC has an i-cubed EtherLan 600. I'm using EtherH 4.33 (does anybody *have* the later version? it's impossible to find!). 10baseT connection to a MaxiHub 5031 (I bought it 2nd hand for £8, but no documentation, does anybody know anything about this hub?).

The problem is this. The *very* first time I switch my computer on and try to do something network related, the machine freezes. Totally. If loading Fresco, it freezes upon loading the HTTP module. But it isn't Fresco, as VNC will stiff the machine as well.
I noticed that the PC was unable to DNS "alyson" – it is set to ask the RiscPC (Alyson) for the address of Alyson, since the RiscPC is supposed to be acting as a DNS server.
Upon a push of the reset button, it all works just fine.

I do not imagine it is a hardware failure, as it is all okay after a reset (and any resets afterwards). But, on the other hand, I can't imagine it is a software failure...for pretty much the same reasons.
I suspect the most likely occurrence is that I've screwed something up in the change over from the big disc to the smaller disc. I cleaned a lot of crud out of the boot-up.

Does this sound familiar? If you've had this problem ... please get in touch!

Brian Cowen mentioned (in his hardware column) about recovering a stuck drive by opening it up in a reasonably clean environment and oiling the mechanism. For most of us, opening a drive is an instant kiss of death to it!



These old stuck drives can usually be resurrected by holding as shown on the right, and jerking your wrist around as shown. The idea is to cause the platters to spin slightly. Doing this four or five times should allow the drive to spin up. This seems to affect the 40Mb Connor drives (as supplied with the A5000), I've got two that tend to 'stick' after a period of disuse.

The drive in the picture is a 20Mb MiniScribe. It has an Apple sticker, so I guess it is made by Apple! It's a SCSI drive, and not at all fast – the on-board crystals are 8MHz and 10MHz. Of particular interest is the stepper motor "interrupter" that can be seen above the lower arrow. This thing should be in a museum!

While I'm on the subject of harddiscs and advice, I would like to remind you that if your Armstrong Walker IDE drive inside your A3000 (is this HCCS?) fails because it doesn't spot any drives, then it is *not* faulty. The IDE software takes a configuration to tell it how long to scan for IDE devices.
Rather silly, but upon a CMOS reset, it defaults to zero seconds (or it isn't smart enough to figure a zero byte is surely invalid?).
Either way, your A3000 will start up with no harddisc recognised!
I don't have my A3000 handy to tell you the commands, so type `*Status` and look for the IDE configuration commands, they should be listed at the end. One of them will control how long to scan for drives.

Sadly, I know a person who *didn't* realise this, so they threw the drive (and card) across the room. The interface card was broken (shame, I could have used it in my second A3000), but the drive survived. It's currently sitting here doing nothing as it's one of those tiny things (not a 'standard' IDE drive) and the stupid Toshiba T-4400 laptop appears to only work with a fixed selection of drives. I *hate* it when asinine restrictions like that are built into the BIOS! Oh well, if the one in my A3000 packs up (unlikely, I don't use it much!), I'll have a spare.

The person with a temper? He went and bought himself some sort of Mac and has gone into the advertising business. We're no longer in touch, but last I heard he was having fun and raking in the cash. It's all right for some, ain't it? `:-)`

I guess that is about it for this article.

All that remains for me to do is to wish a good, happy, and successful year to Ray and Sara at The RepairZone, and also to Paul Beverley at Archive.

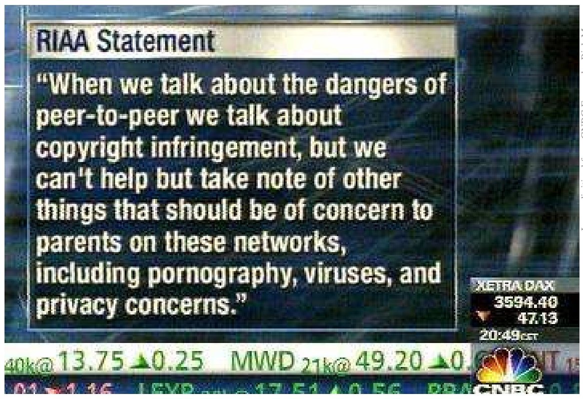
You can find out more about Archive by looking at:
`http://www.archivemag.co.uk/`

I am not aware of a website for The RepairZone, however I have heard it recommended; not just for Acorn machines, but PCs and other machines as well. Apparently Paul also slipped in a tea urn amongst a collection of computers that needed repair!

You can contact The RepairZone at:

Address:	Repair Zone, 421 Sprowston Road, Norwich, NR3 4EH
Telephone:	+44 (0) 1603 400 477
Email:	repairzonelay-teelbigfoot.com

We're *all* criminals, right?


RIAA Statement

"When we talk about the dangers of peer-to-peer we talk about copyright infringement, but we can't help but take note of other things that should be of concern to parents on these networks, including pornography, viruses, and privacy concerns."

Image digitised from CNBC Europe by Rick Murray

Or maybe that should read:
We're all *stupid*, right?

It seems not a day goes by when the movie and music industry is pushing and lobbying for all sorts of rights to combat the "amazing flood of pirate copies" that are causing *billions* of dollars of losses. And in the case of the RIAA, if all else fails, spread FUD – as shown by the screenshot on the right, taken from CNBC taken in early September 2003.

This whole attitude is just as prevalent in the computing sector. I read an article in a popular computing magazine about an early release of XP that didn't like installing on the test machine. After a few calls to Microsoft (apparently each installation requires a call to Microsoft for some sort of 'key", but I don't know for sure as I'm not stupid enough to think of installing that garish colour-mismatch known as *XP*), the people there got a bit shirty. The reviewers didn't reveal themselves, they just ploughed on until they eventually gave up and tried a *different* copy of XP on a different computer. This, it must be said, is possibly a *worse* idea than those dumb key discs and dongles. Okay, in *theory* it is a wonderful way to both support your (legitimate) users and keep a track of your software. But, to behave like that with an early release (which isn't always going to be 100% reliable on the gamut of potential systems!) is really *naff*...
Why? Because it seems the whole system (software sales in general) is set up on the principle of us (you and me) being a thief. Which, frankly, is the *last* thing I care to be reminded over and over again.

I will look at the wrong way, and the right way to 'protect' software.
The wrong way is *Interdictor II*, the flight simulator. I quite liked flying around chasing the enemy planes. It was nice and relaxing. Unfortunately, I do not know where the user guide has gone, so the software is now useless. Okay, once or twice I have correctly chosen the two colours, however each time there are some fifty combinations (five colours, twice; 5×5×2) and usually I get bored before I hit the correct combination. I understand that the game *Flashback* uses a similar system, only with codes that accompany weird drawings. If you lose the codes, the game is useless. Given that I bought *Interd*ictor II many many years ago, and have moved twice, losing the book is not exactly an unthinkable thing.
The correct way is *Ovation Pro*. You are supplied with a registration file that is necessary for the software to start. The software may be copied onto another computer, but the registration file must go with it. When *Ovation Pro* starts, a window opens with your name and address. So if you give a copy to a friend, it'll be obvious where it came from!

Our local "Super U" supermarket – classy, isn't it?
This picture taken 2004/02/03 and it is 16°C and mostly sunny!

In a similar vein, I take my backpack with me when we go shopping. In it you may find floppies, a video camera, a basic medical kit, and all sorts of other "junk". Our closest supermarket, the Super U, is not bothered by me. Many of the girls that work there know us, and say "hello" (okay, they say "*Bonjour*" but that means "hello"!). On the other hand, a local E.Leclerc gets nasty at me thinking about wearing my backpack. I have to take it off and leave it at a desk where, frankly, it would be remarkably easy to swipe. My mother accidentally went in with a book bought in the attached bookshop (obviously just a book) and they were not nice about it. The general idea seems to be to treat everybody as "guilty". The shop itself is quite nice, but I really don't like the attitude, so I've told mom that I would prefer not to go back if there's another choice.
As you can see, I am not a fan of the Leclerc supermarket. I use this example as the entire world knows that I'm certainly not a fan of Microsoft! Impressions *do* count, as do attitudes. I'm not one of those people who steals "*because I can*"; however one might wonder how much that Leclerc store brings upon itself.
The British power grid was knocked *days* after some politician said "it can never happen here"... *they asked for it.*
Then again; maybe, just maybe, my backpack contains three bombs that I'm stupid enough to say are in it... Who knows?

Qu’est-ce que c’est, ça?
a retrospective

In a move that some magazines would be ashamed of, I am going to look back over my old articles and see how they compare against current events.

I’ll skip all the things relating to BBSs and Fidonet. While both are actually still active, the Internet has all but taken over. In the community sense, this might not be a good thing – but you cannot deny the power of Google to identify a song you last heard fifteen years ago and only really know a bit of a line... ten seconds later, there is it. Who, what, when, and the lyrics. Wow.

> *RiscOS. Well. That is, I guess, like the Volvo of the OSs. Solidly built. And whilst it won’t win races, it will get you there and back long after the other cars have been sold off as scrap.*
> [issue 10; autumn 1996]

Well, that was a bit of a no-brainer really. People are *still* going on-line with 4Mb ARM3 machines. On usenet, certain individuals would berate those who still hope for some sort of support for the old systems. Don’t *even* think to ask how much software now has pointless checks that are not necessary. An example that cropped up recently is Orion v0.10 by Ray Middleton. Dated March 1997, this software turned up a Foundation Risc User CD-ROM. I quote from the !ReadMe file:

```
This little program is written entirely in BASIC and
gives a fairly sad performance on older machines.
Until I get around to shifting at least the major
number crunching routine into assembler I have
deliberately restricted Orion to Strong ARM users.
Life would be A LOT easier if Acorns had a floating
point accelerator!
```

All it took was to load the code into !Edit, and chop out the test for the StrongARM. The program now works, albeit more slowly on the ARM710. Again, a no-brainer. Older systems do things more slowly, users of such systems expect that (though unless they use a newer system too, they may not realise it as much).

> *To use a metaphor, unless Intel can squeeze more out of their processors – we shall surely p*** all over the PC market.*

Okay, I guess I *sorta* got that wrong. Amazingly, that old hunk of junk microprocessor keeps getting more and more powerful. I really never would have guessed there would be a 2GHz 80x86! *But* – as Intel have indicated that more ARM processors (which they fabricate) are shifting, there is something going on here. Only, the war isn’t taking place in the traditional desktop market. And, sadly (or maybe “amazingly”?), RISC OS still does a lot of stuff in a more logical and consistent way than Windows (any version to date). Is it that Windows is held back by broken support for old broken software, or is it these guys near Seattle *still* haven’t sussed how much better the GUI is when it does things *properly*? Oh, I know! It’s simple to fix! Just make it look like something from a children’s television program. Or, as some have said, a Fisher Price™ operating system!

Yes, I *still* watch EuroVision. This time, it was on ARD1 so there was a guy talking in German. I believe it was also broadcast on one of the RTLs, but that’d just be a different guy talking in German. :-)
2002 wasn’t bad, a few nice songs. 2003 was more sucky. 2004? Who knows?
In a future article, maybe I’ll look back at other articles! :-)

Diary of a Hacker...

My head hurt. More than you could imagine, it hurt.
Oh my God! What the... something was choking me. Seriously. This wasn't a good state of affairs. Okay, so the college principal probably wanted me dead – I didn't plan to die by being choked, that is just to made-for-TV-movie-ish. Besides, it hurts.
With some diffi culty I open my left eye. A dimly lit white ceiling. There's some sort of white box beside my bed. I don't remember putting the LaserJet there, but I guess I must have. Possibly some stupid experiment to get decent 300dpi printouts from a BBC micro. Never mind, I'll pop the lid, yank out the drum, and hit it over the head of the person strangling me. Then, hop up, and give that bastard a few blows with the toner cartridge. That black gunk is lethal, and ought to incapacitate anybody.
I reach out and wince in pain as my hand hits something unseen. I try again, and I crunch my fi ngers again. Feeling around, it's like a cold metal tube. Yup. It's a cold metal tube. Firmly attached. Now I'm sure as hell that I didn't put that there, and I'm pretty sure my erstwhile attacker didn't either. If I was going to kill me with a lump of metal, I'd use that solid-looking golf club that they keep advertising on TV.
Sod it. The strangler is doing a pitifully bad job of it. I'll just sit up and push 'em over. They won't be expecting that.

I sat up, and it was me who ended up being knocked over. There was no attacker. It was a weird white tube that was sort-of taped to my face. A bed. A polished fboor. In the almost misty distance I could see other beds. Everything else was pitch black.

Just what I need. Really. Can't those stupid aliens abduct somebody else?

I pull all of the tubes and wires off of me. A box behind me began a frenzied meeping noise, so I reached around the back and pulled the IEC lead out.
"Funny", I thought, "The greys use IEC leads? This is as bad as that Independence Day movie!"

Well, now I know what it feels like to dress as a girl. I'm wearing a light blue gown that ends millimetres below my privates. That's it. Oh well, the last time I saw the X-Files, I didn't recall the aliens wearing *anything*, so at least there's that.

I walk down the room. There are people, humans, both genders. All are hooked to machines. I wonder what that's for? I pat myself down and... nothing *feels* wrong. Maybe they were keeping me doped until my investigation?

In a split second she appeared. Knee-length blue dress. Shoulder-length brown hair. Mouth open. Screaming. One hand fbw to her crotch, and her other hand let go of a styrofoam cup. The cup bounced across the fboor sending hot dark liquid in all directions. The hand that was holding the cup? It just stayed there, hovering in mid-air. This woman, she looked late twenties, faded away, like a lost ghost.
I looked down. There she was, sprawled across the fboor.

I stepped over her and reached a door. The upper half was glazed and it said UCI in big letters. Freaky, aliens run by a chemical company? Or maybe the aliens run the chemical company?
Oh, wait. It's supposed to be read from the other side.
Okay, so what's ICU?

Oh, damn! It was a weird feeling. On the one hand, I was more relieved than I could describe to not have been abducted...by aliens. How American would *that* have been? On the other hand... Damn! It'd have been interesting to meet aliens, and couldn't I just go and wind up those poor saps with their faulty religions? Well, maybe there is a God and he/she/it can't be bothered with me? Hehe...

I walk back to the woman, obviously a nurse of some sort. She doesn't look very comfortable, so I grab her in various places and shift her into something that approximates the recovery position. I pick up her cup and sniff it. Coffee. Instant. Sugared. I down the last little bits last in the cup, then walk to the small desk.

The fi ling cabinet was locked, so I nudged it to a forty-fi ve degree angle against a wall. Then it's a simple matter of reaching under and pushing (or pulling, depending on the model) the metal rod that locks the drawers. In a minute, I was looking at my fi le.

It was December 2003. Details were sketchy, but it appeared that during a student uprising in May '99, I was accidentally knocked off of the roof of the college. Three others were knocked off. Two died instantly, one lasted forty hours. I was in a comatic state when the paramedics found me, and I have remained in a coma ever since. At least, until just now.
Somebody had been writing pertinent notes on a blank piece of paper.

June 1999.	Joined to official Cerys Matthews fan club. Thanks for the CD, get well soon buddy.
July 1999.	F. Sopowitz pleads guilty to forty seven counts of gross indecency towards minors, denies six underage rape charges brought forth after his arrest. Imprisoned five years.
June 2000.	Bank investigation finds unexplained anomaly in current account. All accounts frozen pending investigation.
October 2001.	Parents, living abroad, killed in car accident. Suspect (other driver) was asleep at wheel.
January 2002.	Bank changes hands, investigation dropped. Accounts remain frozen due to medical situation.
July 2002.	F. Sopowitz released early for good behaviour and apparent successful rehabilitation.
July 2002.	College destroyed. Arson suspected.
July 2002.	Property suffered a minor fire, computer monitoring raised alarm prior to significant damage. Sixteen canisters of unleaded petrol discovered on roof of adjoining property.
July 2002.	I was wounded in the line of duty. Positive ID on F. Sopowitz committing arson attempt on judicial buildings.
July 2002.	APB - F. Sopowitz.

At the bottom was a yellow Post-It that read "APB outstanding, still. I'm gonna get that bastard".

I came over cold. I was... I was what, exactly? Unwalking dead? For almost fi ve years? My God, I'm in my mid-twenties and it's like yesterday I was having fun raising hell in college. Now? Oh my God! Oh my God!

There were newspaper clippings in my folder. Not only had a Concorde crashed outside of Paris, but they'd all fbwn their last, practically only a month ago. That was sad. An advertisement for Windows2000, WindowsME, and WindowsXP. Good grief! Windows95 turned up two years late, Windows98 added little, and now it's like they're churning them out! Anthrax scares in America. A bunch of clippings about Al Qaeda, whoever the hell they are... I had to take pause for a moment following the twenty-odd cover stories of the World Trade Center in New York, or maybe I should say the lack of it... It was almost unreal, to fly a commercial aircraft into each tower. In response, a war in Afghanistan and a war in Iraq. Thankfully, the president (who it seemed was only elected because Florida cocked up) took his time to fi nd a believable enemy; though it looks like only America is believing in the story.
The year 2000 came and went. Much of Europe became the EuroZone in more ways than one. Nothing blew up. No planes dropped from the skies. The millennium bug was pretty well dealt with.
And that was that.

Nearly fi ve years. Missing. Gone. Not even a memory to remember.

2003/12/25 01h25 CET

I DON'T *BELIEVE* IT!

One of the main problems for me, attending Acorn shows, was transportation. That's why I have only attended the one. It was fun, but London isn't the easiest of places to get to.

What I always said would be nice would be to have one nearby to where I was. And, by a magic stroke of fate, somebody organised a RISC OS show *practically next door* the year *after* I left England. Can you believe it!?

See the A323 Aldershot Road? Go up it, like, ten miles. Or, failing that, get on the train at Guildford and pass through that nowhere station I can't remember the name of and arrive at Ash. Get off the train at Ash and throw a stone. You'll probably hit the place where I used to live.

No, seriously. I *don't* believe it!

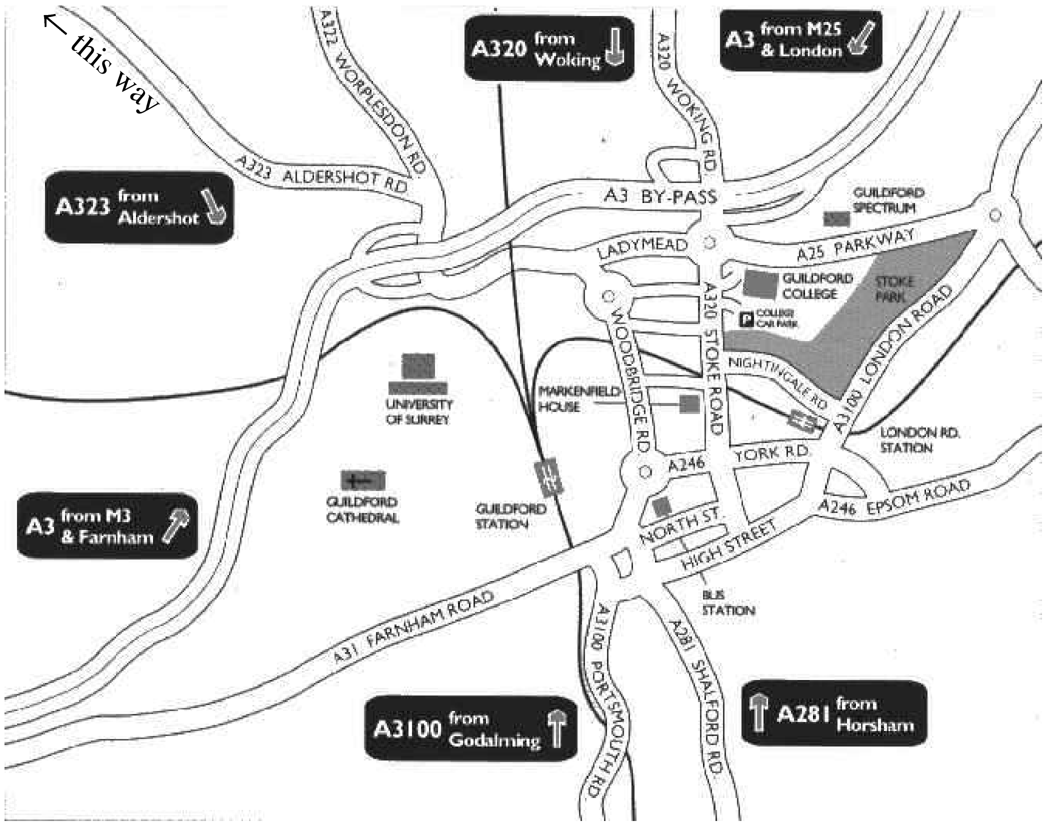


Image from FRU 11

Say, I don't suppose you'd consider holding an Acorn Show in rural France? The scenery is nice. The food is nice (especially if you're big on seafood). I'm sure people like Jérôme Mathevet will be happy to reassure you that while the plugs may look strange, they do actually work. Yes, we get 230V AC here. None of that old 90V DC weirdness. I'm sure I can find an exhibition hall for you in Châteaubriant. Heck, you can probably fly directly into Rennes St. Jaques. Or, failing that, EuroStar or EasyJet to Paris then TGV to Rennes, then chufty-puff to Châteaubriant. Car access is fine. Ferry to St. Malo, then just head south; or EuroStar and head west...

As for names... Well, all the French girls are called *Nathalie*, except the ones that are called *Sandrine*. Simple!

How about it, guys?

The Iyonix - A Year On

Those who receive the *Foundation RISC User*, or were at the Birmingham ARM Club show (in 2002) will know that the new Iyonix was released then and there. You may even be aware that Ron Briscoe was the *second* purchaser of the system. So I contacted Ron and asked him how the last year has been.
Jealous? Of course I am! Tant pis.
On a more personal note - ignore Ron when he refers to himself as an 'idiot tester". From my experience, he is a heck of a lot more clued than your average Windows user (I guess us RISC OS users are a smarter bunch!). For example, he has quoted his screen resolution. You'd be surprised how many Windows users I've met that don't know what that *is*, never mind how to fi nd it. Ron, you undersell yourself.
Anyway, over to Ron for his thoughts...

The Iyonix 12 months on

Last november [November 2002, I asked Ron to write this article in November 2003! – Ed.] I was fortunate to purchase an Iyonix at the Birmingham ARM Club show and have spent the last 12 months enjoying myself with it. When I purchased the Iyonix I was toying with the idea of networking my RPC so as to be able to run software that I couldn' t run natively on the Iyonix. No worries, what with the software developers working overtime 32-bitting and the excellent Aemulor my RPC is almost redundant and to think I had just upgraded it to a two slicer as well.

Impressions? Naturally its quick, when I do go back to my RPC it seems very slow in comparison. Another advantage to me is that I now sit looking at a 1360 x 1024 resolution with 16million colours on my 17 Inch monitor, bliss absolute bliss. Quiet? No silent is more like it, only the CDRW makes noticeable sound.

Software? Right from the start important applications were ready. All of David Pillings stuff, free as well.

TechWriter, TextEase, DialUp, Messenger Pro and other RComp software, CDBurn and not forgetting the ARM Clubs software. If you want to see more visit *http://www.iyonix.com/software/* see the ever growing list of software able to run on the Iyonix.

Of course you would expect commercial/shareware developers to upgrade their software but the most pleasing thing to me has been the response from the PD/Freeware developers who on request 32-bitted their applications, often without having an Iyonix to test on and when dealing with me, an idiot tester who had one.

My special thanks to all of the people for putting up with my demands and to the other users who got their 32-bit requests in before I could.

Now? I sit and wait for Simon Wilson to finish off his TV card drivers, for the Spellings/Lee new bit of wizardry, hope that someone will do some "Sound editing" software/hardware and await Castles next move. Yes the future looks interesting.

Ron

Playing with I²C

Part 1

The I²C protocol is a simple, yet highly effective, protocol designed by Philips Semiconductors. It was originally designed to be a low-cost low-speed way to hook up equipment in televisions and video equipment. Take for example *teletext*. The teletext receiver itself is pretty smart, and it is capable of generating it's own picture. What is needed, however, is some kind of link between the remote control system and the teletext system. In conventional terms, we could combine the systems (actually, this is what many newer TVs do – which is why you can see captions for the TV functions and channel identities when you don't have teletext ... the teletext system is there, is just isn't *enabled* in your set).
For the older sets, where teletext was an *option* (as in an entirely separate bit of circuitry), having all these data lines around wasn't a good idea – especially given the hostile environment you get from cohabiting the space around a device that normally operates at ~32kV!
Enter Philips who figured that it is quite simple, really, to put the commands onto a three-wire bus. On wire is ground. The next is `SCL` which means *S*erial *Cl*ock. Finally, `SDA` for *S*erial *Da*ta.

The way this works is one device can talk to another device by *addressing* it. Any device designated as a *slave* can listen to "talk to me" messages from a *master*. The *master* does this by sending the number of the device it wishes to talk to as the first part of the transmission.
Unless you live in California, where "master" and "slave" is politically incorrect and not to be used...

Each device has a different number hardwired into it, for example the EuroCCT (teletext receiver) is device 34. Some devices have a user-configurable part. This is important in systems where more than one of the same device may exist. The EuroCCT does not, as you're only expected to have the one. The RTC/NVRAM has a configurable part so it can be device 80 or device 82. Obviously, the same device cannot be both at once...

At all times, the *master* controls the 'conversation' and provides the clocking. The *slave* has a limited ability to stretch the master's clock to say "hold on", but this is not something you'll come across often, so it will be skipped over.

Bit zero of the device address specifies a data transfer direction. It is READ/notWRITE, so addressing the EuroCCT as device 34 will permit us to write to it's registers. Addressing it as device 35 (bit zero set) will allow us to read from it.

From this point on, what happens next depends upon the device. Typically, the first transaction performed is a "write". After selecting the device, you write a register number. You then switch to "read" and you can read from the register defined previously. Again, this depends upon the device itself. Many devices support a register auto-increment, so:

```
start
write   device
write   reg #1
stop
start
write device + read bit
read
read
read
stop
```

will read registers 1, 2, and 3. This saves having to set up the register to read from prior to each read.

You will have noticed *start* and *stop* conditions. When `SCL` is high, if `SDA` moves from **high** to **low**, then a start condition is considered to have happened. Every device will then look for it's address in the next byte transferred.
When `SCL` is high, if `SDA` goes **low** to **high**, then a stop condition is considered to have occurred. This may be used to finish a valid transfer, or to abort an invalid transfer.
At *all* other times, `SDA` must change state while `SCL` is low.
Note that both `SCL` and `SDA` float high.

The next thing to consider is how a byte is actually written to the I^2C bus. This is quite simple. We:

- set `SCL` low
- set `SDA` high or low, as necessary
- set `SCL` high
  the bit is now clocked in, we hold this for 4.7µS
- set `SCL` low
- set `SDA` high
- set `SCL` high and wait 4.7µS
- read state of `SDA`

The fake ninth bit is an *acknowledge* bit. The receiver should have pulled `SDA` low. If not, the transfer failed.

When a *master* sends data to a slave, it performs what is, essentially, a reverse of the above. The exception is that the final bit does *not* have an acknowledge bit. This signals to the *slave* that no more data follows.

We RISC OS users are lucky because all of the Acorn machines use the PCF8583P for the "CMOS memory" and clock. For this reason, RISC OS itself incorporates I^2C support. It does this by bit-bashing two lines from the IOC (or IOMD in the RiscPC). Acorn kindly piped I^2C to the podule bus, so it is not difficult to add your own I^2C devices, and call `IIC_Control` to use them. I, myself, have a Ground Control teletext receiver connected in exactly this way – the `SDA`, `SCL`, and `GND` lines are soldered to the backplane connector of my Vision video digitiser.

I like to check the headlines and the weather on CNN, and also to see if anything interesting is coming up on MTV (well, MTV is all in German so that should be "anything interesting that I can figure out what it is saying..."). Given my previous scribblings on thermal stress and the like, the question becomes "do I turn my computer on for fifteen minutes to look at teletext?". The answer, obviously, is "no".
So what are my options. I'm not short of computers, so it'll come down to:

a. Use one of the A3000's and route the video to the TV via the VCR. As I^2C is already built in, I simply run my !Teletext software...

Since I'm not keen on swapping cables, we also have:

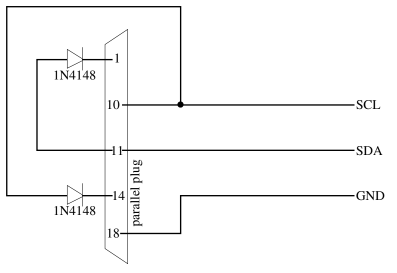
b. Use the laptop. It's a 25MHz PC with ¾ of the screen working. There's no I^2C and it's a monumental pain to program as things you take for granted (like 'int' being unsigned, and 32bit) doesn't happen. DOS is ugly. Windows3.11 is not much better, and I don't *ever* plan to write a Windows program using MFCs.

The answer is pretty obvious. Plug in the A3000.

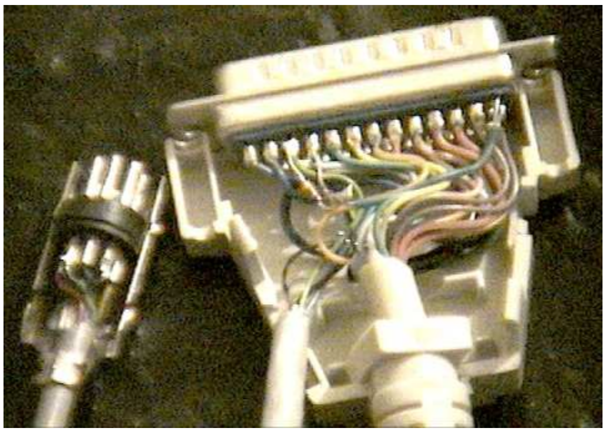
So that's why I implemented I^2C for my PC. It was an interesting challenge.

The question is now a matter of I/O. There are three ports with I/O we can utilise. The first, easy to discount, is the internal expansion connector. These vary from laptop to laptop, 150 pins of direct access. Thanks, but no thanks.
Next is the serial port. This is useful, but most serious ports work at "weird" voltages which is fine for serial-serial stuff, but not so friendly to regular logic chips!
This, really, leaves one option. The parallel port. In our implementation, we don't need a bi-directional port, only one that is "fully wired" in that it can detect common printer signals such as "paper out" and the like. This, sadly, rules out the BBC Micro's parallel port which only really appears to support the data lines, ACK and Strobe (though, you could argue that the BBC is better as you have all those other ports!). Anyway, most computers built using standard hardware will offer a proper parallel port.

Unlike an I/O port, the control lines of a parallel port are uni-directional. You can't stuff a value into Strobe, and then switch to read mode to read what the state of that wire is. This isn't a problem. We simply strap our I^2C data line across two lines of the parallel port. One is an output (for sending) and one an input (for receiving).



The diodes are there to prevent nasty things happening if you're driving the line low while something else wants to drive it high.



Above is a picture of a cable that I built. I have not bothered to remove the parallel part of the cable as it looks like a boring and fiddly job. `:-)`
However *be aware* that it *will* affect the I²C transfers. And, additionally, it will act like an antenna so if anybody in close proximity likes to listen to Radio Four on Long Wave ... you'll hear about it from them!

I²C line	Output	Input
SDA	Strobe	Busy
SCL	Autofeed	Ack

This leads us nicely to interfacing with the parallel port on a PC. Like the Acorn range, the parallel port is mapped into memory.
Unlike the Acorn range, there are no SWI calls to make things easy. There may be a BIOS interrupt, but anybody who has ever tried to use the BIOS to work with the serial port will know that it's hopelessly inefficient (my 80486 flaked out at 4800bps – an ARM2 A3000 can do better while running in the desktop!), so we'll ignore any BIOS options and resort to poking in memory, this is provided for in many BASICs (`PEEK`/`POKE`), C (`inp`/`outp`), Pascal (`byte` and `Ptr`), and likewise in other languages – there is no need to resort to in-line assembler.

The LPT1 **status** register is at **`&379`** and you'll find the **control** register at **`&37A`**. This is consistant.

Something that isn't made clear in the StrongHelp OS manual (v3.08) is that some of the control lines are inverted, thus writing '1' will take the line to logic low.

Port	Bit	Name	Meaning
Status (inputs)	0	TMOUT	Used in EPP mode
	1		*unused*
	2		*unused*
	3	*n*ERR	Low if error has occurred
	4	SLCT	High if printer is On Line
	5	PE	High if paper has run out
	6	nACK	Pulses low to Ack byte
	7	nBUSY	Low if printer is busy
Control (outputs)	0	STROBE	Strobe a byte
	1	AUTOFD	High if auto-LF desired
	2	*n*INIT	(Re)Initialises printer
	3	SLC	High selects the printer
	4	IRQE	Makes nACK cause IRQ
	5	PCD	Select data direction
	6		*unused*
	7		*unused*

Bits 0, 1, and 3 of the control register are *inverted*, the line prefixed 'n' is apparently *not* inverted. Go figure!
According to the datasheet for the FDC37C669:

> **Bit 1 AUTOFD - AUTOFEED**
> This bit is inverted and output onto the nAUTOFD output. A logic 1 causes the printer to generate a line feed after each line is printed. A logic 0 means no autofeed.
> **Bit 2 nINIT - nITITIATE OUTPUT**
> This bit is output onto the nINIT output without inversion.

One would assume that since they are talking about the registers, a logic 1 (or 0) relates to the value written to this bit and *not* the logic value of the data line itself – though this could possibly have been worded to be less ambiguous.

To communicate with the parallel port, in C, we use the functions `inp();` and `outp();`...

Well. That's it. We now know a little bit about how the I²C protocol works. We know how to wire up a lead to connect the parallel port. We know how to bit-bash the parallel port. All that remains is to actually implement the I²C protocol in software. This is *not* as difficult as it might seem.
I'll leave that as an exercise for the reader; and I'll present my implementation of a basic I²C slave, written in Turbo C, in the next issue.

Have fun!

The Alpha Sprite

There appears to be a little bit of controversy regarding the "new" alpha sprite format in that while the basics of the sprite format are known and understood, apparently the new alpha version is not going to be detailed outside of the RISC OS Select scheme.
This, I feel, is little short of madness. While most software will be unable to correctly display sprites with alpha channel masks on older versions of RISC OS, releasing the format allows people to at least code in some basic recognition for the new format and either do the *it works in theory* approach, or the *I know what this is but I don't support it* approach. A hybrid could be to convert the alpha mask to the older on/off mask with a threshold. Whatever. It doesn't matter.
Some have said that the details of the sprite format has never been freely released. That is not entirely true. I learnt of the sprite format in 1989 *without* buying the PRMs or looking for free resources. How? I wrote to Acorn. They replied with a photocopy of the pages from the PRM (the only pages I've ever seen from the RISC OS 2 PRM!). In those days, companies told you things... :-) These days, it is not difficult to document the APIs on a website. Indeed, if RISC OS Ltd has brains, it will document *all* API changes on their website and make this information freely available. Thus, developers who do not have Select can at least use the information to ensure their program works correctly on Select, and maybe some of those changes will be "wow, that's cool" – like, does SpriteExtend do progressive JPEGs yet?
Am I asking Select to give away something for nothing? No, not really. More software compatible with their version of RISC OS is surely a good thing for them? Despite what some may have you believe, for 95% of things, there is little difference between the RISC OS versions insofar as the API is concerned. Refer to my previous comment about the daft StrongARM check in a program. Sure, older systems will do things with less colours, less free memory, and a heck of a lot slower – that's one of the reasons why people upgrade! For those of us using older systems...
...let's face it. I don't actually give a damn whether my software works on Select or not. I write most of my software *to scratch a personal itch.* I tart it up, write a user guide, and release it. If it helps somebody else, then that is good. I will take efforts to ensure my software is 32bit clean because I think it is important to support the new ARM-based machines as it is undeniably the future of RISC OS. If RISC OS Ltd was to send me an HTML (or PDF, or whatever) describing all of the API changes, then I would look over my code to ensure that nothing will go wrong. Why? Because I don't have anything personally against Select, I just don't intend to pay for something I don't use. I don't charge for my software, and I consider supporting *their* OS to be doing them a favour.
Maybe they don't see it this way. Who knows?

Anyway, I have been emailed details of the alpha sprite format, so I will describe it for anybody interested. Please be aware that this is *untested.*

The alpha-channel sprite provides a variable level of transparency (rather like in Photodesk) by using 8 bits of mask data. This gives 256 levels of transparency. As we all know, sprites could be given a "type" as of RISC OS 3.5. This was initially because the old mode number system failed due to their being many more modes than numbers (thanks to shadow modes, the usable range was only 0-127, there are more modes that that in most MDFs (say 40 defs × 5 depths...)). This could be easily expanded to cater for things like CYMK sprites.
From Select release 3, the sprite type word has been expanded by making bit 31 reflect the presence or absence of alpha channel data. Essentially, if set the data is present, else it isn't. If the sprite is defined as not having a mask, this bit is ignored. The mask data is now 8 BPP. Note that it is possibly inverse – 0 means the pixel is completely transparent, 255 means it is opaque. The mask read/write calls now support reading and writing of eight bit values for alpha-channel sprites.
The OS_SpriteOp functions work as expected. Applications that poke the mask data directly, or which display sprites themselves may not work correctly, so it might be a good idea to incorporate a check to see if the sprite you're attempting to display is an alpha sprite or not...

(now, *that* didn't hurt – did it?)

Catching a squirrel...

Catching a squirrel is, actually, pretty simple. There are pitfalls, but largely it is easier than it looks.

Recently, I have been blessed to have access to a "modern" PC. Unlike my wimpy 16Mb P75 machine, this thing has a 2600MHz processor and, well, *plenty* of memory.
Which makes it a perfect candidate for *Red Squirrel*.

Installing *Red Squirrel* was no trouble at all. Simply download it from the website (sorry, I can't remember the URL – but Google found it for me in less than a quarter of a second!). The application itself is surprisingly small.

The first thing we need to do is to extract our RISC OS 3.70 ROMs. These reside at `&03800000` and total 4Mb, thus:

```
*save ro37_r1 3800000 +100000
*save ro37_r2 3900000 +100000
*save ro37_r3 3A00000 +100000
*save ro37_r4 3B00000 +100000
```

will do the trick.
I don't know what *Red Squirrel*'s ROMs are supposed to be called, however it loaded the ROMs when given those names. The ROMs need to go in a specific directory for the version of RISC OS that it is. I am, sadly, writing this without the PC nearby, so from memory I think it is `\ROMsets\370` within the *Red Squirrel* directory.

From here, you can run the emulator. It will initially pop up a window telling you all the good things new to this version of the software. So tell it not to show you that again, and continue.
You'll be asked to select a model. Here we'll go for the RiscPC700 emulation. There is *no* speed benefit to running the StrongARM type, so we'll stick to what we already have (of course, if your RiscPC *is* a StrongARM, choose that option if you prefer).
RISC OS will then boot.
The default mouse settings are as follows:

Left button	Select
Right button	Adjust

The 'menu' key on the keyboard is used for the RISC OS menu button. *However*, it is a Windows convention that the right button is used to call up the pop-up menus. Therefore you might prefer to use the right button as Menu if you have a two-button mouse. Certainly, this makes RISC OS much more 'fluid' as it is a more mouse-oriented GUI than Windows (which can be pretty much entirely controlled from the keyboard). I feel reaching for a key to open a menu is... weird.
Anyway...

Close down *Red Squirrel* by clicking on the 'X' close window icon. Every other method, including a reset, will crash. At least, it did for myself and a friend...

The next step is to get the applications and boot stuff to the PC. So I formatted a 60Mb harddisc and installed a minimal boot/apps setup on it. I then dumped it, sector-by-sector to a file (using `ADFS_DiscOp`, we RISC OS users may forget how *lucky* we are that the OS gives us easy access to things like that). The disc image was then copied to a CD-R as it's the easiest and sanest way to get a 60Mb file between two machines.
Let's cut a long story short and say that it *so* didn't work. In fact, the emulator completely failed to recognise the default 50Mb harddisc image that's supplied on the website (look in the downloads directory, you'll see it). So it appears to me that the option to use a harddisc image is not simply a little bit 'broken', it is completely non-functional.

My original plan? To pull all of the stuff from the harddisc image and copy it to HostFS. That way, we can be sure that we won't have madness resulting from screwed up extensions if I try to do the file copying manually (yeah, there are 3558 files in *!Boot* alone† – so I can *so* see myself doing it by hand!).

So, an alternative approach was needed...

The beauty of RISC OS is that there is *more than one way to skin a cat* (I never really understood that, it seems remarkably cat-unfriendly, but...) and where RISC OS is concerned, there are multitudes of possibilities.

The one I chose was *!SparkFS*. In order to make things quick, I told SparkFS to make a basic Spark

† that isn't including anything within *!Scrap*, by the way...

archive with no compression. Essentially the Spark archive is just a 'container'. I then constructed multiple archives – one for *!Boot*, one for *Apps*, and so on. These were then committed to CD-R. ***Do not use broken `/zip` extensions*** as stupid-flamin'-'doze will try to interpret your attempts to copy "`arc/zip`" as a copy of "`arc`" with the parameter "`/zip`" – useful...not! Ensure they're all proper "`.zip`" extensions. Luckily the host PC has a CD writer too, so I simply imported the previous session and twiddled the filenames that failed.

The final piece of the puzzle was how to get *!SparkFS* itself onto the basic RISC OS 3.70 system without any *!Boot* and without any faffing with disc images.
The answer? First, make a copy of *!SparkFS* within a sub-directory on your RAMdisc. Load the *!Run* file and patch it to load a copy of FPE and CLib, which you've just copied into *!SparkFS*. Get rid of that check for `System$Dir`, and ensure it doesn't try to load anything externally. Copy the stuff you need into *!SparkFS*.
If you have Paul John Murphy's *!SparkInfo*, drop that into the sub-directory too, just in case...
Then... here comes the magic... drop the *entire* lot into *!Dicottery* and let it do its thing.
The resultant executable can then be dumped onto a PC floppy disc.

Right. The next step. On the host PC, copy the Dicottery file into `\HostFS\370\` or whatever. Then copy *all* of the Spark archives from CD-ROM into the same place.

When that is complete, fire up *Red Squirrel*. It will boot (slowly), and then dump you into the command line after complaining that *!Boot* can't be loaded. Don't worry, that's to be expected.
Type:

```
    *desktop
```

and when the desktop arrives, the harddisc icon that you can see (oddly, to the *right* of the floppy disc) is the HostFS system. Click on it and you'll see the archives and the Dicottery executable. SetType the Dicottery thing to `&FF8` (Absolute) and then run it. You'll be offered a window with a draggy-directory. That's why I told you to put *!SparkFS* into a sub-directory, see?
Drag it out, and in moments *!SparkFS* will be unpacked. Run *!SparkFS* and begin unpacking the Spark archives.
When it is all done, call up a TaskWindow and check that all the CMOS RAM settings are correct. The more important ones are:

`Ignore` is yours normally 'no ignore'?
`Territory` and `TimeZone` if you are not in the United Kingdom.

While you're there, load up *!Alarm* and ensure the year is correct. Leave other stuff 'til later (do it with *!Boot*).
Now, shut down the desktop (Ctrl-Shift-F12 *twice*) and click the 'X' close window icon again.

Reload *Red Squirrel* and with any luck it should get part-way through the boot sequence before it aborts with some obscure error like not being able to find `FileSwitch$CurrentFilingSystem$CSD` which normally means something is trying to directly address a filing system that you might have had on your real RiscPC, but don't have on the emulation ... like IDEFS. You'll need to load up *!Boot.Utils.BootRun* and the start-up files, mine are *!Boot.Choices.Boot.PreDesk* and *!Boot.Choices.Boot.Desktop* and the directories within *!Boot.Choices.Boot* and tidy up all the stuff that is incorrect or undesired. An example is *PCSleep* which, well, has little relevance when the RISC OS side is what we're emulating!

Pretty soon, you'll get it all sorted and the machine will boot cleanly (don't forget – if your *Red Squirrel* can't do a reset (that includes `SYS 106`!) then shut it down and reload it, else you'll risk wasting large chunks of memory to a stalled program). Double-click on *!Boot* and set up the other options...

The *Red Squirrel* website lists the games that work, so I'll tell you the applications I've used, so far, without problem:

ChangeFSI, *ProFiler* by Thomas Ollson (was once *Filer+*), *FYEO2*, *ArtWorks*, *Ovation Pro* and of course my *OvHTML*, *InterGIF*, *Browse*, *Fresco*, and – of course – *Zap*.

and that's just my first day.
Because *Ovation Pro*, *Browse*, *Fresco*, and *ArtWorks* are "commercial", the shutdown deletes them. But it's a two-minute job to pop in the CD-ROM and re-decompress them as needed.

Verdict? The emulation is a little unstable at times (including the disappearing mouse pointer, though pressing `F12` then `Return` fixes it). I do wonder if these problems are ever likely to be fixed since the 'better' version is a commercial product. But, we should remember there's no such thing as a free lunch. *Red Squirrel* performed only about ¼ slower than my real ARM710 machine so it is *perfectly* usable. And it is free.
Overall: Wow! I'm impressed. It is useful. **`9/10`**

Software protection...

Something I used to like to do when I was younger (and, still, now – only to much a lesser degree), is to break software “protection”. It was often an interesting challenge. Indeed, for the various programs-of-a-certain-nature that I de-protected, the actual purpose of the program was a lot less interesting than the hack. `:-)`
This wasn’t done for purposes that you may consider immoral. Sometimes a game or application was written with the *assumption* that it’d be run from `adfs:0.$` which might have been the case in the dark days way-back-when when a 10Mb “Winnie” hard drive would set you back over £750 and offer a performance worse than a single-speed CD-ROM drive!
As time came on, everybody got harddiscs, but you’d still meet games that did evil things like screwing with the CMOS RAM, demanding to be run off of fbppy, etc.

In this article, we will examine how to hack two forms of protection – key fi le based and date based. These are both taken from real-life examples, but so as not to implicate myself (!), I have generated example code from scratch. Feel free to borrow these techniques for your own code, but *beware* that this article explains how to compromise them! `:-)`

Key file based. This method works by loading in data from a “key” fi le. The data supplied is the registration name, along with one or more checks to ensure the validity of the key fi le.
First we’ll look at it from the user’s point of view. With the software running normally, it says:

```
Program loaded, user "Frobnicate"...
```

and if the keyfi le is incorrect/not present, it’d say:

```
Invalid registration, unable to continue...
```

Before we look at source, we’ll look at the code in code form. The keyfi le system has a cute little EOR encoding system, and two separate checks on the data. Despite all of this, it has one weak point... This may be easily discovered by working back from the message...

```
0000809C : ™..» : BLLT    &000086D8    ; call: __rt_stkovf_split_small
000080A0 : &..ë : BL      &00008140    ; call: load_keyfile
```

```
000080A4 : ÿ.Pã : CMP     R0,#&FF      ; ="ÿ" (255)
000080A8 : .... : BEQ     &000080C0
000080AC :  .flå : LDR     R0,&000080D4
000080B0 : ..•â : ADR     R1,&000080D8 ; -> string: "Invalid regis..."
000080B4 : fi..ë : BL      &00008734    ; call: _fprintf
000080B8 : .. ã : MOV     R0,#1
000080BC : É..ë : BL      &000087E8    ; call: exit
000080C0 : @.flå : LDR     R1,&00008108
000080C4 : ..•â : ADR     R0,&0000810C ; -> string: "Program ..."
000080C8 : –..ë : BL      &00008730    ; call: _printf
000080CC : .. ã : MOV     R0,#0
000080D0 : .¨.é : LDMDB   R11,{R11,R13,PC}
000080D4 : "‘.. : MULEQ   R0,R4,R0
000080D8 : Inva : CMNVS   R6,R9,ASR #28
000080DC : lid  : RSBCS   R6,R4,R12,ROR #18
000080E0 : regi : STMVSDB R7!,{R1,R4-R6,R8,R10,R13,R14}^ ; *** ! and ^
000080E4 : stra : CMNVS   R2,R3,ROR R4
000080E8 : tion : MCRVS   CP9,3,R6,C15,C4,3
```

Well, there it is. We `BL` to *load_keyfile* and afterwards we compare the result with 255. If it is equal, we skip to the “it’s okay” message, else we skip to the “it’s not okay” message. To force it to always report things are fi ne, and to not even bother looking for a keyfi le, simply alter the following:

```
000080A0 : .. á : NOP
000080A4 : .. á : NOP
000080A8 : .. á : B        &000080C0
```

If your assembler tools don’t support `NOP` (I use *!Zap* with Darren Salt’s *ExtBASICasm* and *Debugger Plus* modules), then use `MOV R0, R0` as it’s the same thing.
Fancy putting your own name in (since there’s no keyfi le to read)? This is not hard, since the username is stored in plain form once it has been loaded. It is shown to you in the *printf* which makes things really easy.

```
000080C0 : @.flå : LDR     R0,&00008108
```

Here is the statement that sets up the parameter load to the *printf*. At `&8108` is the value `&8CE4`. This is the pointer to a blank section of memory where the username is written. Hard-code the name you wish to use in that place (I used “Rick”), then run the program again...
...it now says:

```
Program loaded, user "Rick"...
```

Congratulations, that’s the fi rst hack performed.

The second hack is one you'll come across from time to time. The software works, but only for a while. In our case, *demo2* will work up until the end of 2003. It won't work now, unless you fiddle with your system clock – a very unsatisfactory idea.
To make things harder, this program is compiled *without* embedded function names (as would be common in commercial software). Additionally, it splits the 'failed' message from the check code by "scary" use of the *longjmp* facility. In essence, it takes a pretty sick mind to come up with this code!
But, as you'll see, it *still* has a point of weakness.

There are two ways that we can approach this. The first is to find the message "*This program has expired. Please purchase the full version if you wish to continue using this software...*" and work backward, while the second is to look for the time/date routine.
With *!Zap*, this isn't too hard. Load up the code and look for something to do with time. There are three ways that this may be achieved. The first, and most likely, is a call to the standard C functions, such as *time()* and *localtime()*. The second, a call to the *OS_Word* SWI to read the CMOS clock – this may be either `OS_Word 14, 1` (BCD, deprecated) or `OS_Word 14, 3` (five-byte value). The final way, the mark of a lame programmer, is to read out the values from the environment, namely `<Sys$Time>`, `<Sys$Date>`, and `<Sys$Year>`...

In our program, *demo2*, we encounter it fairly quickly with the assistance of *!Zap*:

```
000080F0 : .. á : MOV     R0,R13
000080F4 :  ..ë : BL      &00008710         ; call: time
000080F8 : .. á : MOV     R0,R13
000080FC :  ..ë : BL      &00008720         ; call: localtime
00008100 : ..'å : LDR     R0,[R0,#&014]     ; =20
00008104 : Û.. â : ADD     R0,R0,#&036C     ; =876
00008108 : .. .â : ADD     R0,R0,#&0400     ; =1024
0000810C : }ÎPâ : SUBS    R12,R0,#&07D0     ; =2000
00008110 : ..\£ : CMPGE   R12,#3
00008114 : `..Å : LDRGT   R0,&000080BC
00008118 : {. Ã : MOVGT   R1,#&7B           ; ="{" (123)
0000811C : Þ..Ë : BLGT    &0000889C         ; call: longjmp
00008120 : .¨.é : LDMDB   R11,{R11,R13,PC}
```

Here we clearly see the calls to the C time functions to set up the time block (`timeblock->tm_year` is offset +20, hence the &014 offset in the `LDR`). The `SUBS` and `CMPGE` are the compiler being exceptionally clever (it's a test against '2003', or `&7D3`). Following this we have a set of three `GT` condition opcodes – the compiler making the best use of the processor's pipeline.
So, what's a girl `^W`hacker to do? The answer is simple. Those three `GT` conditionals? `NOP` 'em all.

And now, despite the strange use of *longjmp* and putting the message in a separate assembler chunk to disassociate the message and it's test... we *still* encountered *one* place where it came down to a "*this-or-that?*". One place where a few `NOP`s could render the protection useless.

Now, it should go without saying... but sadly I know from past experience that I must say it... **I *will not* engage in any discussion about how to "crack" any program**, regardless of it's vintage or distribution status. *This includes "shareware" and "freeware"* . I write this article to give *you* some food for thought.

If you need some help with the ARM assembler part, then I refer you to my assembler programming tutorials, at:

`http://www.heyrick.co.uk/assembler/`

or otherwise you can track down some things in Google. If you don't have *!Zap* nor Darren Salt's useful extensions – there are two things for you to download.

I am obliged to point out that there is a competing file editor that is in the same class as *!Zap* and it is called *!StrongEd*. I did not like it myself, but I won't go into detail as the preferences and attributes of each editor come together to form what is known as the "editor wars". Both do the same thing, essentially, and that's that they *royally* kick *!Edit*'s ass *all over* the floor. If you've only ever used *!Edit*, or whatever (god help me if you're an `ARMBE` fan!), then you'll get two feelings out of using *!Zap* or *!StrongEd*:

1. A feeling of confusion. These editors are very capable and are *no* way near as complex as they look. Trust me, I've got a pitifully tiny braincell, and *I* cope!
2. Tingly-sexual feelings as suddenly stuff that was hitherto-quasi-impossible (ever tried to hack using `*MemoryA`?) suddenly become *so* blindingly easy!

I'll finish with a short message for those reading the PDF version of *Frobnicate* on their (Windows) PC...

We are talking about hacking protection systems under RISC OS on an ARM processor. The methods, under Windows, are going to be similar, but expressed in 80x86 assembler.
The "it should go without saying" message above applies to *you* too, and doubly so since it is 80x86 code you're talking about. Don't waste your time, or mine. Okay?

Happy hacking!

My PDAs

A “PDA” (`Personal Data Assistant`) can be a very useful thing. While *proper* PDAs cost over a hundred pounds (sometimes a lot more), I’ll be looking at two less expensive versions; but please note that it is not *all roses*.

For our purposes, keep in mind that the definition of a PDA is rather vague. The writers of the various PDA magazines, and those fortunate enough to own a *PalmPilot* will probably laugh at my 256K dinky-toy, and of my other PDA, they’ll ask “does it actually *have* an operating system?”
My reply is that if I wanted an OS, I’d use a frigging laptop. I’m typing this on my old 486 laptop while cooking pasta and listening to Laura Pausini (the Italian gets me in the right mood, you see). And, since this laptop runs DOS, can it be considered an OS? `:-)`
That’s the obligatory PC-bash for this article.

Anyway, look, I want something that fits in my pocket and will remind me what Glenn’s mobile number is, what my *own* mobile number is (like, how often do you call yourself anyway?) and remind me what I’m doing on Friday. I’d also like a memo facility – whether it be to jot down Norbert Dentressangle sightings, or to figure out how best to wire the IRQs in my homebuilt 6502 project while mom is driving me around the Nantes périphérique...
...and for these purposes, my two ‘toys” count as PDAs. Why? Because both *claim* to be linkable to a PC.

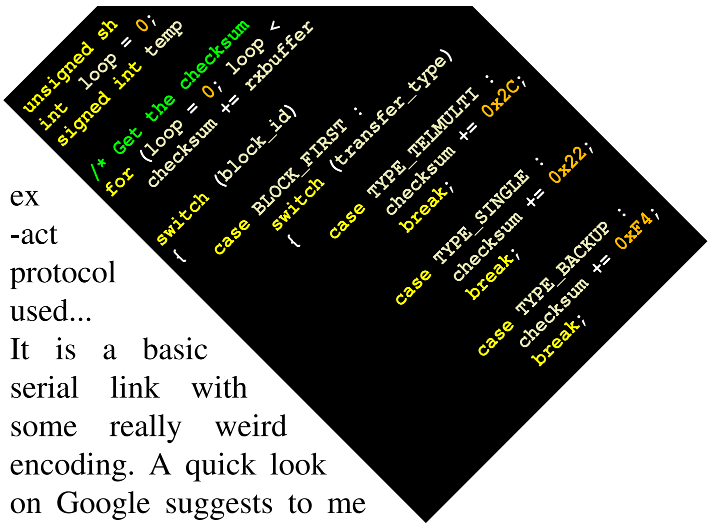
This last point is very important. There is absolutely ***no point at all*** in entrusting anything to a PDA if you’re at the mercy of a couple of CR2032s.
You might ask, what’s the difference between a ‘PDA” and an “organiser”? The difference is simple. A PDA links to a computer. An organiser doesn’t. Typically an organiser will have a smaller memory capacity. I saw one that offered 2K of memory, but once the storage overheads were taken into account, you got 1¼K! It should have been ashamed of itself!

So we’ll look at my first PDA. It is a `Sharp EL-6890`.

It offers 256K of memory to store telephone numbers (two directories plus ‘secret’), with space for address and email. A WWW directory (quite a useful idea!). A schedule with separate anniversary reminder. Memo. Two clocks (home and some-other-place). A calculator, and built-in conversions to/from various things. You can set up some of your own conversions, useful for pound-euro and the like...
The device sold for about £20 and included a PC link cable. An electroluminescent display allows it to be used in the dark.

Drawbacks? Mainly relating to power. It is powered by two wide flat button cells, with a third for memory backup. This in itself is not bad, however the backlight is fairly power hungry, as is the serial interface. The serial interface is a three-pin 2.5mm jack, and since it is serial, I’m guessing TX, RX, and ground. No power is claimed from the PC, so the PDA side is completely powered by the internal batteries. And, well, two button cells won’t power a serial line for long!

The software supplied is pretty basic. A third-party package offers more facilities, but it is commercial. So, me being me, I decided that I’d rather connect the thing to my RiscPC... It took a while to figure out the



```
unsigned sh
int  loop = 0;
signed int temp

/* Get the checksum
for (loop = 0; loop <
  checksum += rxbuffer

switch (block_id)
{
  case BLOCK_FIRST :
    switch (transfer_type)
    {
      case TYPE_TELMULTI :
        checksum += 0x2C;
        break;

      case TYPE_SINGLE :
        checksum += 0x22;
        break;

      case TYPE_BACKUP :
        checksum += 0xF4;
        break;
```

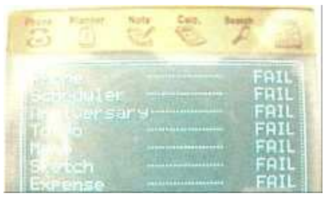
ex
-act
protocol
used...
It is a basic serial link with some really weird encoding. A quick look on Google suggests to me that Sharp are apparently rather protectionist with their protocols – and it seems that the Sharp PDAs fall into "families" each of which use a slight variation of the protocol. Are Sharp out of their minds? It seems to me that if I was making a PDA of *any* sort, I'd document the protocol on-line. Let the Linux hackers turn it into a `/dev` and let people like me write RISC OS software.
What would they care? More software support means a more desirable end product. After all, not *everybody* uses Windows. In fact, it seems like every day more people leave the Windows platform.
Now I don't expect Sharp to support all sorts of non-Windows platforms – hell, the number of different Unices around is mind-blowing. So why not leave the support to geeks like me? As it happens, I have a module I've been working on. It is called 'Sharp_Org" (name registered) and it will interface your EL-6890 to a RISC OS machine. In order to make things easy (maybe as a plug-in to *!Organiser*?) it will offer high-level SWIs to read/write data to the device. At this time, the receive (PDA → RISC OS) works well. The transmit and the higher-level SWI interfacing are not yet implemented.
Watch this space.
This was all achieved by experimentation and what amounts to brute force staring at screenfuls of interrupt-read serial line status bytes. I hated that job with a passion, but now I have working C code... `:-)`

You know what, if Sharp are willing to share information with me, I'd be happy to include support for anything similar. But, for now, I can only say it'll work with the EL-6890.

This brings us on to my next PDA. This is the one I use right now. It is a 'Navytech `i`PDA *k8*".

That is, as far as I can figure, the French branding. For others, it is an `i`PDA *k8* made by a company called Kessel Electronics.
It offers, I think, 1Mb of flash memory. The interface is via a touch screen that responds to a little pen-like stick. There is a facility to 'write' on the device, but it only understands American handwriting, and us Europeans don't write *that* badly! (no, not even the French with their barely-decipherable script) An electroluminescent display, telephone/address/info, planner, schedule, anniversaries, notes, drawings, and so on. Powered by two AAA batteries, this device has a really cool facility in that when the power fails, it will retain the information in a static state until you put in new batteries. Then it'll ask you if you wish to keep the contents in the flash memory (of course you say yes!).

Drawbacks? The most major is the supplied software ***does not work***. I've had 'most' of the data transferred to the PC, but on sending it back to the PDA (or in any direction subsequently), it fails. 'Linkage error", whatever that means...



```
Phone       ............  FAIL
Scheduler   ............  FAIL
Anniversary ............  FAIL
To Do       ............  FAIL
Memo        ............  FAIL
Sketch      ............  FAIL
Expense     ............  FAIL
```

Other drawbacks are less major – if you try to "compact" the memory while the batteries are low, you risk trashing all the memory – the compaction process is not, apparently, recoverable. Also, the calculator doesn't include conversions. They're not essential, but for those of us used to working in two systems (imperial and metric), it can be pretty useful. Sadly, this PDA, the one I use now – and I quite like despite the drawbacks (come on now, it only cost 35 euro) – is a mere toy. I do not feel confident in using it *seriously* until I can at least dump all of the memory to my RiscPC (or PC, if need be) and restore it to the PDA with *no* errors. It'd be nice to separate the data items, but most important is in knowing that I have a reliable backup.
Since the original supplied software does not work on my real PC, I have contacted Kessel Electronics, twice by letter (one to the US, one to Japan) and three times by email. They could not even be bothered to send *any* sort of reply. That really sucks.

Rest assured that I'll (eventually) crack this bugger and make a RISC OS module to talk to it. Here we come right back to the "are they mad?" question. I know nothing about Kessel – but they've not made a bad device for the inexpensive end of the PDA market. It is simple to use and contains a fair old amount of memory. They're kind enough to supply the PC link hardware. So, I have to ask myself, why don't they seem willing to share information? Surely having more software out there that supports their product works in their favour? The device doesn't cost that much, it is real small, it is quite study (I've not broken it in a year!) and I do wonder how many RISC OS users might consider it if I made a two-way link with the popular *!Organiser* software? I sure-as-hell would consider it seriously; if I didn't have a PDA.

I wonder if the serial failure is system-speed related? Having said that, I would be very disappointed if a 75MHz Pentium system was unable to talk to a 19200baud serial link *even if the software is in VisualBasic* (as it appears to be). I mean, really, 19200bps with a UART isn't exactly a squeeze; a 25MHz ARM3 can multitask to two 57600bps links at the same time, while multitasking, so how much processor power can the `i`PDA software reasonably need? The box says W95 or better, 486 or better, 8Mb RAM or more. I have better on *all* counts.

So what are the conclusions of this article?

1. I suppose you get what you pay for. Having said that, I can de-qualify it by saying that I can't afford (nor justify right now) the sort of PDA that the PDA magazines would consider to be a "real" PDA.

2. I'm surprised that companies making what is essentially a hardware device are so protective of their interfacing protocols.
Let's face it – how to talk to an `i`PDA *k8* is unlikely to be of interest to people who don't have such a device, or plan to write software to support such a device. It is left to those like myself to figure it out by looking at what amounts to be a bunch of bit patterns. The chance of my screwing something up is quite high – so Kessel (and Sharp) would benefit from sharing details as they can then know that those who are genuinely interested in supporting the product will read from the "official" spec, and not use *any* intuition or guesswork. So everybody can be happy. If Kessel or Sharp asked me to sign NDA, it'd be a tough call as it goes against everything I'm interested in doing here (after all, my interface software *will* be released for free); I'd have to weigh it against "personal gain" in that it would give me a lot more use of my PDA...

3. I'm *seriously* disappointed with the response (or lack of) that I have received from Kessel. I was really hoping for them to agree to help in some way, and not say "sorry, can't" which gets nobody anywhere.
What I was *not* expecting was total silence.
Is *this* some sort of administrative screw-up or is it *really* how they run their company? I will try contacting both companies again, and refer them to this article.

I really do hope that they agree to help. We'll see how it goes – watch this space.

HELP! If you have a Sharp EL-6890 and you are interested in being an *alpha* tester for the driver module, please contact me.
*Be **very** aware that the tests may well result in **total** loss of **all** data in your EL-6890, so don't contact me unless you're okay with this!*
Contact me also if you have something "in the same family" as the 6890, I can let you try the read-only code to see how far you get...

Qu’est-ce que c’est, ça?

PART ONE

There are, primarily, two types of software “out there”. There is the “free” stuff (this includes unregistered shareware), and there is the stuff that you pay for.

The “free” stuff deserves to have a software licence that states “*no liability blah blah no warranty blah blah*” because, essentially, some person took his or her time to create the software – probably of use to herself – then polished it up a little and released it because she thought it might be of use to you. A lone programmer, or maybe a team of two or three friends, they’re essentially doing you a favour, so their software can be “take it or leave it”. You aren’t paying them, no soft of contract has been drawn up between you. So they deserve the no-warranty no-liability status.

Commercial software, on the other hand, should fall under the Sale of Goods Act; basically in that there *is* a warranty and the company *is* liable. The liability can be directly related to the cost of the product, the “damage potential” and whether or not the fault can be considered wilful negligence.

Let’s compare two products: Our first product is a game, costing £16. If it acts up, you might suffer the frustration of a lost hour of play and maybe a crashed computer, but essentially is isn’t a lasting thing.

Our second product is a operating system costing over a hundred pounds (your computer came ‘supplied’ with an older version, of which few things want to work so you’re forced to upgrade), which is supplied with email software that has a long track record of security failures (*gee, who can I be referring to?*).
In this case, your computer may not exactly crash, but sensitive documents may end up mailed all over the place, without you ever being aware of it.
Every time the media talks about a new “internet virus” you begin to panic. You look for, and download, security “patches” that cause other, different, problems, and it all costs more in phone charges than it does to purchase the operating system in the first place; lest the mighty Americans forget that a good number of us *pay* (actual money) for phone calls, even ones to our ISP.

PART TWO

Most software licences state terms along the lines of “*blah blah blah single-user blah single-CPU blah blah*”. What this essentially means is that only one person is entitled to use this software.

Firstly, why the “single CPU” statement? Does it actually matter if your computer has more than one processor inside? What do you define as a “single CPU” anyway?
In the ARM computer, the on-board FPU is actually spoken to in terms of co-processor instructions. If you take an FP instruction, say, `FMLS F3, F4, F5` and disassembled it into a binary pattern, you’ll see it’s really a bunch of `CDP`’s and the like to co-processors 1 and 2. That, incidentally, is how the FPE module works, it hooks onto the “unknown instruction” vector and sees if the instruction the ARM didn’t understand is one that it does understand... So, the ARM has floating point, either in software or in hardware.
Older floating point (ARM3/FPA1) are two separate chips, while newer floating point (ARM7500FE or the more recent VFP ARMs) are all in the same silicon. Where, though, does the actual distinction lie? What constitutes a “second” processor? In the Acorn RiscPC range, the majority of users will have an 80486 co-processor. The price was right, and the computer actually has slots for two processors. Those that take their PC emulation seriously will have an 80586 co-processor. It’ll run Windows95, and the actual processor is the only part of the “PC” that is real. The rest is clever software. That, clearly, is a two-processor system, but *does it matter*? (actually, if we are to be pedantic, the ARM+FPE, the x86 and on-board FP could add up to *four* processors, two ‘real’, one ‘almost real’, and one ‘faked’)
So what if I have multiple processors? Hell, a modern PC has more processing power and memory on the graphics card *alone* than an entire PC from just a couple of years back. Does this count as a second processor?

Secondly, the vastly irritating issue of the “site licence”. I fully agree that if somebody wants to install a copy of a program for use by five hundred kids (or five hundred corporate drones), they should pay more. What I vehemently disagree with is the ‘assumption’ that you should pay more if you use the software on a network.

I will admit to openly breaking such licences with ***NO*** trace of guilt. Here I have a network consisting of a RiscPC, a real-PC, and an A5000. I run software to/from these machines, and often run the same software on more than one machine at any given time.

Why? *Productivity.*

I don't fancy waiting twiddling my thumbs as some tedious and boring task completes. As we all know, complex print jobs are not as fast as we'd like them to be. So why not run a second copy of my DTP software, and the printer driver, and get a second computer to do the actual printing while I carry on. There's no telephone line, this setup is purely an 'intranet'.

Mom's PC is not connected to the network (it tended to crash nastily if the hub wasn't connected and switched on, so I removed all the network drivers; it's not as if mom actually understood all that network stuff anyway). Mom sure as hell doesn't know her way around my heavily-customised RISC OS.

So whatever way you cut it, it's a single-user network. Why should *I* pay more just because I happen to be more clued up on how to get the most out of my available hardware? Bugger that for a game of soldiers!

PART THREE

The other day (in 2002, to be precise!) I walked into the bank and was unable to perform any transactions. Because "*the computer went down*". You could, apparently, pay money in but you couldn't take it out (just like a bank, to take take take). And like the sheep we all are, we just nod and mutter some vague agreement about how that happened at the library just last week.

What the f##k is the matter with us?

That is NOT a reason. At best it is a thin excuse.
You know, if I had more disposable cash than my current pathetic amount, I would actively boycott organisations that suffer from:

• **Regular computer failures**
From time to time every system crashes. Even *my* computer crashes, when there's millions of instructions combining the coding efforts of hundreds of people, things *can* clash. My playing with hardware vectors for 'fun' is a sure way to see the computer vomit over itself. :-) However, that is a domestic home computer. If it was running a banking system I would *expect* full logical analysis and flow charting to be performed, so the system failures are measured in *years*, and nothing less. These bank people get paranoid over losing a hundredth of a penny (it can accumulate to a staggering amount in little time), why aren't they as paranoid with the rest of their systems?

• **Lame excuses**
People, PLEASE BE AWARE that computers do not 'just' crash unless they are actually faulty. When my bank tells me "*the computer went down*", I expect to be told exactly why. This is *not*, and I repeat *not*, a matter of great secrecy, but the bank will tell you that about as readily as they'd have signed a legal document stating that your money would be safe over the Y2K changeover.
I expect to know because my confidence of the bank relates directly to the point of failure. This is none of that need-to-know crap, this directly affects me and I want to know what went wrong. If a similar error took out air control and planes couldn't land, people would set up "committees" to find out what went wrong. When the bank suffers a failure that renders your cash untouchable, we're fudged off with some excuse that "the computer went down". Judging by the reaction I get, I must be the *only* person that expects to be told how and why, in detail.
Look at it this way – if the bank system is out for an entire day because a harddisc failed, they're a crap bank (you can buy full-scale RAID solutions for a home PC these days, less then 500 euros for the whole kit'n'caboodle *including* the harddiscs). You can, obviously, cut them some slack if a stupid workman backhoed through the main fibre-optic cables.

So maybe more of us should *expect* an explanation. They aren't deities. They're basically "just another company".

• **Email viri**
It appears, at the moment, that the "MyDoom" virus is lurking, though it seems to be leaning towards being as much a damp fizzle as London's Y2K fireworks display.

Barely a few months go by before there is some other "internet" scare. And when you listen to the media, they always include the magic words "that appears to exploit a loophole in a Microsoft product". What they really mean, *and I'll say it*, is that Microsoft's Outlook mail client is a security failure (or maybe that should read "a security nightmare"). Is anybody bored enough to work out for me how many different software viri have affected it? Is anybody at Microsoft willing to admit that maybe putting *any* kind of script interpreter into an email program is a particularly losing idea? Yeah, okay, VB for Apps is a 'neat' idea, now pull in the reins and apply it *logically*.

I used to feel sorry for those organisations that were so badly affected. Now I laugh at them. You don't need to obey Microsoft's every command and download service updates unending, instead you can do something very smart and install a different program. Many exist. Indeed, some companies are wising up and installing an entirely different operating system.

You might think that Open Source software is screaming for people to exploit coding errors and bugs. Not so. People spotted and fixed the bugs quicker than exploiters could devise malicious code. So, today, ten years on, the Linux operating system is suffering problems of code bloat, and not rampant security failures. I **will not** have anything to do with any organisation that I know uses Outlook. Hell, not even my local library is *that* stupid.

PART FOUR

Ever heard of UCITA? If not, look on the Internet for opinion.
Basically, to dumb down a boring concept, UCITA allows a software company to sell you a licence to use their software, and they also sell you the right for them to slap you across the face afterwards, at their desire.
UCITA allows such things as:

• Exactly a year after installation, your software may refuse to run, popping up a message to "call 555-xxxx" to re-licence.
• When you call up "555-xxxx", the droid on the phone can command your software, remotely (if your machine is on-line), to de-install itself if she (the droid) is at all suspicious about whether your copy is valid. Or, if she's just feeling like being a total bitch. What does it matter to *you* anyway, your licence had expired anyway.
• Your re-licence may cost you more money.

Of course, none of this rubbish is stated on the box when you make your purchase. Advocates of UCITA complain about massive software piracy. Everybody else says, "*if the software wasn't so damned expensive in the first place...*".
This is now legal in several states in the US, and ironically several states have countered this by outlawing UCITA. Doesn't that say something?

It's only a matter of time before people lean on the European law makers with similar stupid ideas.

I'll finish here on the topic of UCITA. It took me two re-writes to remove all the obscenities. :-)

PART FIVE

It is perfectly valid to write a licence in "legalese", but *any licence written in such terms should only be applied to people who have studied law*. The licence should also be copied in "plain English" (or French, Italian, etc as is appropriate).
Licences that are *not* written in plain simple language should be *ignored*; and any legal dispute that may arise should assess the legal comprehension of the person accused and if it is considered that the licence is too complex, it is null and void *in its entirety*. Basically, I don't want to spend years attending law school so that I fully understand the frickin' licence!

CONCLUSION

It is time for a sea change in two different directions.

1. Licences giving **YOU**, the user, more rights and more say. Maybe a legal *right* where you are entitled to contact the company with your specific situation to have a bespoke licence for you. This *will* cost *no more* than 10% of the retail cost of the product.
It is time to rethink the idea of what licences are. Is it any surprise that people are ripping off software when licences these days are forever trying to take away more and more of your rights?
YOU ARE THE CUSTOMER. Why do you let them treat you like dirt?

2. NO, THE COMPUTER DID NOT JUST "GO DOWN".
Are they using under-specified "cheap" hardware? Are they using bug-ridden software? The recent power failure in the US shows what happens when corners are cut. In big, important, organisations, such things should be considered criminal liability. No time frame, no excuses. We must all stop accepting excuses about the computer going down. So what if it did? Isn't there a backup system? If not, why not? Obviously, the amount of spleen you went depends upon who you're talking to. If you local library's internet link is dead because lightning took out the ADSL, then you should be much nicer than if you are due your benefit payment and all the people in the post office just shrug and point to blank monitors.
We must start questioning what went wrong, and why, and getting real answers instead of phoney excuses.

Right now, a lot of people think I am *an annoying bastard* for asking these things.
I hope for a day, soon, when we *all* just *expect* genuine answers to these kinds of questions.

Programming Myths EXPLODED

I guess I’ve been reading too many ‘how to program correctly” books, and I’m sick of reading the musings of all of those over-zealous pricks who reckon they know best.

Let’s face it. The world in which people write functions no longer than a screen in size, and where each and *every* function has one clearly defined entry point (that’s the easy part) and one clearly defined exit point (that can be much harder) ... it’s an idealistic world.
If it were truly like that, we would *all* be coding in Pascal, and the OS would be so remarkably anal as to require *you* (the operator) to ask the OS permission if/when you need to fart.

The real world? It’s a bug-strewn mess littered with the wreckage of companies large and small, mangled dot-coms lying amid the ruins. It’s a scene that makes Iraq look like some sort of paradise.
While we should always aspire to something better, we should not get so caught up in the rampant idealism as to render our efforts minimal, if not rather useless...

So, those myths...

Every function must be less than a screen in length
This is complete and utter crap. Using BBC BASIC V, I can write an entire multitasking calculator program, and it won’t fill the text area of an 800x600 screen. On the other hand, in ARM assembler I’d be hard pushed to get a Basic Boring Bubblesort to fit into a screenful.
Why? One allows multiple statements (of reasonable complexity) on each line. The other allows only *one* statement of beautiful simplicity per line.

You should break your program down into small, useful, functions; but whether or not your function fits within a screen depends upon your screen dimensions and the language you’re using. Ergo, that rule is rubbish.

Every function should have one clearly defined entry point...
This, too, is nonsense. Unless you are writing amazingly hairy code, or assembler, you’ll only ever have one entry point. For example:

```
int main (void)
{
   ...some stuff...
int main_again(void)
   ...some stuff...
   return 0;
}
```

simply won’t work. I don’t think the C language even supports multiple entry points in a function!

There’s no need for them, and they’re pretty irregular, hence this rule too should be counted as “static noise” to fill the pages of books telling you how *not* to program. Like, *who* would write that sort of rubbish anyway?!?

...and one clearly defined exit point.
Well, excuse me but I prefer efficiency.

For example, we shall consider a function to read values from a file to check the file is the one we’re looking for:

```
read bytes
test bytes
read more bytes
test those bytes
read some other bytes
test them
```

If the test fails, we should drop out of the function with a suitable ‘hope” return code.
Why? Efficiency.
In the real world, let’s assume you’re cooking pasta but you’ve run out of pasta. Do you carry on and boil water for ten minutes and heat up the sauce? Of course not! As soon as you realise you’re out of pasta you stop, grumble, and look for some baked beans...

This, too, should be how your program works. There are three ways we can do this.

a. An ugly set of nested IF statements
b. A "goto" to the end of the function
c. Multiple exits

Let's dissect. The thing wrong with our nested IFs is that they're ugly. If something is complex (like figuring out TIFFs) then we understand nested IFs are a necessity. However using them simply because somebody decided on only allowing one function exit? It is stupid.
A goto... Well, we'll cover goto statements later. That leaves us with multiple exits.
In the spirit of optimisation, I would *always* choose multiple exits. We test something, then drop out as soon as any test fails. Quick, simple, elegant.
Some people might say that I can avoid this using sub-functions for the tests themselves. Well, here we have two problems. The first? Moving the tests to sub-functions not only wastes time in the function call, but it also reduces the readability of the program, as simple blocks of code will now be separate (and possibly disparate) chunks of code. Secondly, it's a useless idea. So the chunks have moved. You'll still need if/goto/exits in order to know which functions it is safe to call. If you can't handle multiple exits from a function, *get over it*.

GOTO is evil
First, get a life. There is no such thing as "evil" in a computer language. It's just a collection of statements that do stuff.

You should, really, purge all goto statements from your code and you should avoid their use, as they can make a program difficult to develop and debug. One example is that - even with multiple exits from a function - when you enter and exit a function it is 'safe' in code terms. What if you "goto" outside the current function? Or into a sub-block with it's own local variables? The potential for chaos is quite high with misuse of goto. However, goto is *not* evil.

Those of use who program in assembly will read this and snigger to ourselves. `BEQ` is a conditional goto by another name. `JMP` is yet another. Need I go on?

The question of code commenting
Code should be commented. Here are two examples of bad code:

```
DEFFNconvert_to_string(offset%)
  SYS "OS_IntOn", offset% TO output$
=output$

MOV  R0, #34     ; make register 0 = 34
```

The first example may cause confusion with a great number of programmers. It is obvious what is supposed to happen – given an offset, we extract a string from it. However, is this some special undocumented feature of the software interrupt used to turn on interrupts?
No. It is a side effect of BBC BASIC V. IntOn preserves registers, so we pass the offset in R0 and we tell BASIC that we expect to see a string from R0 on exit. As IntOn preserves registers, and interrupts are usually on while executing a user mode program, the SWI itself does nothing. It is BASIC doing the work. Now, a comment might have made that clear. What'd be better, though, is to stop being clever and read the string byte-by-byte like everybody else does...
The second example illustrates the other extreme. Yes, MOV R0, #34 puts 34 into R0. Anybody that doesn't know that has no business looking at the code – they'd only be wasting their time.

The rule I use is to comment fairly liberally, and be nice. Remember, you may not be the person maintaining the software in five years. And, hell, in five months you might not even recognise half of the program. So comment well.

If your tutor marks you down for having too many comments, tell them they're an "*over-zealous prick*" and tell them that I said so.
In no language that I've used has the degree of commenting made *any* difference to the final "production" program.Even in BASIC, you can run your code through a "cruncher" that shortens variable names and strips out comments. In compiled languages (C, pascal, Visual Basic, assembler...) you could include the entire works of Shakespeare as comments and your finished program won't grow a single byte. It'd just take longer to compile, and it'd make for *really* boring reading, but otherwise...

Top-down or bottom-up?

This one is easy. Learn the idea behind them both. Try them both out, and pick the one that suits *you*. Ignore your teacher if they command you to use a specific style, it may suit *them*, but we're more interested in the one that suits *you*.
Sorry, there is no short-cut. You might "cop out" and use the one your teacher recommends, and it might be the best for the way you think. But you won't know if you don't try...

I tend to code with my main procedures first, and sub-procedures following. The order of sub-procedures in a single-source program is usually the order the sub-procedures are written; it is much tidier in a multiple-source program in which related procedures are grouped into modules. A teacher may say that is a bad way to program. Well, I invite them to sit down and try *!Zap* where I can press `Ctrl-L` and instantly see a listing of all the procedures in the source file I'm looking at. I guess the sub-moral here is *learn and understand your tools*.

Inspirational programming.
This is a polite name for the act of sitting down and bashing out code. No flowchart, no planning, you just "do it". This is not necessarily a bad thing. I code like this quite a lot as I am trying out ideas (and I'm lazy). I want to see results – even crappy results, I can refine it later. If I did all that flowchart rubbish, I'd get bored and *no* code would ever be written.
So, don't be afraid to sit down and just 'write' code; but don't do it for anything you consider to be really serious.
Have you ever watched the painter Rob Ross on TV? He starts his program with an empty canvas and finishes some twenty-five minutes later with a beautiful masterpiece. He doesn't pencil in guidelines and plan his shapes, he simply 'paints'. Spontaneously.
You can code like this too. I wrote my PtrIIC (IIC protocol via a specially-wired parallel port) module in this way. I knew what I wanted to achieve, I had a well defined protocol to follow, so I sat down and began coding. The module has made it to version 0.03 and it works beautifully. There is one thing I'd like to fix – that is the write-byte code is duplicated. Why? Because the SWIs to directly control the IIC bus were an addition, and since the write-byte there has different exit conditions than the normal write-byte, I decided that code duplication would be the easiest option (we're talking about ten instructions!). It is a concern to me, but in a module under 2K in length, it isn't the end of the world...

Code is art
It is indeed. However if your friends and/or teacher(s) judge your 'art' badly (with meaningless phrases like "not his best work"), then here is a little story.
I watched a programme on BBC 2 a few years ago. An excitable woman was exalting the wonderousness of a bunch of coloured squares and circles on a plain white canvas. "My god!", I thought, "I was drawing crap like that when I was in kindergarten!"
I look in antique shops and I see the most ghastly furniture for an equally ghastly price. It's the sort of stuff I can imagine people having difficulty *giving* away, yet they expect to sell it for hundreds. And we simply must consider the fashion abominations that fill our screens and minds every time one of those pointless awards shows comes on TV. In short, your 'art' may not be 'art' to other people. *Like you care?*

Programming is like learning a second language.
In a way, it is. Even with C's overloading of the asterisk (the little `*` has some eight different functions, several of which may be brought into service in one line of code (!)), the computer language is tightly defined.
Logical.
Precise.
Why not try learning French (*Je t'aime!*) or German (*Ich liebe dich!*) or or Russian (Пока! Не забывай писать.) and the like, and wait until you get into the verb conjugation nightmare. Is the verb first-person or third-person? Passive? Who/what is it relating to? Which of the many tenses is it in? French has a peculiarity in that the genders of relating words depend upon things that may seem illogical to English speakers – *he* may own a car but words to do with the car are conjugated feminine because *voiture* is a feminine word (the 'he' owner is ignored). Learning a spoken language is so much more difficult than a computer language...

The role of your OS
Possibly the most shocking thing I ever read was a discussion on how "bad" RISC OS is because it can allow a user-mode program to bring down the machine by bad memory accesses. On the older machines, typing into BASIC `!8=0` will lock up the machine (it replaces the SWI hardware vector with zero, instant chaos). True, it *is* bad of RISC OS not to protect critical areas of memory from errant user-mode programs, especially since null pointers are a common bug (later versions of RISC OS make "page zero" read-only in user mode). However the assertion of the author was that a decent OS (in this case, Linux) will catch such bugs making the program 'safe'. What a load of crap!
Okay, so a strong OS will protect your system from the nastier effects of a badly written program, but that doesn't mean diddly-squat. A bad written program is a bad written program. It doesn't matter what your OS does, the program will still be bad.

As I am a RISC OS advocate, I'll quickly add that RISC OS is a system designed for use by hackers. Why else do you think we have the SWI "`OS_EnterOS`", which puts you right into supervisor mode? People often say the Unix clones are systems by-hackers-for-hackers but they forget that one of the primary points of Unix is it's extensive multiuser support. This, over all, explains why Unix traps all sorts of things in a 'safe' manner. It is assumed in RISC OS that if you want to poke crap on top of the hardware vectors... well, either you're an idiot or you know what you're doing. `:-)`

BASIC encourages bad programming style
This is often heard, and usually one can trace it back to lesser BASICs that required extensive use of GOSUB because the flow-control was so poor. Modern BASICs (some even allow you to work with the object-oriented method of coding) are far superior to the BASICs of old.
But, you *really* want to know why BASIC has a bad reputation?
It's because of a veritable army of coders growing up with the BBC micro, the Oric, the Vic-20, the Speccy... It was a simple language, it got quite a lot done. Okay, maybe every one of these self-taught programmers taught themselves bad habits.
Well, I ask you, where were all these wonderful modern languages then? Why isn't there a fully-blown ANSI C compiler for the Oric?
Oh, wait, these small home computers probably wouldn't run a C compiler sensibly from tape; memory (lack of) issues aside.
Well, there's your answer. So maybe instead of maligning BASIC, we should thank it for getting a generation interested in programming. How about it?
As for the idea that BASIC encourages bad programming style; it is possible to program badly in any language. No compiler is smart enough to know "what you actually wanted". I notice the Norcroft C compiler pops up errors about "expected X, assuming X" – but thankfully (since it's assumptions are wrong as often as they're right) it considers the assumptions to be errors so the compile fails.

The generation gap
You know what? I actually read a very funny article as a "white paper" for the Mastering series on Visual Basic 5. It was included in a demo version, and I'm *so* sorry I have lost that CD-ROM as I could write an entire article simply pointing out the faults in this white paper.
One thing I remember clearly was a comment on how BASICs should not have things like `PEEK` and `POKE` and how good VB is that such things have been removed (after all, folks, VB is so advanced that you don't need to know the addresses of things, blah blah).
Well, this perfectly shows the endemic lunacy demonstrated by today's young coders (to make a mistake *that* bad, this person must be fairly young).
The lunacy? To slag off things that they see as poor design and unnecessary disaster areas in BASIC. After all, why on earth would anybody need to `POKE` bytes into memory?
Actually, the reason is very simple. Many of the original home computers offered small amounts of memory – the BBC Micro offered 32K, the Oric offered 48K, and so on.
From this, take away screen memory. The BBC's MODE 7 (teletext) was highly innovative in that a single screen consumed only 1K. Many many BASICs did NOT have a built-in assembler (another plus for the BBC micro), while others had assemblers so restricted that it is a wonder why they bothered.
Into all of this, enter `PEEK` and `POKE`. It would be a veritable nightmare to work out the correct 6502 or

Z80 code to then `POKE` into memory, but the older generation of programmers did exactly that.
BASIC's `POKE` (or ? and ! indirection in BBC BASIC) was the *only* way.
People have entirely disassembled their OS armed with nothing more than a 6502 datasheet and the `PEEK` command. It is the way things happened in the mid-'80s. End of story.

Pascal is the best language to learn
I bet you aren't surprised to learn that I disagree. People say Pascal is good because of it's strong typing and all the other rules that make it possible to teach you good habits. I maintain that you need to become fairly proficient in a language before you understand why all this is important. And by the time you are proficient in Pascal, you might realise that next to nobody in the "real world" uses Pascal.
I learned Pascal at college, and since then I've not touched Pascal†. As awful as it might seem, we should learn Visual Basic, C, C++, or one of these other "everyday" languages that college-leavers will be able to make use of...which isn't Pascal.

As to the question of which language is best? There are so many factors to consider, and so many different situations that it should be plainly obvious that there is no "best".
I want to write a "desktop silly" with eyes that follow the mouse. I'd probably do it in BASIC, as it is quick and simple and I could have the program running in an hour or two.
I want to write a module to provide IIC over a parallel port, well that's BASIC out. We're left with C which requires about 6K of baggage simply to get going; or assembler. Since we need direct access to Timer1 to tick away a few microseconds, it's assembler (but note, we can mix C and assembler quite freely if we wanted to). In a good number of cases, there is no *one* language that is suitable. Sometimes you get the best results from mixing languages.
Hence, it is a fallacy to assume that there is a "best" language.

† Actually, I used Pascal once more. I write a program containing BasicAOF, C, assembler, and Pascal code, all APCS compliant. It worked too! Cool. `:-)`

C is portable
This is a lie promoted by "lesser" books written on the subject. The standard C library has many omissions – you cannot even determine the current directory, nor directory contents using standard calls. You cannot arbitrarily reduce the size of a file using standard calls. Heck, you cannot even clear the text screen in a way that'll work on *any* C.
Sure enough, C is a standardised language so the C written for one ISO (ANSI) compiler will compile on another – it just requires you to supply suitable libraries or alter the code to suit your system... ...sometimes, this is more trouble than it is worth, as systems themselves can behave quite differently.

That's why RISC OS has no Mozilla. But we do have SimCity and SimCity2000. Some you win, some you lose...

Conclusion
I hope this article has given you something to think about. After all, for a number of people, coding is a fun way to pass the time. If you have to endure a dozen pointless "rules", the fun will vanish...
I'm a big believer in your job should be non-boring. As a Care Assistant, there were highs and there were lows of unbelievable tragedy (imagine sitting with a dying person who knows she's dying and is asking you what heaven is like while confessing every secret she thinks might be an issue "up there"), and then there are the amusing moments (ask an African person to do Last Offi ces and see how many colours they manage to turn before running away; or try explaining to the Charge Nurse why it *is* important to cover the mirrors and leave the window open...). Every day was different and every day was something new. It was mostly enjoyable. It was *never* boring.

If you plan for a career in the IT sector, or in *any* sector, set yourself the same career goals – after all, who needs a boring and sucky job?

This episode brought to you by
the Hissing Spinach Workshop.
Today's letters were 'R' and 'O'
and the number was '5'.

Redneck Rampage *rides again*

and why violence is good

In a *first* for *Frobnicate*, we shall review a "PC" game. This is not to say there are no review-worthy RISC OS games; it's just that I don't have any! **Please note that you may find some of the opinions expressed to be disturbing.**

In France is a shop called "Noz" (I guess that is some variant of "nose") that seems to end up with a strange assortment of end-of-line things. I quite like it as they sell *d'Oro* biscuits from Italy (sorry Bretagne, but I don't actually like those *sables*). Once in a while you can fi nd some good stuff in amongst the piles of "who the hell would pay for that...ever?"

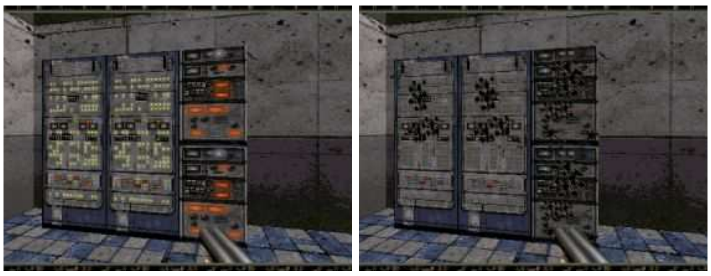
Among my acquisitions is a game called "Redneck Rampage Rides Again" by Interplay. Seriously, I *had* to have it for the title alone! It runs from DOS and is happy in a 16Mb machine with 1Mb VRAM and a crappy P75. It's not often I see games that will actually work on my old PC!
So I take it home and unwrap. The user guide is in the form of a newspaper and it is written in French, like you'd expect. There is no English version on the CD-ROM like the ReadMe says, so the best I can guess is there is lots of alcohol, lead, chickens, killer mosquitos, and aliens.

So, you thought cheerleaders were cute and fluffy, if maybe a little bit dim? Well meet *Daisy Mae* and her friends. This little chick is a psycho and will happily do the splits and flash her blue panties at you if it'll give her a chance to break your neck. She's a tough one too, just punching her won't keep her down. You'll need the pump-action to silence her incessant cheer rhymes.

That's the thing about this game. It is one of a long string of "*if it moves, shoot it*" games. You, obviously, are a "redneck". For those who have missed this particular derivation of American culture, my dictionary describes it as "***1.** a white member of the rural labouring class of the southern USA* ***2.** a bigoted reactionary*", which is sickeningly politically-correct. Your stereotypical redneck drives a pick-up truck and carries a gun and sees nothing wrong with using blacks for target practice. Also, there's nothing wrong with having sex with your own sister and monogamy is a word they can't spell, never mind know the meaning of. The types of people that end up on Jerry Springer that aren't "poh whaht trayash" are probably rednecks. But, hey, a dictionary won't say that.

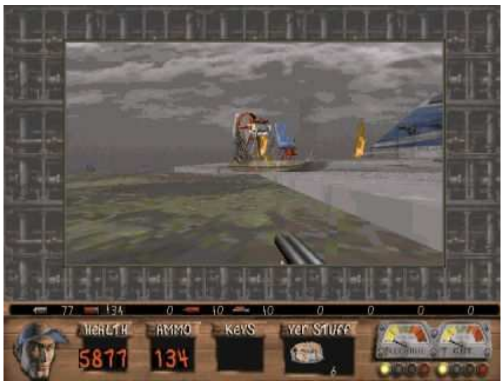
Episode one begins with us climbing out of a UFO crashed in Nevada. We have to break into a military base and work our way around solving a variety of puzzles in order to fi nd the exit and progress to the next level. The game-play is very reminiscent of the likes of *Doom* and *Quake*.

There are plenty of amusing little things along the way, such as the UFO that is in the base just happens to have a hammer-and-sickle emblem, and the letters CCCP. Well, who'd have guessed? :-)
Also, you can shoot the equipment; here is the Microsoft website server *before* and *after*...

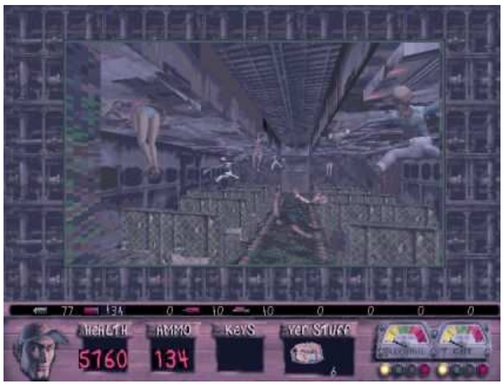


If you don’t fancy running around a military base, you can elect to try episode two that drops you in a swamp. As you prepare to walk, you witness a plane crash. Then, it’s time to check your ammo and get ready to kick some ass.

After dispatching the bad guys and running from killer mosquitos, you come across a fan-boat as is often used on the *Everglades*. This is a fun little thing. Sadly, you’ll encounter other people on them chasing UFOs through the swamp. If you’re lucky, you can shoot down a UFO or two and have fun ploughing through the aliens that eject before the crash. Sooner or later, you’ll come across the wreckage of the plane. As you can see, I’ve shrunk the game view size to speed it up a little. Those with meatier PCs don’t need to do this!



It is essential that you hop off of your boat because an important key is hidden in the cockpit of the plane. This picture shows us inside the plane, and all the dead passengers. And, yes, the panties of *yet another* cheerleader.



You’re a redneck, remember, you are interested in what’s beneath her skirt – her face is, well, it’s the boring bit above the breasts...

And so the game continues.

Now we come to the ethics of *cheating*. You see, some people consider games like this as a challenge. A thing to be battled, and won. For me it is a little bit different. I consider games like this to be a *therapeutic exercise*. I am, essentially, a non-violent person. While I liked (*past tense*) making small explosives, to use anything to harm another person is seriously bad for the karma. Those that know me will know I’m very laid back and few things bother me. I made a good Care Assistant as I wouldn’t lose my temper with the evil old woman that nobody else wanted to go near. That doesn’t mean it doesn’t annoy me, it just means I don’t show it. Sometimes, especially when in a supermarket surrounded by inconsiderate bitches who insist on dragging around a pram when the obnoxious bastard that belongs in it is running riot in the shop along with other kids, I get the urge to find myself a whole heap of ammo and turn a boring day into CNN ‘Breaking News”.

Why don’t I?

Because, unlike some people I am quite well aware of what is real and what isn’t. And unlike some people, I am quite well aware of what is right and what isn’t. And, finally, while I might like to rationalise things into black and white (grey bits annoy me as they can be used by evil men (re. Bush) to make a wrong become a right because it can’t be conclusively proven to be a wrong (re. Iraq)). There are, also, varying degrees of wrong. Letting your child run amok is wrong, yes. But blowing its brains through the wall is *slightly* more wrong.

Enter, thus, the violent game.

Critics and emotional do-gooders are busy telling us that such things are in seriously bad taste and how our society would be much better without them. Sorry, people, but that’s a whole heap of bull. There is *nothing* wrong with violent games. As much as we humans might hate to admit it, is is *us* that is the violent one. We simply delude ourselves to believe otherwise. You might reply ‘oh yes, back in history, but we’ve changed”. Sorry, but that’s simply more delusion.

It was a horrendous event when two airplanes crashed into the World Trade Center, and it showed a bit of planning to pull off such an event. And, yes, it has made the world a different place. However I fear that in my lifetime a nuclear weapon will be used in the name of terrorism. And amid the hue and cry, litigation and counter-litigation that will follow, instead of asking "why did they do it?", we should ask ourselves "why did they build the damned thing in the first place?"

It is said that only five nuclear weapons, strategically placed on fault lines, and set off at once, will be enough to tear our planet apart. Until the end of the "Cold War", America and Russia had much more power than that pointing at each other. And who else has nukes these days? All these little countries want to have nuclear weapons as they see them as the ultimate deterrent. What they fail to realise is that the modern nuclear weapon is too much. It is no longer a deterrent, it is almost amusingly unreal in it's destructive capabilities. Entire countries can be made uninhabitable for *tens of thousands* of years.

Given that the recent news is that apparently *everybody* except Tony Blair knew 'the truth" about the so-called "weapons of mass destruction" that Iraq supposedly had (you know, the "ready in 45 minutes" was a speculation and not an actual fact), and that Tony Blair was the one who lead England into the war alongside America; we should begin to realise the power that one person holds. Is Tony Blair the man who can say yes or no to the deployment of the nuclear weapons that England holds? If not, who is? What safety checks are there?
A "war" in Iraq is one thing. Nuking the place... That is something that should never happen, not even in times of war. *It is too much*, and as countries argue and reason for why they need nuclear weapons of their own, we come full circle with the realisation that *they have not figured it out*. Because it is *us* that is violent. Not our movies, not our games.

Sure, there are people out there who will watch something violent and want to do it. What kind of logic is it to say "because of that, we'll ban all violence in movies"? Hey, people, why not ban sex because some people commit rape? Why not ban cutlery because some people kill each other with them. Terrorists used planes to kill a few thousand people at once. So, no more planes. No more staple guns. No more wine bottles, electricity, stiletto shoes...

You might think I'm a complete madman for writing this. Fair enough, that is your privilege.
It is my privilege to analyse the thoughts and feelings that I have and work out what they mean.

The conclusion I've reached? I *enjoy* murder on a massive scale. In fact, the more people I can mow down at once, the better. *But **only** in a computer game* (or maybe a movie).
It is *therapeutic* as it is a *safe* way for me to release any anger or aggression that I may have. Nobody actually dies. Nobody actually gets hurt, though my wrist will ache after a few hours of mayhem.

This brings us to the cheating. I don't *want* to lose because I'm not actually trying to win. So I will look for "cheats" to use in the game. In *Redneck Rampage*, it is a simple matter to edit the user definition file to give yourself a starting 'life' value of 32,768 instead of the usual 100, along with lots of ammo. It's a PC game, remember, so it's a signed 16 bit integer. You can't give yourself 99,999 as you'll instantly die.
Why?

```
    99999 AND (1<<16) = 65536
```

is the same as:

```
    -1 AND (1<<16) = 65536
```

We'll end the lesson and I'll point out some other cheats that you might find useful, should you play this game:

`rdhounddog`	Toggle "god" mode
`rdall`	Get all (guns, keys...)
`rdmeadow#**`	Jump to ep# level**
`rdshowmap`	Toggle map show-all
`rdjoseph`	Get the motorbike
`rdrhett`	Commit suicide (!)
`rdaaron`	"Mushroom mode"

There are more, I'll leave them to you to find out what they are.

This game? I'd give it 8/10 for amusement and bump that up to 10/10 because it only cost me three euros.

DeskLib v2.30 [RM/32]

http://www.heyrick.co.uk/software/desklib.html

The library to be known officially as "*DeskLib 2.30 [RM/32]*" is a 32bit version of DeskLib version 2.30.

I cannot take credit for DeskLib. It was originally written by Julian Smith and other people. I have simply made this version 32bit and (finally) incorporated all the stuff I kept meaning to add... `:-)`

Why DeskLib 2.30? That's ancient!

Yup. That's right.
However... Since DeskLib v3.20 and the later "Desk" library required all of the functions and globals to be prefixed "`Desk_`", I never bothered to do an upgrade.
I later discovered that DeskLib 3.20 (& Desk) attempt to use a central error handling mechanism. This is good in that it'll catch stuff that I (or you?) might miss, but it'd royally screw up my own error handling. I don't mind upgrading to newer versions of libraries, but I don't fancy rewriting half of my applications in order to do so. So, I stayed with DeskLib v2.30.

Don't you suffer name clashes?

Not a single one!

I see people with DeskLib, RISC_OSLib, OSLib, ThisLib, ThatLib, and even SomeOtherDamnLib... *WHY!?* I can understand UnixLib if you're into porting stuff, and I've used Jason Tribbeck's ROVLib when working on NewsAgent, but I myself (or "moi-même" in French, I think that's cute) only use DeskLib.
Indeed, all of these libraries offer the same RISC OS functionality for stuff like Wimp, SYS calls, and all the cruft that you wouldn't ever find in a 'standard' C library. Apart from the supposed "typing", how does OSLib differ from DeskLib when you get down to it? My logic? Pick one and go with it. I have. DeskLib.

Actually, DeskLib *may* be better (than OSLib) as the library itself offers routines for attaching *events* to windows, icons, and the like.

If you are like me and you use SharedCLib and DeskLib *only*, you'll have no problem.

If you use a billion libraries that all do pretty much the same thing, you might run into all sorts of chaos.

How to use it

For you, and for me, if you use DeskLib 2.30, simply recompile with this library!
It is called "DeskLib32" so you can bung it in your DeskLib.o directory and use it alongside the older 26bit version, should that be necessary. Note that you will come unstuck if you try using 26bit code in a 26/32 neutral application...
Simply alter your MakeFile as required.

Is it *really* that simple?

Yes. When my *!OvHTML* went 32bit, the library stuff was 'converted' simply by changing the library "DeskLib" in the MakeFile to "DeskLib32". Obviously some of the assembler in the project needed alteration, but so far as the library support it concerned, it wasn't diffi cult.

There's already a 32bit DeskLib!

Version 2.30?
There is, if I remember correctly, a 32bit version of *Desk*. I believe Peter Naull's website links to it (sorry, I don't have the URL to hand).

What new stuff is in this version?

Turn the page...

API changes:

It may be assumed that if nothing is mentioned, then the details in the existing DeskLib v2.30 header applies...i.e. no API change.

In all cases, if you are calling *any* of these functions, or any other DeskLib function, from assembler – please be aware that flags are *not* restored on function exit.

BackTrace (new – based on DeskLib 3.20):
Added "`BackTrace_OutputToStdErr`".
Added "`BackTrace_OutputToStreamWithPrefix`".
Added "`BackTrace_GetNestingDepth`".
Added "`BackTrace_GetFrameInfo`".
Added "`BackTrace_GetFunctionName`".
Added "`BackTrace_GetCurrentFunctions`".
Added "`BackTrace_OutputToFFunctionWithPrefix`".

Note that file pointers, if required, are *standard library* file pointers (i.e., the "`fopen(..)`" kind), not the DeskLib ("`File_Open(..)`") kind.

ColourTrans (updated):
Added "`ColourTrans_ReturnColourNumber`"; returns '-1' on error.
Added "`ColourTrans_ReturnGCOL`"; returns '-1' on error.

These return '-1' and not '0' as '0' is a valid colour/GCOL.

Added macros "`ColourTrans_RGB`", "`ColourTrans_SetGCOL2`", and "`ColourTrans_SetGCOL3`" from DeskLib 3.20.

CPU (new):
Added "`CPU_Mode`".
Added "`CPU_ID`".
Added "`CPU_Type`".
Added "`CPU_Class`".

These functions return the expected results on an ARM710 RiscPC, and may need testing on other machines. Note that the 32bit library appears to make you believe you're running in 32bit.

Dialog (bug fi x):

Fatal bug which caused crash (of entire machine...) should be fixed. This occurred if you click the close icon of a dialog displayed as a menu, i.e., one that hasn't been opened 'static'. Was fi xed in DeskLib 3.20 (and later) sources, but present in 2.30...

DynamArea (new - based on DeskLib 3.20):
Added "`DynamicArea_Create`".
Added "`DynamicArea_Delete`".
Added "`DynamicArea_DeleteAll`".
Added "`DynamicArea_SetSize`".

File (updated):
Added "`File_CreateDirectory`"; returns '0' if failed.

Hourglass (updated):
Added "`Hourglass_LEDs`".

KernelSWIs (partially based on DeskLib 3.20):
Added "`OS_ReadVarVal_GetLength_Raw`".
Added "`OS_ReadCMOS`".
Added "`OS_WriteCMOS`".

Menu (updated):
Added "`Menu_Recreate`".

Misc (new):
Added "`Misc_WordAlign`".
Added "`Misc_PageAlign`".
Added "`Misc_PageSize`".
Added "`Misc_PageCount`".
Added "`Misc_MemorySize`".

Module (updated):
Added "`Module_Lookup`".
Added "`Module_LookupAddress`".
Added "`Module_LookupAddresses`".

Resource (updated):
Added "`Resource_SetResDir`".

The variable "`resource_pathname`" is now longer, to allow for this.

Str (updated):
Added "`strcattcr`";
like "`strcatcr`", but allows TAB characters.
Added "`strcpytcr`";
like "`strcpycr`", but allows TAB characters.

Terri (new, *incomplete*)

```
char *Terri_ReturnPath(char *resource_path);
```

This function only really exists to support "*Resource_SetResDir*", it is described here for completeness and in case you need it...

The 'Terri" module will eventually include a variety of other Territory-specifi c functions.

Wimp (updated):
Added "`Wimp_PlotIcon2`".

Window (updated):
Added "`Window_Open`".
Added "`Window_ShowCentred`".
Added "`Window_ShowWherever`"
(and "`Window_ShowWhereever`" alias).

This is a work-in-progress, though I've not worked on it for a while (other things to do, you see!), however I hope before too long to have the following included within DeskLib v2.30 [RM/32]:

- Support for the serial blockdrivers
- Support for sounds, possibly *PlayIt* veneers
- Territory support (like, given a number, format it as a currency value, 1234.56 → '£1,234.56')
- Time conversions and validations
- etc...

If you have any ideas, please let me know!
http://www.heyrick.co.uk/software/desklib.html

In the next issue...

we'll look at how to install a ***Digibox*** abroad with *minimal* expense, what the options are, and what you can watch *without* a viewing card...

FOR YOUR INFORMATION

No satellite signal is being received

Watch this *space!*

The Wrap Party

Well, here we conclude the *largest* issue of *Frobnicate* ever written. It comes about a month behind time, but that's to be expected when I'm doing this myself. `:-)`

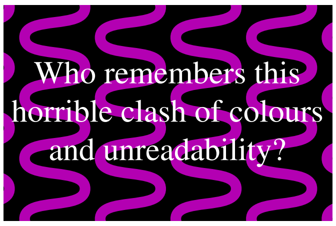
What I would like *from you* is some feedback. Did you like this issue? Would you read future issues? Most importantly, will you send me articles to publish?

I am happy to be your editor. I am happy to take time to put together *Frobnicate* and get it ready for you so all you need to do is download it and press the `Print` key. But I can't do this alone.

This magazine, and its future, lies in your hands...

Writing *Frobnicate* in *Ovation Pro* was an absolute delight! As you can see, I have retained the largely-simple format of earlier editions. This is not because I am lazy, but because I have seen some truly ghastly colour clashes in the professionally published magazines, and I'd rather have lots of empty white-space than something that is unreadable.

I extend this thinking to my website as well. I'd rather have something that looks plain but carries the content, then something the looks "sexy" and isn't worth reading...


Who remembers this horrible clash of colours and unreadability?

The three features I found *most* useful when designing this issue were:

1. The "undo". It is remarkably easy to make a mistake. A good example is editing a draw diagram. You want to select a line to delete it, and you click a couple of pixels too far to the left – selecting something within *Ovation Pro*. You hit `^X` and nothing seems to happen. Then you realise with horror that you just deleted something that you spent several minutes trying to get right. Unlike Windows, RISC OS doesn't pop the 'selected' application to the foreground, so you didn't know the selection went to an entirely different application. Panic? No, none whatsoever. Just bring *Ovation Pro* to the front so you can see it, then press `F8`.

2. The ability to import JPEGs. This is extremely useful as it allows me to trade off perceptual quality against file size. In the old days, with sprites, I had to trade off pretty much all the quality for a small size, and it was a monumental pain. Hell, some of the images in early issues are *so* bad it is embarrassing! Later versions of *Ovation* supported JPEGs, but I didn't feel it was something that I could rely on, as many people (myself, for ages!) were using a JPEG incapable version. Now? *Ovation Pro* does it as seamlessly as could be.

3. The variable-sized frames. This allows you to define an image frame, then set it up so the text wraps itself to the contours of the image. This means that we can now have a variety of non-rectangular images. It was possible to do this in *Ovation*, but if I tell you how I did it, David Pilling will never speak to me again (I'm not sure he has recovered from my use of lines of 1pt text for horizontal adjustments!). This text, as you can see, has wrapped itself to the left-most outline of the lovely *Elena Sokolova*.

This is the first issue that I have written using *Ovation Pro* and I can honestly say that I found the software a pleasure to use. I hope you submit articles as I look forward to doing this again.

Thank you *so* much for reading *Frobnicate*. Goodbye.

Rick
2004\02\27