FROBNICATE

Look what I photographed in our back field...

Hissing Spinach
Summer 2006 Issue 27 €0

Contains sustained moderate geekery.

Index:

[#27, 2006/06/21 20:01 CET]
(Summer Solstice)

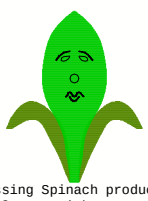
Credits:

Designed, written, and created by Richard Murray.
"Ewen's multi-satellite setup" written by Ewen Cathcart.
"Mom's Page" written by Stephanie-Jane Murray.

Images that are not my own have attribution.

Our URL:

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



Keep in touch!
heyrick -at- merseymail -dot- com

Making bread...

Those who have read *Frobnicate* in the past will know that something I enjoy doing on weekends is to walk around *vide greniers*. This translates literally to mean "empty attics", but in essence it is a car-boot sale.

As I have said before, you can see all sorts of useless garbage, but if you look really carefully you may be able to find a little gem. This was one such gem. A "generic" (unbranded) bread maker for *five* euros (about £3!).

After giving the bread maker the once-over, I plugged it in and set about making my very first loaf. This is where the first problem was encountered. The recipe has been translated *literally* from American – because only the Americans are peculiar enough to measure liquids using a dry measure (water g grams) and dry stuff using a liquid measure (flour g cups). You can see this in the recipe copied below:

QUELQUES RECETTES POUR VOUS AIDER

MODE NORMAL

PAIN BLANC

INGREDIENTS	POUR 750 GR / 1 ½ lb
EAU	11 OZ = env. 330 gr
FARINE BLANCHE	3 CUPS
BEURRE	2 cuillères à café
LAIT EN POUDRE	1 ½ cuillère à café
SUCRE	2 cuillères à café
SEL	1 ½ cuillère à thé
LEVURE deshydratée	2 ½ cuillères à thé
OU	
LEVURE du boulanger	2 cuillères à thé

After some research, I worked out that I should need about 420 grams of flour. So I mixed together all of the ingredients.

This is where I encountered the second problem.

Upon checking the mix as it was starting its 'rising' phase, it was a watery slush surrounded by raw flour. By trial and error I discovered that the mix works a lot better if you put the bread maker into 'dough' mode and as it finishes, switch the bread maker to 'normal' mode and let it run its normal cycle. Though, to be honest, it helps to give the mix a whisk around with a spoon half-way through!

This peculiarity arises because it seems that the machine behaves slightly differently in each kneading cycle. "No problem", I think, just open it up and observe its behaviour to locate the sticky relay. A bit of sewing machine oil, and all will be fine.

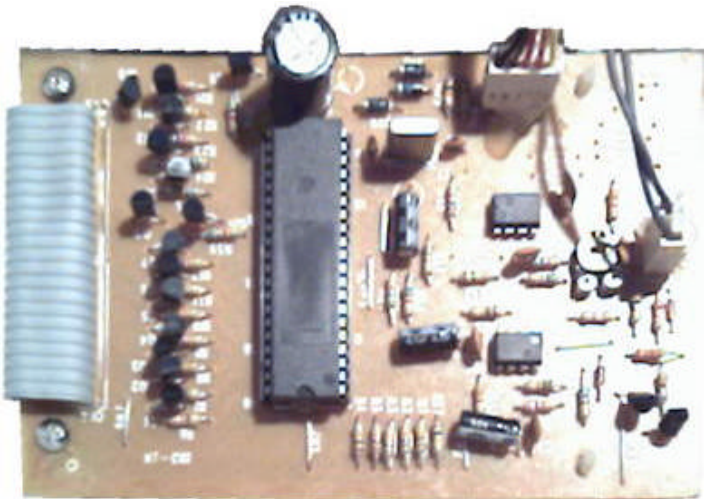
Well, that was the *theory*. Upon looking inside, it appears that there is only the one relay – for switching the heating element on and off. The motor is controlled by a triac, or something... solid state in any case. Tracing back, we must either have a loose connection (perhaps a damaged track on the PBC? though there is no evidence of overheating or physical shock) or we have some form of weird behaviour inherent in the machine's control software. To be totally honest, those are just off-the-top-of-my-head guesses.

I don't plan to look too deeply into this issue because the double kneading seems to be working okay. The only real problem that I am having is that the bread is coming out quite heavy and with a texture not unlike a toasting muffin. The instructions tell me that baking problems can often be corrected by adding or subtracting the water or flour spoonful by spoonful. So I guess this is just going to take a bit of trial and error and experimentation!

If you have a bread maker with a recipe for plain bread that is expressed in better measurements, please can you email it to me? Thank you!

(refer also to page 18...)

Now, you *are* reading *Frobnicate*, so it is only fair that we now take a deeper look. For this, of course, we shall turn to our trusty screwdrivers and disrobe the machine with gusto and aplomb! Yay! The fun part!



The first thing to note is that the control board is rather simple in appearance. This is because microprocessing devices are extremely cost-effective compared to state-machines created from logic gates or diode arrays, and the order of flexibility available simply doesn't allow sane comparisons to be made.

On the left, the interface ribbon to the user controls, LEDs and push-buttons. Beside, a line of transistors. At a guess, these are likely to switch the indicator LEDs – doubling as buffers so the microcontroller doesn't suffer too much current drain on its outputs.

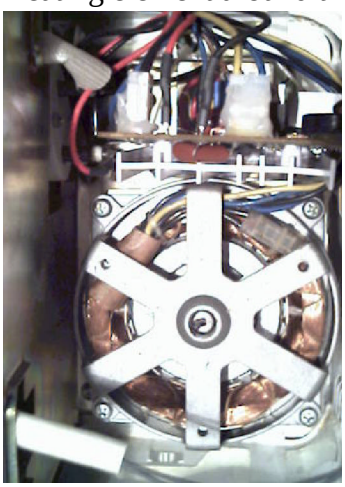
The crystal (top right of the microcontroller) clocks 6MHz. Much of the right hand side of the board is concerned with interfacing logic. Two LM gates and a lot of resistors and capacitors. The variable resistor is right beside the input from the temperature sensor, so logic would suggest it is for calibration.

The microcontroller itself is an Intel *87C51*, which is one of the vast collection of devices based upon the *8051* series. It is likely to feature something in the order of 4096 bytes of PROM (no window, so most likely OTP'd or mask programmed) along with 1024 bytes of RAM and a further 256 bytes of miscellaneous storage space (akin to a 'stack'). This may seem quite paltry in comparison to today's high-flying 32 and 64 bit processors, but the raw reality is that these simpler eight bit and 16 bit devices are selling more per annum by several orders of magnitude. Heck, even the MP3 player I talked about in the last issue is a fairly simple processing unit bolted to a powerful 24bit DSP.

The vast home computer market is stagnant compared to every microwave, washing machine, TV, remote, burglar alarm, and loads of other "embedded" things!

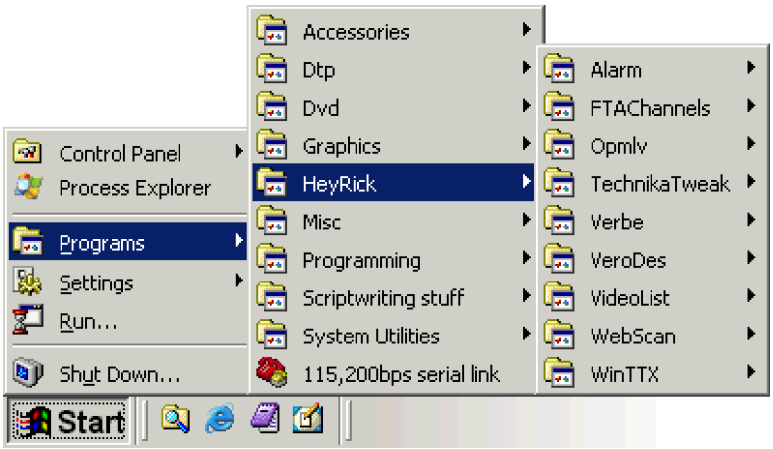
This picture shows the inside of the bread maker. A non-stick 'pot' with a rotor. A mixing 'blade' is

supposed to be attached but it was out soaking as I took this picture. If you look carefully, you'll see the heating element around the bottom of the pot.



Here is another view of the control stuff on the right. The thermal insulated wire at the bottom of the picture leads to the temperature sensor. The thermal insulated wire at the top of the picture leads to a cut-out fuse in case of overheating. The control board separates the microcontroller and user interface components from the high voltages switched to the motor and the heater. The control board rests about where I held the camera to take this picture.

For me, the most interesting part was the simplicity of the mechanics of the bread maker. This is because processors allow all of the intelligence to be expressed in 'code' form. While this device will never be able to replace a professional baker, it certainly brings the ability to enjoy fresh home-baked bread to those that found all this kneading and rising to be a flour-coated hassle, or (here's my excuse) those with old ovens that seem '*pathologically*' incapable of baking the same way twice. Once I have the recipe cracked, I can see myself making a lot of bread to put my own jams onto! Furthermore, as it is heating in an enclosed insulted area, it is a lot more cost-effective than heating an entire oven for one loaf! Rick 2006/04/15



HeyRick software

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

While I have written a number of programs for RISC OS, there is a growing collection available for systems running Microsoft's *Windows* (98 or later). Some of this software is free to download and use, others are demo versions to which you must purchase the full version.

Alarm FREE

http://www.heyrick.co.uk/software/alarm/

One thing that surprised me was that Windows offered all the Internet stuff "out of the box", but it didn't have an alarm clock. So I set about implementing something not unlike *!Alarm*, and this is it! It is small, simple, and it reminded me to watch *Surface* every week!

FTAChannels FREE

http://www.heyrick.co.uk/ricksworld/digibox/

This provides a simple way to browse the free-to-air channels available with *SkyDigital*, either by channel number or channel name. The software is updatable by importing a new CSV file; and you could – if you wanted – include your own channels, if you have a FreeSat card or the like.

OPMLV FREE

http://www.heyrick.co.uk/software/opmlv/

If you download the *OvationPro* maillist files, you'll notice that they are not suitable for importing into mailing software, and anyway very little (Windows) email software would even consider something as obvious as importing an "rmail" format file. Finally, earlier files have a different concept of "end of line" than do later files. So, the obvious solution was a custom viewer for the mailing list files. Messages are available by month and then by subject. You can 'find' within each month's list of messages. And, of course, it'll look for message files either direct from the *OvationPro* CD-ROM or from downloaded files.

TechnikaTweak FREE

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

This allows a simple point'n'click interface to the settings of a Technika ML-2 MP3 player, as sold by Tesco last winter (and may work with compatible devices).

Verbe No demo available

http://www.heyrick.co.uk/software/verbe/

This is a commercial program (*cost €10, plus €2 P&P within the EU*) which will help you conjugate French verbs. Includes automatic conjugation of around 125 verbs, built-in English-French lookup, lessons, and the ability to test yourself. *Email me if you are interested.*

VeroDes FREE - *COMING SOON*

This is a veroboard (stripboard) design program. Early versions should be available sometime in the summer; related to the *Amélie* project.

VideoList 2 DEMO - *COMING SOON*

VideoList is exactly as its name suggests. A way to keep track of your video collection. Also has support for DVD media, along with things such as DivX...

Expected price, *around €25.* A demo will be available.

WebScan DEMO

http://www.heyrick.co.uk/software/webscan/

If you make your own website, you may appreciate something to tell you what has changed "since last time". That something is *WebScan.* The registered version (*€15 on CD-ROM*) is faster, nicer, better; but the demo gives a pretty good idea of what the software can do.

WinTTX FREE

http://www.heyrick.co.uk/software/winttx/

A fully-featured teletext viewer and hardware interface for PCs. Requires a Ground Control/Octopus teletext receiver. *Note – due to direct hardware access, works on 95/98/98SE/ME* ***only.*** ***Does not work with Windows XP!***

A complete list of software available for Windows/DOS may be found at:

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

Embedded systems

I briefly mentioned embedded systems in the previous article, about the bread maker. Time, now, to cover this in a little bit more detail...

Many children with "electronic inclinations" have, at one time or another, built themselves a burglar alarm. A few logic gates. When the looped chain around the doors and windows is 'broken', this triggers a timed circuit connected to a little buzzer. If the alarm is not deactivated within the time-out, it'll switch on a Darlington transistor which is connected to a big-ass *loud* siren. It'll remain in that state until reset.
A few logic gates, a few transistors, and a 555 timer will do all of that. However, if we put the matter into the hands of a microprocessor, it will easily double (possibly triple or quadruple) the build cost, and it will require a more complex board layout – not to mention software design issues.

But it is worth it.

Suddenly you can have an alarm that is aware of 'zones'. This can be as simple or complex as your programming skills. It isn't out of the question to make each zone behave differently depending on the time of day and, as in the case of weekends, dependant upon the day itself. The alarm system can also be instructed to "monitor" the current state of the alarm triggers. Instead of blindly firing off the alarm upon the circuit being 'broken', it can work upon the idea of a 'change' in alarm circuit state. You can easily program multiple systems (a "normally open" circuit next to a "normally closed" circuit), along with special inputs for smoke detectors that have outputs built into them. Perhaps instead of a keyswitch you can have a push-button pad, like a telephone, with a PIN to control the alarm. The PIN can be the length you desire (personally I find protecting your bank account with *four* digits to be ludicrous, especially when nobody freaks out at expecting people to remember 11 digit telephone numbers). Furthermore, you can make behaviour different depending on who exactly is controlling the alarm. Perhaps after 8pm your kids are allowed to open their bedroom windows but only you are allowed to go through the front or back doors when you come home...?

In a future issue you will be able to read about my *Amélie* project. This is an embedded system based around a 6502 processor, with 2K SRAM and a 4K EPROM. I choose the 6502 because of its extremely low interrupt latency, and because it is pretty simple to work with (the 8088/8086, for example, requires a 1:3 phase clock, yikes!).

In actuality, the 6502 was chosen as much for those reasons as for familiarity. The pre-ARM computers built by Acorn were based upon the 6502. Exactly what processor you choose is as much dependent upon its facilities as on how well you 'know' the processor.
If we keep with the CPU+RAM+ROM design methodology, you could as easily employ the 8088 (yuck!) or the Z80 or the 6800 series...
Switching to a *microcontroller* (basically a CPU with the RAM and ROM built-in), this can greatly reduce design costs and complexity for a reduction in I/O and firmware facilities. A 6502 can, with a bank of 6522s and some logic, provide *easily* 128 digital inputs/outputs, 32 analogue inputs, and a 32K program to oversee everything, storing data into 1Mb of banked SRAM. A *PIC*, on the other hand, cannot ever do this. But for more down-to-earth systems (such as our burglar alarm or, indeed, a bread maker), one of the PIC family could well be a good choice.

The final option is a half-way house. It comes in the form of the *vast* 8051 family. Many versions of the 8051 offer small amounts of built-in ROM (some have glass windows so can be erased and reprogrammed like a conventional EPROM) along with small amounts of built-in SRAM. *But*, with suitable logic on the I/O buses, they can suddenly become conventional address and memory buses. Sort of the best of both worlds.

I am, for the time being, sticking with the 6502, and you can read about this in a future issue, when I look to designing an open-spec embedded system.

But I've not ruled out the 8051 family, nor PICs. :-)

What I shall look at now is initialising a 6052 based system. With modern computers there are all sorts of complicated rituals to perform. The average Pentium starts off thinking it is an 8086-class processor. As the operating system starts, it must set up the page tables and enter protected mode. For the ARM in a RiscPC, the process is much the same, but instead of entering protected mode, you must tell it to use the 26 bit addressing mode, providing handlers for the 32 bit modes that are 'new' (I think the FPE uses these?).

Thankfully our simple 8 bit processor is nothing like this complex. We have no modes, we have no protection rings (i.e. "user" and "system"). Since we are going to be working as an embedded system, we can keep stuff nice and simple.

The 6502 requires three built-in vectors to be defined. These are:

```
&FFFA   nmi_vector
&FFFC   reset_vector
&FFFE   irq_vector
```

Coupled with the page zero and the hardware stack (in page one), this necessitates that the EPROM will be at the *end* of the memory map, and the SRAM at the *start*. I/O typically falls into the middle someplace.

Here is example code for *reset_vector*:

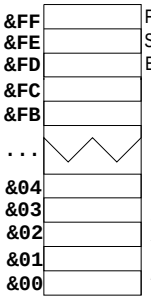
```
.reset_vector
    SEI
    CLD

    LDX     #&FF
    TXS

    CLI
```

Then the system code can follow, or you can `JMP` to where the system code is located.

Let's work through it...
The "`SEI`" sets the interrupt disable bit, so nothing can cause an interrupt while we are resetting.

This is followed by "`CLD`" which clears the 'decimal' maths mode of the 6502. This isn't used a lot, but we should always clear it as the state of the flags on processor initialisation is pretty much "undefined" on the 6052. Things were tidied up on the later 65C02.

The "`LDX #&FF`" pushes the value &FF (or 255) into the processor's 'X' register. The following "`TXS`" transfers the contents of the 'X' register into the 'Stack pointer'. As the 6502's stack works *downwards* in memory, we've just set up the stack.



As you can see from this diagram, the first byte of data is placed into the 6502's stack at the highest address – `&FF`. Because the stack lives in page one, the *actual* address is `&01FF`. The stack pointer is then adjusted to point to the next *free* location (`&FE`), which is where the second byte would go, and so on... A 256 byte stack may seem small to you, but recall that "assembling code on the stack" (an evil in itself) is not something that was performed with this processor; hence the tiny 256 bytes would permit something like 48 levels of nested interrupt! Entire operating systems (like Acorn's MOS) found 256 bytes to be plenty. I'm sure it won't be a hardship to us!

At this stage, it would be prudent to initialise the other (I/O) hardware. I have omitted this stage as it would obfuscate matters, not to mention being highly dependent upon the hardware that we have!

Another idea could be to write `&FF` to the end of each 1024 byte block in memory. Then write `&00` to address `&0000` (to clear registers and data bus). Try to read back the value previously written, and when it is no longer `&FF`, we'll be able to tell how much SRAM is present. You may also wish to clear the memory to zeros at this point.

Our final command is "`CLI`", which means "CLear Interrupt disable bit". It is a bit of inverse logic which means "switch interrupts back on".

For a most basic initialisation of a 6502, that's it! That is *all* that is actually needed to initialise the processor. Everything else, mainly, is concerned with setting up the attached hardware.
Next time, a simple look at interrupts and interrupt handling...

Rick, 2006/04/16

:) Go figure! (:

a ramble around Rick's mind...

The vide grenier, St. George's Day 2006

As I write this, I am sitting at a table in a manoir at a local town… A friend's laptop rests beside me, running a slideshow application I cobbled together last night! I am selling a copy of my "Verbe" software which helps teach French verbs ("quel[le] horreur!"). This is a vide grenier and exposition for St. George's Day, so it is well attended by the local English community. So far I've sold three copies (at 10€ each). But – amazingly – the main comments I seem to be getting are "it's too hard for me" (my easy-to-use software?) or those who live here all the time and simply don't get why speaking *French*... ***in France***... might perhaps possibly be useful, if not required! One of them even said to my mother (who is beside me selling balls of wool) *"oh, I can't be bothered"* (learning French). I could say something, but it'd be rude and people would send me emails of complaint. I mean, seriously, what *can* you say.

Tell you what, put a French girl in England, give her the "why should I speak English" attitude and then see what the English say. I'll say no more about this as regular readers will know that I have a whole heap of disdain for *certain* English people that live over here. I just hope *Jacques-Moyenne* doesn't meet them first, he'd have no interest in knowing anybody English if he did! (*maybe that'll be Sarko's excuse*?)

Anyway...

I'm quite pleased with the setup. My laptop is running the registration key software (plus *OvationPro* so I can write this!). It is connected to an Epson colour printer. There is no connection between the computers – this laptop doesn't have Ethernet (*if anybody has an old PCMCIA card they no longer want?*) so I am moving software between the computers by using my MP3 stick as a kind of floppy disc. My laptop rearranges the display to fill the screen (CRT *or* LCD) so the full screen size is always in use. Unfortunately my friend's laptop isn't that smart. If I drop to 800×600 (the size of the slideshow, as it is the size I habitually use on the CRT), it appears as a little space in the middle of his LCD. Not a problem – tweak the software in order to read the screen dimensions (erm... erm... like I *should* have done in the first place!) and then set the picture holder to stretch the picture to the size of the screen. The aspect ratio is the same, so it scales up reasonably well. Create the `.exe`, drop it on the MP3 stick, pull it off on the other computer... lovely. Okay, VNC would have really rocked, but this is not bad. Now let's just hope a few more punters come along, like my spiel, and cough up the very reasonable 10€ I'm asking.

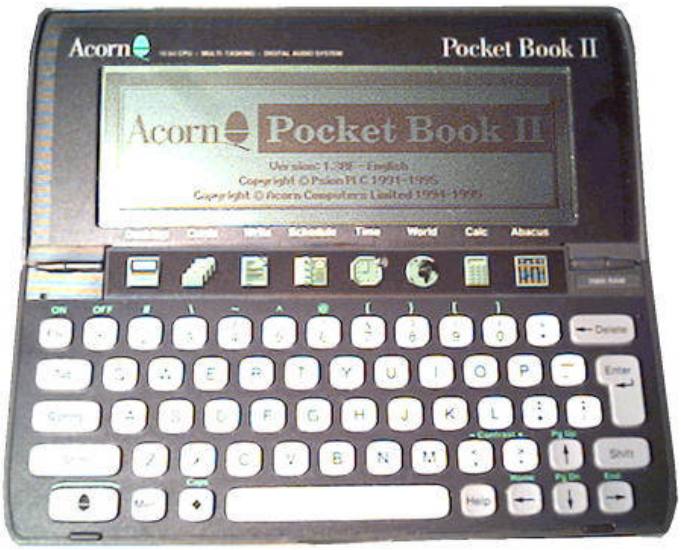
An Acorn PocketBook II (Psion 3a)

Another friend (*hi John*) gave me an 'old' *Acorn PocketBook II*. This is, more of less, a rebadged Psion 3a series organiser. This model has 256K built in (other models were available with 512K). You can plug in two expansion cards, which may have been mask-programmed ROMs for purchased applications, or SRAM cards which you can use as additional storage. I'd imagine this'd be a fairly common upgrade as the built-in "Desktop" takes an obscene amount of memory to get itself going (79K, which is a lot on a 256Mb machine). The flexible memory system provides an internal "C:" drive within the memory also used by the applications.

To be honest, I am pretty impressed by this little machine. It provides a fairly rudimentary word processor, a flexible database application, an agenda with plenty of options (including a "to do" list and "anniversaries"), a powerful spreadsheet/database, and as if that isn't enough, you can even write your own little bits and pieces using the built-in OPL which is quite similar to BASIC).

Now get this. If you're as old as me (or older), you'll remember the IBM XT range, probably running some version of *GEM*. The amazing thing is that this little organiser is just the same, only about the same sort of size as a "Walkman". The processor is an NEC V30H clocked at 7.684MHz. This is, essentially, an improved CMOS version of the 8086 with the ability

to be clock-stopped. This means that you can hit `Acorn-1` to switch the organiser off, and then you can press one of the task bar buttons to go right back to *exactly* where you left off. No reboot, no auto-resume load-up. Bang, back on! The operating system is "SIBO" (evidently: *SI*xteen *B*it *O*perating system).
The only limitation that I have that I have at the moment (asides from memory) is that I have no way to connect the machine to a 'real' computer for safeguarding the data. The *3a* series communicates by a six-pin connection (the expansion port is a third 'card' interface) and logic inside the expansion connector translates the proprietary signals into standard parallel (for a printer) or serial (for computers or modems) interfacing. I hope I'll be able to get hold of a serial interface (can *you* help!?).



I had reason to open up the organiser. The right-most two task bar buttons were erratic, and one day the entire task bar stopped working. As a side effect, pressing the 'T' or 'L' keys caused the organiser to freak out and spew about six or so seemingly random characters to the current screen. Turns out that the ribbon connector is not an especially tight fit. A little piece of paper inserted behind the ribbon cable should help to hold it more firmly.
Inside, there are four main parts – a ROM containing the system firmware, two RAM chips (128K each), and a whacking great ASIC. This contains the V30 (8086-like) processor core, plus all of the bits and pieces that Psion have added. The good news, if soldering SMCs doesn't faze you, it looks as if upgrading your organiser may not involve anything more than soldering the new memory ICs in place.

One of the best features for people like myself is the ability to write custom software for the organiser. I have not had much of a play with the language, but one of the first things that I did was to implement a registration program for my software, as shown above. In this respect, my work with VisualBasic came in quite useful as OPL does not have the ability to perform binary shifting – it seems BBC BASIC was rather advanced in this respect. Furthermore, for some reason known only to Psion, OPL provides `AND`, `NOT`, and `OR` but not `EOR`/`XOR`! But knowing that it was an 8086-class processor, I was able to poke three bytes into memory (`xorl ax, bx; retf`) to then call that whenever I needed something `OR`'d exclusively!
OPL gives easy access to numerous features of the grahical user interface. Bordered rectangles, dialog*ue* boxes, menus, etc etc.

At one time, it was planned that "every schoolchild in Great Britain will have one". I don't have any figures for how many children did eventually have a PocketBook II to use. I think the problem is that technology moves so quickly – the average modern pocket organiser (or PDA, as they are more commonly known) has as much processing power as my main computer, iRDA, WiFi, MSIE (or compatible) built it, can play DivX and the like... But tomorrow? Already I'm seeing multi-Gb MPEG4/DivX player devices smaller than a cigarette packet. I guess one of the primary constraints will be the need to get information into the machine. Many modern PDAs have done away with the keyboard – which represents a large area of the Psion 3a/PocketBook II. Either you can tap little characters on the screen, or it will attempt handwriting analysis. My other PDA (re. Frob issue 20, p28) implements both methods, but sadly you cannot "train" it. It recognises *American* handwriting. I specify the word American in italics as some of the characters are formed very differently to standard English teaching so writing in stuff can be a rather alien experience.

If I had to pass comment on the Pocketbook II / 3a, then I'm afraid I'd be lying to you if I didn't point out two things. The first is the lack of a backlit display. Sometimes (at any contrast level) the screen is not entirely clear. I think they've done a clever thing by sandwiching two layers of liquid crystal to achieve "grey" and "black". Unfortunately this lowers the overall reflectivity as there is more for the light to have to pass through. And, as always, the "contrast" isn't. The contrast control makes all pixels appear stronger or weaker, so at the extremes you can have a blank display or a display with all pixels set – as opposed to true contrast which makes the dark pixels darker and leaves the light ones alone. This is not specifically a criticism of the Psion, all LCDs of this type exhibit the same behaviour. But it can be hard to read. Try reading the button prompts in a room lit by a tungsten bulb!

The second criticism *is* more of an issue. It *eats* batteries. John says the ones he used ran for over a term (approx. 4 hours use a week) on two Duracells. At the moment, I seem to be getting around 40-60 hours *in total* using NiMH rechargeables. Reports on the Internet suggest around 80 hours actual use on proper alcaline cells, or 20-30 with rechargeable cells.

Now we must come to talk about the stability of the software. There *are* a few little things that you can do to trigger a 'crash'. For example, you can cause the Desktop manager to crash by doing the following:

- if you aren't in the Desktop, press the `Desktop` button.
- press `Tab`
- find an *empty* directory (perhaps *WVE* or *OPO*?)
- hold the `Acorn` key (or `Psion` key on true Psion organisers), then press `Left Arrow`. This is a shortcut for "go to first file".

There is no "first" file, so the next thing you'll see is:

```
SYS$SHLL.$05

Process exited
Exit number 130
```

When you press `Esc` to continue, the Desktop will have restarted itself.

On the whole the organiser is *extremely* stable. Heck, my mobile phone crashes more frequently when I'm playing the built-in mini-golf game!

I found, on-line, a file describing the built-in spreadsheet. So I downloaded and printed it. I'm numerically dyslexic so I can't say I understood the purpose of many of the functions! I know one thing for certain – I'm only scratching the surface of what this little machine can do!

Home from the vide grenier

I'm back home now. I sold five copies of my software, which means I made €20 profit. Not big, but it is more or less the first public outing of the software. I think I may well have done better in making contacts than I did with selling the program. But, it is a start. Get the ball rolling, y'know?

Stephanie-Jane Murray

To my surprise the association where I go for French lessons has expressed an interest in it. Their *raison d'être* is in countering illiteracy and helping to re-integrate (French) people who have had problems, be it depression, drugs, alcohol, etc. As a spin-off, it has attracted a number of people from other places (Turkey, the Maghreb, lots of Les Anglais) who find French to be a difficulty. The person in charge of our local 'branch' has taken a copy to show the overall director. She feels that the software can equally be used to aid these people as well as the target audience (i.e. the English). So I guess this weekend I'll convert the software to use the "MessageTrans" system that I developed (based largely upon how the DeskLib "Msgs" thingy worked) so that it can be translated into French and easily swapped between French and English. There is a benefit for English people too, as it can sometimes be useful to switch a teaching program to be all-in-French for a deeper immersion. Sadly, for Turkish, they're on their own. One of the Turkish girls did try to teach me to say "hello", but it was a bit of a mouthful. What I remember know is something like "go-gotcha-kan", which knowing my luck is Turkish for "S-bend" (as in toilet)!

Stupidity comes someplace to happen

We've all had it. Some sort of electronic cock-up. Your bank paid out money on a cancelled card or direct debit? You're told you've given your birthday wrongly when you haven't. Your payment is rejected because something else got stuck thanks to a checkout bod incorrectly using the card swipe. You are asked to return a book you took back to the library last week.

I've experienced *all* of that. If I thought about it, I think I'd have experienced more but I don't want to bore you. Anyway, the point is that "*you*" *are wrong*. You are obviously mistaken. Or an impostor. Either way, *the computer is right and you are wrong*. End of story.

If you bang on, demand to speak to the manager, that sort of nonsense, people tend to start paying attention to you. If I was a fake me (imagine!), then would I be willing to "take it to the top"?

This highlights a very important piece of social engineering that we should bend over backwards to stamp out. People are all too quick to blame the computer when things don't work correctly. I have, in a recent issue, explained the bogosity of saying "the computer is down" and why anybody that trots out that excuse should be shot. So why, when a computer coughs up information, are people ***so*** willing to believe it? They don't believe it wholeheartedly, else I'd have written a two-line BASIC program to inform the girl of my choice that I'm a great person, marry me... :-) But in many things (especially those things that would require the operator to exercise a modicum of independence from the corporate machinations), they'll just stare glassy-eyed at the monitor and tell you that your name may well by Britney Greta Hauer, but you're actually a middle-aged black Jazz man from Louisiana. If it says it on the screen, it *must* be true.

Which leads us to Identity Cards. On the whole I do not have a problem with ID cards. I know there are some people (perhaps a certain Acornite reading this? :-)) that would rather eat animal excrement and die horribly than consider owning an ID card; but it doesn't bother me. Because I plan to be extremely careful who I give my card to. You want it for booking me into a hotel room? You can stick that right up your... You want it for an internal domestic flight? You gotta be joking!

Sadly, I can see this ending in disaster, with companies asking you to insert your ID card instead of taking all that time to fill in stupid little forms.

Imagine: you walk into a hotel. Press a button. Insert your ID card, count to five, then insert your credit card, count to five again. You've just booked a nice room with all the mod-cons and it took about twenty seconds. Furthermore they didn't need to employ anybody to actually talk to you.

That's when the disaster will begin. People were blasé with their credit cards and they got ripped. Luckily the credit card companies *had* to accept some liability for it. If people are equally blasé with their identity cards... well, what do you have to lose? Your personality? Who you are? Oh no, it is much more insidious than that. *Especially* when people believe whatever it says on the screen.

Or, if you are not convinced that the concept of ID cards is doomed to fail, ask yourself this. They say ID cards are needed because they are more secure than passports. If the criminals of the world can trick *experienced* customs and immigration officials with a faked passport, then what hope have we for a fake ID card given to an *in*experienced person who is inclined to believe what it says on the screen because free thought is a dangerous thing?

Yeah, I know, they'll reassure you that it the system will be safe and secure...

Rather like those 1,023 foreign criminals that the Home Office, erm, "*lost*" (including three murderers and nine rapists, if you're dumb enough to believe the other 1,011 were only banged up for "*possessing* skin colour *likely to be that of a terrorist*").

Rather like the "victims" of miscarriages of justice that are supposedly not to receive any compensation in order that they can afford to pay "real victims" more. As if being arrested, tried, and then jailed for something you didn't do is only a minor inconvenience in anybody's life...

Rather like a certain MP's insistance that the NHS is doing "quite well thank you" despite hospital closures and job losses at all points of the compass.

Here's the truth. It's *really* simple:

They're lying to you.

(*the country went to the dogs years ago, now even the dogs want out*)

Scanner troubles

If you went to a McDonald's last summer, then the side of your drinks cup may have featured an Asian girl that appeared to enjoy getting herself soaked in numerous different ways. I hope in this picture she is out in the rain, or perhaps standing in a shower (mmm, kinky!)... so long as it isn't an obscure advertisement for the *C∗ca-C∗la* product that is written elsewhere on the cup... because... well... that'd be *really* sticky! :-)

Scanned from a Mcsofdrink cup! :-)

Anyway, if you look closely you'll see lines in the picture (vertical, as I scanned it sideways). The lower picture is a close-up. Sometimes it is seems to be worse than others, but it is usually present to some degree. The scanner itself is a *Plustek OpticPro 4831P* (parallel port) model. The problem shows up using the original drivers under Windows 95 on the PC co-processor, the latest drivers under Windows 98SE on a 'real' PC, and with David Pilling's drivers under RISC OS. Hence, I would imagine this is either a fault endemic to the interface logic or some form of optical misalignment (though it appears that the optics are screwed in position and then cemented firmly in place).

Can you help? I'm none too keen to play around with the optical mechanism if this fault lies somewhere else. Besides, if the scanner is inputing at 200dpi (I usually scan at 200dpi to trade off image size (bytes) against quality) then why is there a gap of several millimetres between each 'problem'. I'd have thought that misaligned optics would have simply caused a "registration" error, sort of the opposite to that which you sometimes see in newspapers where the black and the colour have not printed in exactly the same places. The picture at the bottom of a previous column shows this with more colours. Everything appears to be lined up, except for the little hiccup lines. The girl was scanned from an advert for a service provider called ***1and1*** (`http://www.1and1.fr/`). Cute, isn't she? :-)

Anyway, if these pictures give you a hint as to what is going wrong with this scanner, please get in touch!

Scanned from a printed advert for 1and1.fr

Where do we go from here?

You'll perhaps have noticed that there is less geeky stuff in this issue. Well, that's because I have an off-line copy of my website available. Shall I write about bogus myths of programming? Nope, sorry, did that in issue 20 (p33). Let's rip open a CD-ROM drive. Oh, wait... did that in issue 16 (p11). How about an article on the corruption of our minds by insidious sneaky advertisers. Uh-uh, my mother wrote that for issue 9 (p32) a *decade* ago! If anything, the situation is worse, not better. Why do people slag off BASIC? Because they only see the bad implementations? Issue 22 (p21). There are reviews, explanations, various contributed articles... but is this what *you* **want** to read?

If it isn't, if you think this magazine is boring... ***you have no right to complain!!!*** There's a little PHP script running on the website (basically to track file formats) so I know when you download an issue of the magazine. Out of the hundreds and hundreds of people that download *Frobnicate*, I only seem to get feedback from five to ten people. *The **same** five to ten people.* So I will take *their* suggestions on board. For all of the rest of you... if you can download *Frobnicate* then you can email me. Heck, if that's too much like hassle then feel free to dispense with *all* of the formalities. Just send a message such as:

`Dude, why don't you write about` *whatever*`.`

It can be *that* simple (but, you know, keep it legal!). So, I'll look forward to reading your suggestions, okay? *Hint-hint*?

Rick, 2006/04/26

Virus 101

Introduction

One of the more challenging programming projects is that of a computer "virus". It has to:

- Insert itself into existing code – paying attention not to 'infect' the same piece of code multiple times!
- It (usually) has to try to hide its presence somehow; a simple method may be EOR decoding using a key randomly chosen at 'infection' time, so the small decoding code is the only part of the virus that doesn't change each time (though it is an obvious 'weak' point).
- It has to hook into various system events in order that it know to copy itself into new code. The most common method is to hook into the "application starting" event, though it could be possible to hook into "directory opened" and then do a background scan for files typed as applications (RISC OS) or ".EXE" (DOS). This could allow it to 'infect' software that isn't even running.
- Some viruses look for the virus scanner starting, and attempt to apply in-memory patching to render the virus detector unable to detect it. The jury is out on whether or not this is cheating!
- Obviously, all manner of methods need to be exercised in order to render the virus invisible to the human operator. Under RISC OS, a common method is to create an unkillable module with an innocuous name, or you could use the system calls to insert a section of code into the RMA, and then hook it to the desired vectors. This, then, would behave like a module without appearing in the module list.
  Under plain DOS, it is possible to create a TSR (Terminate and Stay Resident) module which behaves like a RISC OS "module". You may have been used to TSRs in the past – hit `Ctrl-Alt-F11` and a clock will pop up in the upper right corner of the screen, that sort of thing.
  Under Windows, it is actually a magnitude *harder* to write the virus code (perhaps as a device driver or some sort of .DLL), but it is a corresponding magnitude *easier* to hide it. All manner of rubbish is loaded when Windows starts. The `\Windows\System` directory is packed out with resources, and a snapshot taken with *Dr. Watson* shows hundreds of bits and pieces loaded. Any one of them *could* be our virus. How are we to know without already knowing what we are trying to find?

All in all, the creation of computer viruses is an interesting topic. It is almost like "the game of life" gone big time. Or, perhaps like "core wars" out of the sand box. Some of these coders are very intelligent and capable people, it is therefore very nearly a tragedy that most viruses are malicious in nature. There're so many *good* things they could do.

The biggest weak spot (a small diversion)

Every computer has a serious weak spot, and that weak spot is the master table of entries on the harddisc. Scramble this, and there is a good chance that it will be impossible to successfully recover *any* of the information that was stored on the disc. Okay, the data will all still be there – but the operating system will have no way of knowing how to find it. For example, imagine trying to locate the position of the file `C:\Windows\System\PtrIIC.dll`. We go to the root to find where 'Windows' is. Then, in 'Windows', we must look for 'System'. Finally, in 'System' we can look up which sector holds 'PtrIIC.dll'. Now *PtrIIC.dll* is small. Imagine if it was a big document with lots of images (an issue of *Frobnicate*, perhaps?). Then, there is a good chance that the file is *fragmented*. In layman's terms, imagine dropping a plate on the floor. The plate will smash. It has become fragmented. The operating system, when you access the file, knows where all the pieces are, and it transparently super-glues it together for you. You will only ever see "the file", but in reality it could be twenty or more itty-bitty bits. The Acorn `E` and `F` format discs get fragmented, but they include an intelligent automatic system to minimise fragmentation when possible. That is sometimes why your disc may trash for a few seconds when you open a file – it might be defragmenting it. Those used to any version of Microsoft's `FAT` formats will be used to

fragmentation. That's when your computer, over a period of time, gets slower and slower. Sadly commercial utilities to fix things up are neither cheap nor particularly fast – it can take several hours to reasonably defragment a 6Gb harddisc (and that is considered 'small' these days). Microsoft's own *Defrag* is worse than useless. One unexpected disc access (which Windows can perform frequently) and it'll restart from the beginning. As for speed? When I first got hold of this laptop I set it to defrag the harddisc. 2.5Gb used, the rest free (ha! those were the days!). I gave up when I came back nearly 12 hours later and it had only done a third.

Anyway, the point is that all of this location information is useless if the initial place to look is corrupt or missing. It'd be like trying to find your way around Paris with half of a map of London...

So what are we going to find out?

One of the main questions that people ask is "how does a computer virus infect?". This, I shall answer. But please note that we are talking about a true *code* virus. The ones that come by email rely upon duping people into opening an attachment, which is actually a program that is then executed quite legitimately by the mail software. It's just a program that does bad things. There is nothing clever about it, it works more by "social engineering" than code engineering.

Running a program

Both RISC OS and Windows operating systems are capable of running a program that begins with code. But in this day and age, that sort of thing is *very* unusual. Most programs actually begin with a "header" laid out in a particular format.

Under RISC OS, this header is part of the *APCS* which is a specification governing all sorts of things such as how parameters are passed between functions and, of course, how a program begins.
Here is an example:

```
00008000 : E1A00000 : NOP            ; Decompress call
00008004 : E1A00000 : NOP            ; Self reloc. call
00008008 : E1A00000 : NOP            ; Zero init. call
0000800C : EB00001B : BL   &00008080 ; Main entry call
00008010 : EF000011 : SWI  "OS_Exit" ; Exit instruction
00008014 : 000000A8 : Read-only area size
00008018 : 00000000 : Read-write area size
0000801C : 00000000 : Debug area size
00008020 : 00000000 : Zero initialisation size
00008024 : 00000000 : Debug type
00008028 : 00008000 : Current base of absolute
0000802C : 00000000 : Workspace required
00008030 : 00000020 : Addressing mode and flags
00008034 : 00000000 : Data base address when linked
00008038 : 00000000 : Reserved header word (should be 0)
0000803C : 00000000 : Reserved header word (should be 0)
00008040 : E1A00000 : NOP            ; Debug init. instr.
```

This is actually a "hybrid" header containing both data and executable code. The first five instructions form a series of branches to perform specific functions (in this case, the decompression code call is a *NOP* as the code is not decompressed). The "OS_Exit" call is a safety net in case the program does not, *itself*, exit correctly.

The MS-DOS header is a true header, and it is laid out as follows:

```
00    "MZ" – .EXE file signature (from Mark Zbikowski?)
02    Length of image mod 512
04    Size of file in 512 byte pages
06    Number of relocation items following header
08    Size of header in 16 byte paragraphs, used
      to locate the beginning of the load module
0A    Min number of paragraphs needed to run program
0C    Max number of paragraphs program would like
0E    Offset in load module of stack segment (in paras)
10    Initial SP value to be loaded
12    Negative checksum of program (used while EXEC
      loads the program)
14    Program entry point (i.e. initial IP value)
16    Offset in load module of the code segment
      (in paras)
18    Offset in .EXE file of first relocation item
1A    Overlay number (or 0 for root program)
```

There is one thing that each type has in common. There is either a branch-to or a pointer-to the first instruction of code in the program. And *this* is where a virus will want to hook itself.

Making a 'victim'

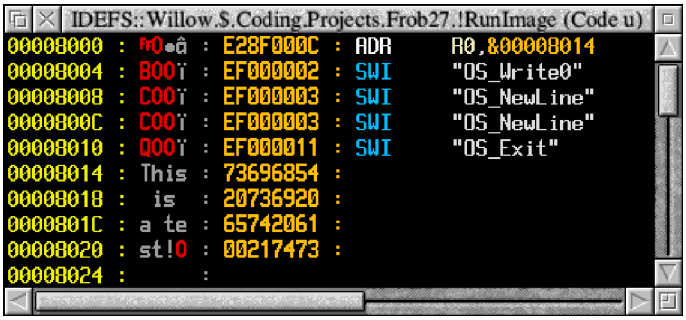
In order to demonstrate, we shall make a program that we will then 'infect'. It is pretty simple:

```
DIM code% 128
FOR x% = 0 TO 2 STEP 2
  P% = code%
  [
    OPT x%

    ADR   R0, msg
    SWI   "OS_Write0"
    SWI   "OS_NewLine"
    SWI   "OS_NewLine"

    SWI   "OS_Exit"

  .msg
    EQUS  "This is a test!" + CHR$(0)
  ]
NEXT
```

```
OSCLI("Save <Frob27$Dir>.!Test &"+STR$~ É
(code%)+" +&"+STR$~(P%-code%))
OSCLI("SetType <Frob27$Dir>.!Test Absol É
ute")
```

The two lines with the red arrows mean that the line is actually continuous, it is only split *here* because of the width of the text column.

Run this BASIC program, it will create an assembly language program that prints "This is a test!" on the screen. The program, internally, looks like this:



We now have a test subject. It is a *headerless* executable (roughly akin to a `.com` within DOS).

Infecting the program

We are going to implement a stand-alone BASIC program to show you the principles. If you were hoping for an assembler listing of a virus, forget it! I am intentionally *not* providing that.

The first thing to do is to load the program into memory and examine it to see if it is already 'infected'. We then check for APCS (if the *fifth* word is an "OS_Exit" instruction) and, if not, go for it!

```
file% = OPENIN("<Frob27$dir>.!Test")
size% = EXT#file%
CLOSE#file%

DIM code% (size% + 256)
OSCLI("Load <Frob27$Dir>.!Test "+STR$~code%)

IF code%!(size%-4) = &00002E65 THEN
  PRINT "Already 'infected'..." : END
ENDIF

apcs% = FALSE
IF (code%!16=&EF000011) THEN apcs% = TRUE

IF NOT apcs% THEN PROCinfect_null
END
```

To 'infect', we need to *replace* the first instruction in the program with a branch to *our* code. This is quite easy, as our code is appended to the end of the main code so we just branch to the end. We then append our code. After that, we must poke a branch to the original address into our code, and *finally* (!) we must poke in the original instruction that we replaced, modifying it if necessary.

Here's the infection routine:

```
DEFPROCinfect_null
  REM Read the first instruction
  opcode% = code%!0
  branch% = 0
  base% = &8000 : REM An assumption!

  REM And replace it with a branch to our code
  ours% = (((size% - branch%) / 4) - 2)
  ours% = ours% + &EA000000
  code%!branch% = ours%

  REM Now write out 'our code'
  PROCwrite_our_code

  REM Now we have to see if the opcode that we
  REM stored now needs to be patched.
  PROCpatch(opcode%, virus_patch)
  REM Write correct branch address
  writeaddr% = &FFFFFF - (((virus_branch - É
base%) - branch%) / 4)
  !((virus_branch - base%) + code%) = É
&EA000000 + writeaddr%

  REM Save modified program
  OSCLI("Save <Frob27$dir>.!RunTest "+ É
STR$~code%+" +"+STR$~end%)
  OSCLI("SetType <Frob27$dir>.!RunTest &FF8")
ENDPROC
```

From this, we must write "our code", as follows:

```
DEFPROCwrite_our_code
  FOR loop% = 4 TO 6 STEP 2
    P% = base% + size% : REM Exec address
    O% = code% + size% : REM Assembly address
    [
      OPT   loop%

      ADR   R0, virus_message
      SWI   "OS_Write0"
      SWI   "OS_NewLine"

    .virus_branch
      MOV   R0, R0

    .virus_patch
      MOV   R0, R0
      MOV   R0, R0
      MOV   R0, R0
      MOV   R0, R0

    .virus_offset
      EQUD  0
```

```
    .virus_reg
      EQUD  0

    .virus_message
      EQUS  "This is the attached code."
      EQUB  0
      EQUB  0
    ]
  NEXT
  end% = (P% - base%)
ENDPROC
```

Some explanation may be in order here...

This is what *normally* happens:

- The operating system loads the file (or part of the file, if demand-paged).
- The operating system passes control to the program, starting with the first instruction.
- The program now runs.

This is what *now* happens:

- The operating system loads the file (or part of the file, if demand-paged).
- The operating system passes control to the program, starting with the first instruction.
- Actually, as *we* are the first instruction, control passes to our code.
- Our code runs.
- We now execute the original first instruction (the one that we replaced).
- We branch back to the instruction *after* the one that we replaced, thus handing over to the original program.
- The program now runs.

The next conundrum... If we replace *one* instruction, why have we got space for *four*? The reason is because a relative LDR can only directly address 4096 bytes forward or backward from itself. So we may have to put in code to perform an absolute LDR instead.

Patching the instruction that we replaced

This brings us to the *patch* procedure. It is definitely the most complex of all, and getting hold of an ARM processor datasheet may prove useful in deciphering what is going on.

Ready?
Okay, let's do it!

```
DEFPROCpatch(opcode%, addr%)
  LOCAL out%, offs%, dest%, c1%, c2%, c3%

  IF ((opcode% >> 26) AND 3) = 1 THEN
    REM Instruction is a Single Register
    REM Data Transfer
    REM
    REM We'll work out where we are supposed
    REM to read from, and read it in directly.
    REM
    REM Will *ONLY* work for:
    REM   "LDR R<reg>, &<addr>"
    REM Ignores: Byte loads, STR, and
    REM register/indexed offsets.
    offs% = (opcode% AND &FFF)
    offs% = offs% + branch% - 8
    !((virus_offset - base%) + code%) = (bas É
e% + offs%)

    REM Now work out the destination register
    dest% = ((opcode% >> 12) AND 15)

    P% = virus_patch
    O% = ((virus_patch - base%) + code%)
    [
       OPT   6
       STR   R12, virus_reg     ; Preserve temp
       ADR   R12, virus_offset  ; Convert LDR
       LDR   dest%, [R12]       ; to absolute
       LDR   R12, virus_reg     ; Restore temp
    ]
  ENDIF

  IF ((opcode% >> 25) AND 7) = %101 THEN
    REM Instruction is a Branch
    REM Read branch type, and dest. offset.
    offs% = (opcode% AND &FF800000)
    dest% = ((((opcode% AND &007FFFFF) * 4)  É
+ 4) + branch%)

    REM Recalculate destination offset.
    writeaddr% = (&FFFFFF - (((virus_patch - É
base%) - dest%) / 4) AND &007FFFFF)

    REM Rewrite the branch instruction.
    !((virus_patch - base%) + code%) = offs% É
+ writeaddr%
  ENDIF

  c1% = (opcode% >> 26) AND  3
  c2% = (opcode% >> 21) AND 15
  c3% = (opcode% >> 16) AND 15

  IF ((c1%=0) AND (c2%=4) AND (c3%=15)) THEN
    REM Instruction is a positive ADR
    dest% = (opcode% AND &0000FFFF) + 8
    c1% = ((virus_patch - base%) - (dest% - 8))
    P% = virus_patch
    O% = ((virus_patch - base%) + code%)
    [
       OPT   6
       ; SUB 'cos we can assume all
       ; of the code is before us.
       SUB   R0, R15, #c1%
    ]
  ENDIF
ENDPROC
```

The 'infected' program

When run, this program patches the original and converts it to the following form:

```
00008000 : ...ê : EA000007 : B       &00008024
00008004 : ...ï : EF000002 : SWI     OS_Write0
00008008 : ...ï : EF000003 : SWI     OS_NewLine
0000800C : ...ï : EF000003 : SWI     OS_NewLine
00008010 : ...ï : EF000011 : SWI     OS_Exit
00008014 : This : 73696854 : Data
00008018 :  is  : 20736920 : Data
0000801C : a te : 65742061 : Data
00008020 : st!. : 00217473 : Data
00008024 :  .yâ : E28F0020 : ADR     R0, &0000804C
00008028 : ...ï : EF000002 : SWI     OS_Write0
0000802C : ...ï : EF000003 : SWI     OS_NewLine
00008030 : $.Oâ : E24F0024 : ADR     R0, &00008014
00008034 : .. á : E1A00000 : MOV     R0, R0
00008038 : .. á : E1A00000 : MOV     R0, R0
0000803C : .. á : E1A00000 : MOV     R0, R0
00008040 : ïê : EAFFFFEF : B       &00008004
00008044 : .... : 00000000 : ANDEQ   R0, R0, R0
00008048 : .... : 00000000 : ANDEQ   R0, R0, R0
0000804C : This : 73696854 : Data
00008050 :  is  : 20736920 : Data
00008054 : the  : 20656874 : Data
00008058 : atta : 61747461 : Data
0000805C : ched : 64656863 : Data
00008060 :  cod : 646F6320 : Data
00008064 : e... : 00002E65 : Data
```

When the *original* program is run, the following output will appear:

```
This is a test!
```

When we run the *infected* program, the following output will appear:

```
This is the attached code.
This is a test!
```

APCS?

This method should work with APCS code, though you may wish to alter the **B** to be a **BL**, and return can be pushing R14 into PC, as is the APCS way.

I specifically do not provide support here for APCS as we'll need to ensure that there are no side effects with things such as the decompression code.

Because *valid* APCS programs *always* start with one of the following, it may be easier to implement:

- **BL** *somewhere*
- **MOV** R0, R0 (a NOP instruction)
- **MOVNV** R0, R0 (***deprecated*** old NOP)

We won't have to worry about **ADR**s and **LDR**s and suchlike... Really, we *only* need to detect and handle the **BL** instruction. Anything else is a **NOP** and can be simply omitted.

Bugs?

I'm sure there are many. This is intended as an explanation into the mechanics of how a virus can attach to existing code and get itself executed *first*, and not as a paint-by-numbers listing of an example virus. For that reason, the example code is expressed in BASIC and not assembler – it makes it much easier to follow; and thus easier to express the concepts. As a side effect, I've not handled every possible instruction that you could come across. I'm sure experienced coders will find lots that can be fixed and optimised.

One final thing, please remember I'm numerically dyslexic, so before you email me to say "why the hell d'you do it like that!?!" – if it is to do with anything mathy, please remember this!

The *painfully obvious* end note

One of my primary beliefs is that the *best* way to fight something is to understand it, not to outlaw it and then pretend that it is no longer an issue.

For this reason, you must surely realise that this information is provided as an explanation to help you understand how certain aspects of viruses work, and *not* as a guide to writing your own.

But, like anything, it can be used for good as well as bad. A document on the construction of a nuclear weapon can be a fascinating look into the most destructive thing we humans have created and the reasons why it all came to be, or it can be "information likely to be of use to a terrorist".

Knowledge is power.
Let's be sure to use it wisely.

Rick, 2005/05/09

Picture from *Euskadi TeleBiste*, Winter 2004.

Bread Revisited

Since I wrote the article on the bread maker, almost a month ago (refer to page 3), I have had opportunity to experiment.

What really tipped me off was when I offered some bread to friends John and Irene who both commented on its lack of salt. The handbook warns against adding too much salt as it affects the action of the yeast. However, the yeast was possibly acting *too* much. I was not really able to tell that I was under-salting as I do not normally use salt in food that I cook (except pasta-water).

There was also a need to modify the amounts of some of the ingredients. So here, then, is a new recipe for you to try. It makes a fairly heavy 'damp' bread. It may not be ideal for slicing and toasting, but if you cut it while it is still fairly warm and cover it with butter (or a decent marg.), it is lovely, and would beautifully accompany a home-made soup.

It is essential to have electronic scales that can measure in metric *and* imperial.

***Step one*:** Put the pot onto the scales and tare to zero. Add **3** fluid ounces of **milk**.

***Step two*:** Add **water** at *room temperature* to take the scale to read **11½** fluid ounces.

***Step three*:** Add **vegetable oil** (preferably the type used for salads and *not* frying oil) to take the scale to a nice round **12** fluid ounces.

***Step four*:** Switch the scales to *metric*, and add **500** grams of **plain white flour**, on *top* of the water.

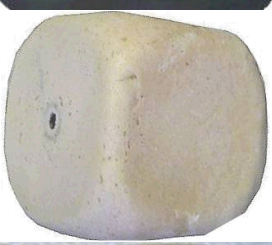
***Step five*:** Take a `heaped teaspoonful` of **salt** and sprinkle it around the edges of the pot, away from the centre of the heap of flour.

***Step six*:** Do the exact same thing again, only with `1½ teaspoonfuls` of **sugar**. Heaped, as before.

***Step seven*:** Make a hole in the centre of the flour heap using either your little finger, or the handle of the spoon. Add a `sachet` of **dried yeast** (about `5g`).

I am not sure if this isn't perhaps a little too much, however the pack (made by *Francine*) said to add a packet for mix to be used in a bread maker, so... **:-)**

***Step eight*:** Insert the pot back into the bread maker, and set it to the *normal* programme. I find the "Dark" baking control setting helps to give a better bake, but go for "light" if your particular bread maker is over-enthusiastic!



Exactly two hours and forty minutes later...

Ewen's multi–satellite setup

by Ewen Cathcart
(pictures by Ewen Cathcart)

In this article, Ewen describes his own implementation. It is hoped that seeing how it works in practice will provide you with some ideas of your own. However please note that neither the author nor Frobnicate can accept any liability for variances. This goes especially for non-UK readers who should bear in mind the necessary differences in dish sizes and – depending on locality – the availability of satellite reception and/or equipment.

–Ed.

This issue I will be providing details about my own multi–satellite setup. Hopefully you will be able to see how I managed to put into practice some of the advice given in the previous articles.

First of all let me tell you a bit about the satellite dishes that I use. The first picture shows my main dish. This is an 80cm solid type, approx. 14 years old (I am not sure of its exact age – I bought it second-hand some 12 years ago). The dish is attached to a washing line pole in the back garden. It is mounted as close to the ground as I can get away with in order to minimise the effects of pole "wobble". Currently pointing towards Astra 2/Eurobird at 28.2ºE, a second LNB arm provides reception of Astra 1 at 19.2ºE.

The dish shows signs of corrosion, with noticeable rust on the upper rim and around the screw heads. There is also some bubbling underneath the paint-work. Despite this, it is still in reasonable nick for something that is nearly a decade-and-a-half old! One advantage of the solid design is that the paint is given a fairly even coating throughout the surface. This provides an effective seal against the elements. Unfortunately, the same cannot be said for mesh dishes. The minute holes are like a magnet for particles of dirt. A dirty surface will retain moisture over a longer period of time. The holes are also where the coating is at its thinnest. Water finds a way in through tiny cracks, and corrosion becomes quickly established. The first signs of rust can often been observed on mesh dishes that have been out in the open for a couple of years. In some cases the process is accelerated further by exposure to salty sea air.

My mesh dish is showing only minimal signs of corrosion. In my case, the dish is at ground level and held in place by ridge tiles and a rock wedge. The mesh dish has the distinct advantage that wind can pass right through it – it has survived gales. The same cannot be said of my larger solid dish that would be displaced by anything more than a gentle zephyr. For weaker mountings, it may be worth considering a mesh dish... Perhaps Ewen's salty air has more of an effect than he realises? –Ed.

Other parts of my dish assembly have not fared too well. The LNB boom arm has become a rusty tube (Rick keeps telling me that it looks like a wooden broom handle!). It appears that the corrosion has worked its way from the inside out (the outside of the arm was painted, but the inside remained uncoated). It would probably take nothing more than a good karate chop to snap it in half. Having said that, it has still managed to outlive LNB arms of quite a few mesh dishes in this area. I have seen several dishes that have their arms missing completely!

Also note the rusty condition of the bracket on the back of the dish. Fortunately, it still retains enough strength to keep it secured to the pole. I am not too concerned about the dish taking flight during a gale, at least not just yet!

Now let's turn our attention to the second LNB arm. The arm is actually a short length of galvanized steel tubing (part of an old FM aerial). The tubing is attached to the washing line pole using a TV aerial mast clamp. Interesting to note that today's solid dishes also use galvanized steel for their LNB arms. Anyway, the LNB is mounted on an improvised holder. The arm had to be attached below the dish, hence the need to raise the holder up to the same level as the existing holder.

Locating the focal point of 19.2ºE wasn't too difficult. A simple adaptation of methods outlined in previous articles. One advantage of having an LNB on a separate arm is that you don't need to worry about accidentally knocking the dish and/or LNB out of place. There is one minor problem though. I found that tightening the clamp resulted in the arm moving up slightly. I was able to compensate for this by positioning the arm slightly lower than required.

Now, let me introduce a recent addition to my family of satellite dishes – the camping dish! This is actually a small 40cm plastic offset dish that is included with the (*Lidl*) Digital Camping Satellite Kit. The one I have was purchased second-hand. I got all the bits and carrying case, minus the *Porty II* receiver (which would've been a nice addition – it runs off both mains *and* 12/24V!). The dish itself is attached to a second washing line pole in the garden, and is currently pointing to Hotbird, 13ºE.

The plastic design (actually a thin reflective layer sandwiched between two moulded sheets of plastic) means that it should outlive similar metallic dishes. The LNB arm is also made of plastic, so no fear of anything dropping off here! The use of plastic as a material means that it is also very light, therefore ideal for mounting in situations where a heavy dish might prove too much of a strain on masonry. Also, the small area makes it much less of a wind-catcher.

The camping dish is quite easy to assemble. There is also a choice of mounting options. In this case I am using an adjustable clamp to secure it to the pole. Another option is to use a base containing 3 sucker feet (for use on a glass window pane or metallic skin of a caravan). There is also a more permanent fixture via a conventional wall mounting bracket.

The main problem with this dish is its small dimensions. I found that reception of Hotbird is not too good. I consider this installation to be a temporary measure until I can get something better sorted out. My original plan was to add another LNB to the second arm of the 80cm dish using a simple 2-LNB bracket. The design of the bracket allows it to be placed directly over the existing LNB and holder. Unfortunately, I found overall reception to be unsatisfactory, even under dry conditions. I am not sure why this is the case. Perhaps the degree of separation between the primary focal point and this focal point was too much? Perhaps there are some objects in the way towards the direction of 13ºE? Perhaps the efficiency of the dish has gone down due to the effects of corrosion? Of course, being at a northerly latitude doesn't help either! This is something that I will need to investigate further.

I also considered installing the indoor dish (one that I use for occasional reception) on to the second washing line pole. This is an 80cm solid type, very similar to the one that I have outside. Unfortunately, I found that this pole was far too wobbly to

guarantee a steady signal. I was also concerned that it would partly block signals travelling to my main dish.

The camping dish is hidden for much of the time under a black plastic bag. I was worried that it might become a target for some opportunist (quite easy to dismantle by hand). The plastic bag looks very much like a bag of rubbish! Only problem is that white cable tends not to blend in very well with green grass! [*It's a dead giant mouse! –Ed.*] Unfortunately, I do not have any spare lengths of black cable left. Given that the current install is a temporary solution, I have decided to leave things as they are. Otherwise I would have painted the cable a more suitable colour or replaced it altogether.

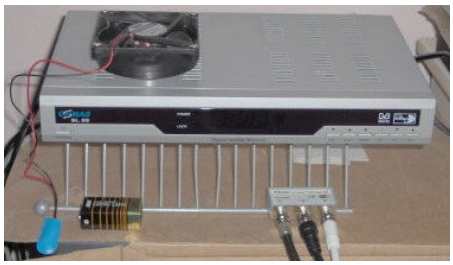
All three cables (two from the main dish and one from the camping dish) go into the house via an open window. Again, this is more out of convenience than anything else. The gaps in the window are stuffed with blankets to keep the heat in and cold out. I like to have a degree of ventilation in my room anyway.

It is also possible to buy thin flat connections designed to pass between a window and its frame. I have not had experience of these. Why? They tend to be expensive. –Ed.

Rick

The cable from the LNB at 28.2ºE is attached to my Digibox, while the cables from 19.2ºE and 13ºE are connected to my Comag SL55 digital satellite receiver via a 2-way DiSEqC Switch. Of course it is normal practice to install a DiSEqC switch on the outside, but then having multiple lengths of cable through an open window isn't exactly normal practice, is it? : -)

The Comag SL55 runs quite hot. This appears to be a common feature of early Comag-badged receivers. It is possible that there is some kind of design flaw in the PSU. Anyway, a good way to keep the temperature down is to provide adequate circulation underneath the unit. In this case I have placed the receiver on top of a metallic CD rack that has been turned on its side. I have also placed an old PC case fan (powered by a rechargeable 9V battery) on top of the receiver. It makes a pretty good job of sucking hot air out of the receiver and keeping it nice and cool. [*I use a 12V fan salvaged from an old computer PSU. It is powered by an old 9V Nokia mobile phone charger. –Ed.*]



Okay, I have told you about my current set-up. Now what about my future plans? Well, for a start, I would replace the old 80cm dish with the one that I use for indoor reception. The good thing about this dish is that the plate on the back is lower down; hence I should be able to raise the dish up a bit while keeping the bracket at roughly the same level as before. Prior to installation I will inspect the dish for scratches and chip marks in the paint-work (from various knocks it has had while in the house!) and paint over these using a clear lacquer "pen". The brackets will also be given a good coating of moisture retardant liquid [*D'you mean WD40?* :-) *–Ed.*] to ward off corrosion. The LNB boom arm is made of galvanized steel therefore we shouldn't need to worry about that one becoming rusty!

The existing LNB holder of the dish will be replaced with a multi-holder version. The aim is to provide

reception of all three satellite positions on one dish. The main focal point will be located over 19.2ºE. LNBs on either side will pick up the secondary focal points of 13ºE and 28.2ºE. Hopefully, reception of 28.2ºE will be satisfactory this time, given the small degree of separation between the adjacent focal points.

I will use the two existing LNBs for this new setup. The third LNB will be a twin-output type. This will be placed at the 28.2ºE focal point. One of the outputs from this LNB will go directly to the Digibox. The other output, the 13ºE cable and the 19.2ºE cable would go to a 4-way DiSEqC switch. The output from this switch goes to the Comag SL55. I think on this occasion I would prefer to install the switch outdoors (it would mean only two cables coming in to the house).

Pictured here, the 80cm dish currently used for indoor reception.

I ruled out use of a Monoblock LNB for reasons that were given in the previous article. This option may become available if/when I obtain a receiver that has support for DiSEqC 2.0.

Having 28.2ºE reception on both receivers will provide me with greater flexibility when it comes to viewing English language channels. No more programme clashes on satellite. I will be able to watch one Astra 2 channel while recording another. Or possibly record both while watching a third channel via analogue terrestrial means. All without the need to cough up £10/month (excluding subscription charges) for Sky Plus!

The old dish will be used for indoor reception. I might get rid of the rusty arm and replace it with the (current) second arm. The camping dish will continue to be used for outdoor "experimental" reception.

I hope this description of my satellite system will be of some use to you when you come to plan your own setup.

Ewen Cathcart

The Frobnicate Quiz

In issue 26, I presented a picture and asked "Who is this?". The correct answer... It is the Italian singer Laura Pausini.

The attribution is at the lower left of the picture. I will assume that this is where the picture came from... I used Google to find the picture!

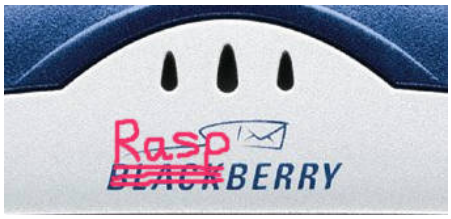
So here is the question for this issue:

Who is this?

Bonus points if you can say where the picture is from!

As always, answers in the next issue.

Mom's page...



The window is open and the little sparrows are frolicking out front, joined by soaring swifts. If anyone ever repoints our eastern wall it will wipe out a large portion of the local bird population. Radio 4 recently had its annual bird census, and reported that the number of sparrows is steadily declining. You know where they are? They are looking for holes in old stone walls that aren't there. How's that for a paradox? We know where the wall lizards are, they are in our cat's tummies...

Farmer neighbour has taken the first cut of hay from the field out front. Rain was forecast, we have never seen him move so quickly, *ever*. The old red tractor overheated and was abandoned down in the corner of the field. Finally the "roundballers" (this is truly the French name for those, well, big round hay bales) were finished, complete with plastic hair-nets on them, to keep them from unrolling across the field. And it was no more than six hours later that the rain came. The barley is nearly ready, it has turned golden yellow and ripe. Last year the entire wheat and maize harvest were declared a disaster in this area (no rain, no gain) and big intervention combine harvesters arrived to slice it all up and leave it laying on the fields like so much Chinese cabbage soup.

Rick and I went to the biennial "Fete d'Écriture" (Writing Festival) in Angers back in April. It is a fantastic get-together for people in the region who are not linguistically skilled with the French language. The theme this time was Seasonal Postcards. Everyone had to write and design a "postcard" which was then hung up in a big display of over 400 cards! And anyone who had participated was invited to the day-long fete. Over 200 people came, and each group was to bring food representative of their area. We had an incredible meal - French food, regional French food, Moroccan food, central African food...it was all so interesting. All of the people we met were very friendly, and when we left at the end of the day I said "Goodbye" to our new friends. They said "Not 'goodbye' but 'see you next time'!"

Has anyone else noticed that Spring and Autumn have practically disappeared? Last September, thanks to the tail end of a hurricane (there were so many last year, it was quite unbelievable, wasn't it?) our Summer ended very abruptly with wind and rain and rapidly plunging temperatures. Remember, our normal pattern is to remain warm until at least the end of October, followed by a bit of Indian Summer. We had a long, drawn out Winter which only ended a couple of weeks ago. One morning it was still only 3 degrees, then in the space of only three days the temperature was already in the mid 20's. Yesterday it was 31 degrees. Will it be a part of Global Warming that we will no longer have a transitional season? What will this do to Spring planting? What kinds of flowers and vegetables will we be growing? My peas, beans and carrots are all looking sorry for themselves...and it isn't even quite Midsummer yet! The only veggies doing well are the tomatoes, radishes and beetroots. Oh, and the Pumpkin Rouge Vif d'Etampes. By the time I read that it should be pinched back at four feet, it was already nine feet tall. It has completely outgrown it's metal twirly cane and is loping along the window ledge, all sixteen feet of it. In the interests of research, you understand, it has been left to grow on, to see how big it will become. Maybe it will climb right over the roof and down the other side, bearing bright orange pumpkinettes to contrast with the not-really-slate slates? Wouldn't *that* surprise the neighbours?

A couple of months ago I asked Rick what a "Raspberry" was. He said "It's a fruit, Mom". I said no, not that kind of Raspberry, this was a thing like a tiny, hand-held computer. Or something. Well, step back in astonishment: Rick had never heard of it. He'd never heard of Blackberries, either! Score one for a non-techno Mom!

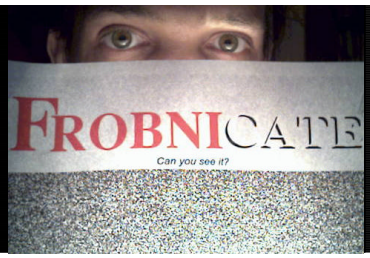
If any of you are thinking of moving to France, be aware that M Sarkozy, our "Sarko", is trying to amend the immigration laws. He will welcome you with open arms if you are a millionaire dentist with a PhD in astro physics. but we are no longer quite certain about the rest of us (even if we do know what Raspberries are...). More about this next time. The proposed law has just cleared the second legislative house (the Senate) this week, and I need to get a copy of it to comment further.

Sort of on the same topic, the Taxmen here are great! They smile and say Thank You when you arrive at the tax office on the last possible day to file, and hand them your return forms, looking deeply frazzled. The roads leading up to our regional tax office are all dug up – we strongly suspect that the Tour de France is going through town. It makes getting anywhere near the office a magical mystery tour (we were once put on a road diversion that was 40 miles, 40 *MILES* long!), but all was well.

This time, instead of a book, I am going to recommend music. No, not Bob Dylan (he has a radio show, imagine that). A few weeks back Charlie Gillett, on The World Service, played a tribute to Ali Farka Touré who had recently died. He was from Mali, in central Africa, and a musician of great talent. I like all of his music – think Africa meets New Orleans jazz – but especially a song called Diaraby. And, anyway, how many people have you met who can play a pumpkin guitar? Listen if you get a chance, it is quite unique.

Have a great summer! ;-)

The Wrap Party



Well, this is it for another issue. And, to be honest, I am currently thinking why Ewen can't dig a tiny trench in the lawn (the edge of a flat shovel, with a bit of wobble, will do it) to push his little dish's cable in to? If nothing else, burying it an inch or so down will mean it'd be safe to whizz over that patch with the *Flymo*!

The last issue brought in the idea of articles with a grid lock – this is so text in both columns lines up, even after pictures. It makes things look neater, but it can be a bit of a pain to set up.

Previous issues introduced us to text flowing around pictures. I try to do that once in a while to liven up the magazine. For sure it is quicker and easier to rectangle-frame it, but it's kinda lame in this day and age *not* to have some variety. I learned my lesson with the "huge" PDF, so I won't be using this effect quite as much as I'd like to – I have to take download times into consideration.

So what's new in *this* issue? Well, if you are using RISC OS then you may (or may not?) have noticed some images missing. That is because they are PNGs. However, the biggest single change is that the page numbering is inner/outer so if you assemble *Frobnicate* to be an A5 pamphlet, it'll look quite nice.

Don't worry! *OvationPro for Windows* can do all sorts of nifty new things, including support for Unicode. As much as I might like to make use of this, I cannot – my PDF creator can barely cope with the 'fi' and 'fl' ligatures. What d'you think it'd do with an omicron? It's simple – it'll do what any good Windows application does – crash! :-) It is especially a shame as there's a nice "smilie face" glyph.

Now you may recall that I did a bit of reminiscing in issue 24 (Spring 2005, p30). I now want to ask what happened to certain foodstuffs – we'll start with *Poppits* in a little purple box. Basically, chocolate covered raisins. Lovely! Another thing I used to enjoy was a *Walnut Whip*. Last seen in Reading railway station circa 1994. I recall when *Snickers* was *Marathon* and – let's face it – the American branding is awfully poncey, isn't it? *Marathon* is good, solid, makes you want to eat one and then get up and do something. *Snickers*, on the other hand... eat one and giggle your way through it.

Perhaps the thing I'm most offended about is the apparent demise of the lovely *King Size Mars Bar*. I think it is offensive and patronising of the nanny-state mentality to get rid of a big chocolate bar because a bunch of people have no self control and got fat on that sort of food product. Well excuse me! I happened to like it, and I wasn't bothered if I got fat or not. I found a big Mars to be a good thing to munch while watching TV. The pitiful-sized offerings that are on sale these days just don't have the same appeal, even if I eat two.

In the same line of thinking, there was a lovely range of ready meals in the late '80s and early '90s that was, essentially, stereotypical "truck stop greasy spoon" food. Scrambled egg, chips, sausage, and bacon. Just press out this part of the lid and put the whole lot in the oven for half an hour. Then take off the lid and give it a further ten minutes cooking. ***Lovely!***

And on that note, I must say that the Heinz baked bean pizza range was *highly* underrated. This is perhaps all the more crushing as the French have no idea about how to make beans. Sure, you can get haricot beans suspended in orange gloop, but frankly those penny-a-tin cheapie tins from the leading supermarkets offered more to interest the palet than the French idea of beans. Perhaps if I cook them slowly with half a bottle of HP sauce it might start to taste like beans should. That leads us to the next problem. Obtaining HP sauce, or anything that tastes like it.

Maybe next time I'll have a recipe for you!

Until then, best wishes!

Rick Murray
2006/06/19