FROBNICATE

A time to change...

Hissing Spinach
Autumn 2004 Issue 22 €0
123>

Index:

[#22, 2004/10/18 20:45 CET]

Page 2 . . . . Index
Page 3 . . . . Editor's Notes
Page 4 . . . . Why Lua is not BASIC by Gavin Wraith
Page 8r . . . The Frobnicate Quiz
Page 9 . . . . Life in France
Page 10 . . . . The spArchive
Page 11 . . . . Playing with I²C (part 3)
Page 15 . . . . Qu'est-ce que c'est, ça?
Page 17 . . . . Reader Survey results
Page 19 . . . . The RISC OS Wars (Whose licence is it anyway?)
Page 21 . . . . What made BASIC sucky?
Page 24 . . . . Diary of a hacker...
Page 25 . . . . A Matter of Language
Page 26 . . . . Go figure!
Page 30 . . . . But I'm glad I was there when it happened by David \Norris
Page 32 . . . . Installing a satellite dish by Ewen Cathcart
Page 35 . . . . Twin Peaks
Page 36 . . . . The wrap party...

The cover picture is of the European "Soho" satellite's view of the sun in ultraviolet. What you can see are three superimposed images to capture the behaviour and characteristics of the sun's surface.

This picture was screengrabbed from a book/CD-ROM pack called "BIG B@NG l'Espace".

Credits:

Designed, written, and created by Richard Murray.
"Why Lua is not BASIC" written by Gavin Wraith.
"But I'm glad I was there when it happened" written by David Norris.
"Installing a satellite dish" written by Ewen Cathcart.
Images that are not my own have attribution.

The Frobnicate website - where you can also find the previous 21 issues...

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

As this is the Halloween issue, allow me to present to you a pumpkin lit by a road flare. I'm sad to say that this isn't my idea. This picture, and lots of other mad pumpkin related things, can be found at http://www.extremepumpkins.com/ and you can also experience the bizarreness of 'carving' a pumpkin using 150 rounds from a semi-automatic.

From www.extremepumpkins.com

Keep in touch!

heyrick -at- bushinternet -dot- com

Editor’s notes

Time to turn from computing and look at the social situation in a much wider perspective...

There are *plenty* of reasons for people to despise each other, and if there aren’t any real reasons then it is fairly simple to invent them. It worked for Hitler, it works for every weirdo that blows something up as a representative symbol of hatred. And it works on a smaller scale between ordinary people. People *we* know.

You know, the strong people are not those who can cast aside insults.
The strong people are not those that can give as well as they take.

No, the strong people are those who are willing to offer the hand of friendship *even* when the odds are against them, those people that feel money is a “necessary evil” due to the way the world has been set up (and they think we’d probably be better off without it).
I aspire to be one of these people. Am I there yet? Possibly not. Will I ever be? I hope so.
And I hope the same for *you* too.
As Jewel Kilcher said in her song ‘Hands” †, *only kindness matters*. It is true. Money helps grease the wheels, sure, but you can be a very nice person and broke as easily as you can be an objectionable turd and rich.

I’m sure some of you are, by now, shaking your heads and thinking this is starting to read like some sort of Communist Manifesto. If so, this is pure coincidence. My “manifesto”, if you like, is a call for peace and cooperation and friendship. No political angle, no hidden agenda. This crap has gone on for long enough.

Will it fall on deaf ears? Probably. But like I aspire to become a better person, I will at least make an effort to hope that others can do likewise. At least I will know that I tried.

Now those of you trying to read between the lines may be getting a little bit confused. You might be thinking “what does he know about Castle and ROL that I don’t?”. Answer? Nothing insofar as I am aware. No, this is addressing a far bigger issue. Sure, I’ll pass comment on Castle and ROL in these very pages because it is of concern to our *community*, however the content of my “manifesto” (for want of a better word) extends far beyond RISC OS. Different computer platforms exist just as different coloured people exist, and different coloured people exist just like different computer platforms exist. While some things may seem “disadvantaged” (certain races seem prone to sickle cell anaemia, certain platforms seem prone to ‘viral’ attacks), it does not mean that those without such “disadvantages” are intrinsically “better”. No, it just means they have different “disadvantages”. Exactly what and how much depends mostly upon your yardstick.

Are we happy to have disillusionment, disgust, anger, con merchants, and losers show us the way and shape our futures? I’m not. I’m heartily sick of it, and I think it is about time something changed.

Only kindness matters.

Let’s make that mean something.

† - ‘Hands” is a nice song so long as you fade out the last thirty or so seconds. It appears that Jewel ‘found’ God and this influenced her music ... but then, what exactly should you expect from an album called ‘Spirit”?!?! :-)

Why Lua is not BASIC

by Gavin Wraith
http://www.wra1th.plus.com/

We are told that ARM chips outsell all other designs as a consequence of their dominance in the "embedded" market.
Lua, which is eleven years old this summer, is in some respects a software equivalent of the ARM. It is a small efficient library of routines written in ANSI C, designed for maximum portability generally, and for embedded devices in particular. You can read about its evolution from a data-entry language for PETROBRAS, the Brazilian state oil company, and about the principles which have guided that evolution, at `http://www.lua.org/history.html`. You can get an idea of the scope of its applications from `http://www.lua.org/uses.html`. There is also a book "Programming in Lua" which you can browse online at `http://www.lua.org/pil/`. For a general review of Lua see `http://www.unixreview.com/documents/s=2472/ur0405k/`.

Just as ARM chips are not only used in mobile telephones, digital backscratchers and inflatable washing machines, but also in our beloved RISC OS desktop computers, so Lua is not just a C library for use in snowmobile testing gear, robot assassins and distributed fantasy games, but also a very neat general purpose programming language. Yes, the suggestion which I am nudging over the table like a bowl of sugar in the direction of your lap is that Lua and RISC OS go together like strawberries and cream and you should give it a try.
You can get it from **`http://lua.riscos.org.uk/`**

Rick asked me to explain how Lua differs from BASIC.Now I am not a fan of "language wars" {*neither am I - Ed*}. Programming languages are tools and different jobs require different tools. You have to learn how to use them, which takes time (whatever the idiot claims of the titles of the paperbacks that clutter the shelves of so-called bookshops) and emotional investment. It is the need to justify that investment that is often the cause of "my language is better than yours" nonsense. Of course, some tools are well designed for their job and others are not. Judgements about such things can be based on aesthetic sense and experience, but ideally they should also be based on a knowledge of how programming languages have evolved, and why. Have a look at `http://merd.net/pixel/language-study/diagram.html` to see some of the genealogies of programming languages.

Unfortunately BASIC was born before many of the most important ideas in programming language design came to light. Milestones passed later include structured programming, strong typing, modularization and information hiding, the importance of referential transparency, polymorphism, object oriented methods and inheritance; all buzz phrases which I shall have to point to and then leave them to buzz.
So if BASIC is all you know you will be handicapped by an ignorance of some major issues. In particular, your notion of what is hard or easy to program will be distorted by the particularities of BASIC. The late Edsger Dijkstra (you can read about an obituary here: `http://www.salon.com/tech/feature/2003/07/09/dijkstra/index_np.html`) was notorious for claiming that a knowledge of BASIC was actually a liability for those wishing to learn programming skills.
To be fair, there are good reasons why in the early days microcomputers only came equipped with BASIC. Shortage of memory meant that sophistications like automatic memory management were out of the question. It was not until the days of the Archimedes 400 that such things were feasible. There was a great gulf between the researchers, with their expensive workstations, and the amateur with a humble microcomputer, that has since closed up.

Without going as far as Dijkstra, I would claim that if you have never done any programming at all, then learning Lua would be easier than learning BASIC. Its syntax has fewer wacky exceptions, (why "`INPUT x%`" rather than "`x% = INPUT`", or why "`c% =`

`GET`" rather than "`GET c%`"?) { *to be fair: fputc(byte, handle) versus fprintf(handle, string) in C. Ed* } there are fewer rules to remember and it is more expressive.

Lua is actually not just a programming language, or even just a kit for building programming languages. As I said at the start, it consists of a collection of C source files.
A lot of software is written in C. One of the reasons for that is that C lets you write programs in a modular fashion, in many source files, each of which can be compiled separately.
You can link together a group of compiled files, if they have no collective external references, so that all the internal references are resolved, creating what is called an "object library". To use it to link into your C programs you also need a "header" file declaring the functions and data in the library that are to be exportable. The Lua distribution consists of C source code to be compiled as object libraries, structured like this:

The Lua core – this contains code for a virtual processor, memory management, and a compiler for a simple programming language (the basic Lua language) into instructions for the virtual processor.

plus

The Lua API (Application Program Interface) – code for translating between C and Lua, so that Lua can be extended by new functions written in C.

plus

The Lua base library – C functions defining Lua's built in functions.

plus

The standard libraries (pick and mix to taste) –

io	file-handling
string	string-handling
table	tables are the fundamental datatype in Lua
math	math functions
os	access to operating system
coroutine	coroutines
debug	functions for debugging

plus

Example code for an interpreter, built out of the foregoing.

The Lua distribution is really a kit of parts, written to be as portable as possible. For example, if Lua was to be compiled for use on a washing machine you would leave out the *io* and *os* libraries, because washing machines do not need files or an operating system. Instead you would write some device driver libraries.
This portability arose because PETROBRAS, being state owned, was not swimming with money, and the early versions of Lua had to work on all sorts of antique equipment. It also made it a good candidate for porting to RISC OS (did someone make a disparaging comment in the back row there?).

{*ouch! Ed*}

You do not need to know anything about C to use Lua, once you have a Lua interpreter (RiscLua provides you with an interpreter as a relocatable module, just as BASIC does).

To get a taste of Lua as a language I advise a look at the online book. Lua the language was designed not to compete with C but to complement it. You would not choose Lua for system programming or for speed. It is one of those increasingly fashionable things, a "scripting language"; this means that it is good for quick little programs that can do complicated jobs with very little effort. Its "*string*" library makes it a whizz for problems involving matching and catching patterns, and for manipulating text. It is generally agreed that Lua is the fastest scripting language around.

Lua can be programmed to alter its own behaviour, using what are called "metaprogramming" features.
For example, the usual behaviour of Lua when you use a variable that has not been defined is to give the variable a special value called "nil", not to be confused with zero or the empty string.
If you would prefer Lua to raise an "undefined variable" error instead, then you can make it do that.
As another example, there is a built in function "`print`" in Lua, analogous to BASIC's "`PRINT`".

The following is perfectly legal:

```
    henry = print;              ; henry becomes print
    print = 3;                  ; print becomes '3'
    henry(print*print);         ; henry "prints" 3*3 (i.e. 9)
```

I put this horror in just to shock you, in case you suffered from a prejudice that "built in" names should necessarily be different in nature from any other sort of name. {*I'd have thought common sense would have choked and died on that sort of code! That it is even* *possible...?!? Ed*} If you do not like this behaviour you can "lock" chosen names so that attempts to change their meanings will raise an error.

The syntax of Lua resembles a mixture of C and Pascal. Its semantics on the other hand have more in common with Scheme, because it has lexical scoping and functions are first-class citizens. {*Gavin explains what this means a little further down. Ed*}

Scheme is a dialect of Lisp, one of the oldest programming languages, in which the fundamental datatype is the notion of a list. In Lua the fundamental datatype is a "table".
A table is like a list or array, but is more general because its indices do not have to be consecutive integers. They do not even have to be numbers; any non-nil value is permissible as an index. Tables with indices that are strings are sometimes called "associative arrays". Awk and Perl have these. If `t` is a table and `i` is an index, `t[i]` denotes the value of the table at index `i`. You can think of the table as a function whose arguments range over just the values of its indices (but you use square brackets rather than round ones to enclose an argument). Another way of thinking of a table is as a dictionary – the global variables in a Lua program are stored as a table. The Lua libraries are tables.
Tables, implemented using "hashing", have the advantage over arrays that they are extensible, and over lists that lookup speed is independent of the number of items in the table.

To illustrate the use of tables, let me remind you of a classic method of speeding up calculations, called "memoization". The idea is that an efficient program should never need to make the same calculation twice. Once you have calculated something, store the result and use that instead, the next time you need it.
Memoization can also be called the method of "dynamic lookup tables".
Here is a definition in Lua:

```
memoize = function (f)
     local t = {}
          return function (x)
          if not t[x] then t[x] = f(x) end
          return t[x]
          end
     end
```

In other words, memoization associates a table (`t`) with a function (`f`), and the memoized function looks up its values in the table. If the table does not have an answer for a particular value then you must use the original function to calculate it, and then augment the table with that value. Because `t` is a local variable it cannot be accessed except by the memoized function. This technique can only be used with functions that never return nil values. You can do memoization in BASIC, on a function by function basis, but you cannot define a generic memoization procedure as in Lua.

Memoization can give dramatic improvements in speed. The traditional, and computationally dumb, definition of the function giving the Fibonacci numbers is:

```
fib = function (n)
     assert( n>= 0)
     if n < 2 then return n
     else return fib(n-1) + fib(n-2) end
     end
```

This function is very inefficient – the time it takes varies exponentially with `n`. The function memoize(fib) is much better. Its time varies linearly with `n`.

In Lua tables can grow as you go. For example, in memoize we started with an empty table (`t = {}`) and added items (`t[x] = f(x)`) when necessary. By contrast, in BASIC and C arrays are tied down to a fixed number of components; inconvenient if you do not know in advance how many you are going to need. {*All too true in BASIC but you can realloc() pretty easily in C. Ed*}

Note that memoize is a function that transforms functions. A programming language is called "first order" if it has two classes of object, passive objects (numbers, strings, booleans, ...) and active objects, i.e. functions that transform passive objects to passive objects. BASIC is first order.

A language is "higher order" if functions can transform functions, so that there is no class division between active and passive. This is what people mean by saying that functions are first class citizens. Scheme and Lua are examples of higher order languages.

To illustrate all this here is an honest little programming task you might like to consider, with solutions in BASIC and in Lua. I do not claim that either program is optimal; I just bashed them out.

Here is the task:

> Employees incur expenses of various kinds. The receipts for these are listed in a file in the format:
>
> ```
> RECEIPT{ "John Le Baker", 15, "taxi" }
> RECEIPT{ "Mary Louise", 12, "lunch" }
> RECEIPT{ "Keith Ng", 100, "trombone" }
> ```
> ....
>
> Write a program that takes as input the name of the receipt file, and the kind of expense, and which prints out a list of the employees who incurred expenses of that kind, together with the total of each's receipts for them.

The programs are listed below, but here are some comments on them first. I have tested both, but there may be bugs I have not found.

Comparison of BASIC and Lua solutions:

BASIC	68 lines
Lua	15 lines

Both have been written in my normal style of formatting, with a blank line between procedure definitions, etc.

More to the point, the Lua program is perhaps easier to follow. Also the BASIC program has to assume some maximum for the number of employees involved, which the Lua program does not. Because analysing the command line is harder in BASIC than in Lua I have used simple INPUT statements instead. The BASIC program would have been even longer if it had obtained its input from the command line as the Lua program does. This comparison is a bit of a cheat because the receipt file uses a format that is actually Lua code.
The Lua program defines RECEIPT as a function and then simply executes the receipt file using the command `dofile(arg[1])`, thereby filling up the employee table.
On the other hand, there is no data format that would give a corresponding advantage to BASIC, though the format used here actually presents no disadvantages compared to any other.

The BASIC program:

```
    PRINT "Input filename"
    INPUT file$
    PRINT "Input type of expense"
    INPUT kind$
    PROCinit
    f% = OPENIN file$
    IF f% = 0 THEN
      ERROR 100,"Cannot open "+file$
    ENDIF
    WHILE NOT EOF#f%
       PROCread(GET$#f%,kind$)
    ENDWHILE
    CLOSE#f%
    PROCoutput
    END

    DEF PROCinit
      Max_employees% = 255
      eptr% = 0
      DIM Name$(Max_employees%), Expense(Max_emplo
yees%)
    ENDPROC
```

this is one line

```
    DEF PROCread(a$,kind$)
        LOCAL i%,name$,val,type$
        i% = INSTR(a$,CHR$(34))
        a$ = MID$(a$,i%+1)
        i% = INSTR(a$,CHR$(34))
        name$ = LEFT$(a$,i%-1)
        a$ = MID$(a$,i%+1)
        i% = INSTR(a$,CHR$(44))
        a$ = MID$(a$,i%+1)
        val = VAL(a$)
        i% = INSTR(a$,CHR$(34))
        a$ = MID$(a$,i%+1)
        i% = INSTR(a$,CHR$(34))
        type$ = LEFT$(a$,i%-1)
        IF type$ = kind$ THEN PROCinsert(name$,
va l)
    ENDPROC
```

this is one line

```
    DEF PROCinsert(name$,val)
        LOCAL e%
        e% = FNfind(name$)
        IF e% < eptr% THEN
          Expense(e%) += val
        ELSE
         IF eptr% > Max_employees% THEN
           ERROR 10 ,"Start again with a bigger M
ax_employees%"
         ENDIF
         Name$(eptr%) = name$
         Expense(eptr%) = val
         eptr% += 1
        ENDIF
    ENDPROC

    DEF FNfind(name$)
```

this is one line

```
    LOCAL i%
    i% = -1
    REPEAT
     i% += 1
    UNTIL Name$(i%) = name$ OR i% >= eptr%
    = i%

DEF PROCoutput
    LOCAL i%
    FOR i% = 0 TO eptr%-1
     PRINT Name$(i%),SPC(6),Expense(i%)
    NEXT
ENDPROC
```

The Lua program:

```
kind, employee = arg[2], {}
RECEIPT = function (expense)
     if expense[3] == kind then
      local name,amount = expense[1],expense[2]
      if employee[name] then
        employee[name] = employee[name] + amount
      else
        employee[name] = amount
      end -- if
     end -- if
    end -- function
dofile(arg[1])
for person,total in pairs(employee) do
   print(string.format("%s  %d",person,total))
end -- for
```

I hope this little taster will tempt readers to investigate Lua further.

Well, Gavin certainly picked a good example of how to make a 'normal' BASIC program become a tiny Lua program!

You'll find Lua at: **`http://lua.riscos.org.uk/`**

Gavin's own website can be found at:
`http://www.wra1th.plus.com/`

*It isn't terribly obvious in Corpus, but his URL is "wra**1**th" and not "wralth".*
If you would like to email Gavin, it is "gavin" at the same domain (without the "www" obviously!).

All that remains is for me to thank Gavin for introducing Lua to us.

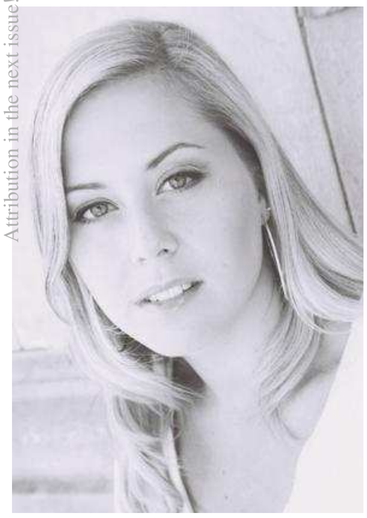
Give it a try, you might like it!

The Frobnicate Quiz

In the last issue I presented you with a picture and asked you to name the person.

The first correct answer (the only one, as it happened) was *Ewen Cathcart*. Well done Ewen!

Above is the original picture revealing... *Jennifer Love Hewitt* volunteering at the LA Mission. I asked Ewen how he was able to come up with so much detail in his answer (he told me exactly *when* too!) and he said he had heard of the Mission and he recognised the face but couldn't put a name to it – so it was simply a matter of finding celebrity photos taken at the Mission and narrowing it down until the person was identified!



So, then...
Who is this?

It's not so easy this time is it? :-)

Answers to:
heyrick -at-
bushinternet
-dot- com

Life in France

If you are English and have travelled in France last summer, did the Gendarmes pull you over?

Apparently when Mr. Sarkozy was in charge of traffi cky things (as Interior Minister), he decided the horrendous death tolls on the roads were due to the English drink-driving. You see, Les Anglais get their yayas from beer and spirits which are much more violence-inducing than your cheapo red plonk. This may even sort-of explain football hooliganism. It seems rather odd, given that the Germans are supposed to be the biggest beer drinkers of all (after all, what would Oktoberfest be without beer?) yet the only German hooligans you hear about are the shaven-head crazies. Maybe the Germans have more national pride than to be seen Pissed In Public.

Now, while I will not deny some drunk Brits smashing themselves up and giving the rest of us a bad name, I think you will fi nd the blame lies rather heavily with *the French drivers*.

Pick a girl. Any cute girl. [or a guy, if you're a girl...] How about *Nicole* from the "Papa...Nicole" series of Renault adverts. She's level headed, even spoke fairly convincing English. She's okay, that girl, right?
Well, put her behind the wheel of a car and see how long it takes for her to sprout horns!
There is *something* in the French psyche that requires everybody to be the fi rst in the queue if there is a queue of cars.
The French are *generally* conscientious drivers, if there is an obstacle you should put your hazard lights on and they *will* pay attention to that. They'll flash you to warn of traffi c police ahead, and they religiously switch on headlights at the proscribed lighting up time. They accelerate at traffi c lights when the car in front goes; unlike, say, the Italians who apparently go when the light turns green regardless of what the car in front is doing (!).

But traffi c queues bother them.
Normal queues don't. They'll sometimes try to barge in, but if a woman natters away for fi ve minutes to the checkout girl, it's often taken as an invitation for others to natter. Hell, even in McDonalds the whole point of "fast food" seems to be lost. *Don't worry, your burger will be ready in... Oh, I don't know – go find a table and talk...* But traffi c queues. They *must* be at the front of the queue.

There is a word in French (sorry, can't remember it!) that actually describes the act of overtaking more than one car. And this happens quite a lot. If there is a queue of eight cars behind a tractor, you just *know* somebody will come tearing up the other lane overtaking everybody, on a solid white line, and a blind bend. And once somebody does it and doesn't become an angel, others will follow.
It is a skillful 'game' to cut out in roadabouts and junctions using other (usually larger) cars as shields on the premise of "they'll get hit fi rst".
In some places there is a central overtaking lane for *both* sides of the road. I guess the logic behind that is akin to the game of "Chicken".
If fact a lot of driving is like a game of chicken. The English sometimes accuse the French of arrogance. I've not seen more or less arrogance than you'd get from a group of English people *except* when they're overtaking.

Locally, three years back: About six cars following a slow moving vehicle. Clear road, down a slight hill, fi elds around, great visibility. A woman in a little run-around hatchback car decides to go and *overtake the lot*.
How *exactly* did she manage to not see the gigantic bright yellow combine on the other side of the road? The newspaper report said she passed about three or four of the cars and just ploughed head-on into the harvester. Logic would have said to jam on the brakes (the harvester was doing exactly that!) and maybe to swerve somewhere? Off the road perhaps.
Arrogance would say "you're Rambo, a combine is no match!".
The harvester needed a new front mechanism. The driver of the car needed a coffi n. Go fi gure.

In August, the most amazing thing happens. The *Bison Futé*. All the southerners come north, all the northeners go south. And Parisiens apparently troop en masse to places like La Baulle. In the middle, around *Bouches du Rhône*, are some of the most amazing traffi c jams. Eight lane toll gates on the motorways, jam packed. Imagine being stuck in a motionless car for a day in 34° blazing sunshine (with no toilet or café nearby). They must be crazy! But every fi rst and last weekend in August, it is the same old story.

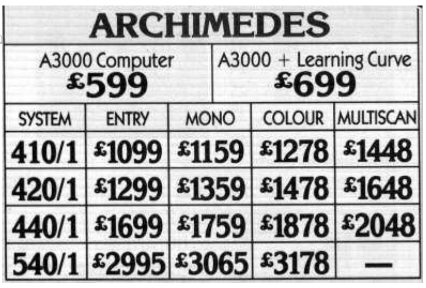
Next month...
I was fortunate enough to accompany a friend to Rennes train station (he was going to Paris to EasyJet back home) and we went on a pretty impressive Métro. I saw my fi rst TGV up close (think an Intercity but *no noise!*) and I'll talk about this in the next installment. *Yes, I know, it is more about trains.* That's because I'm not brave enough to try escargot!!!

Rick, 2004/08/01

The spArchive

The title of this segment is named after a mixture of Spark and Archive (as in PK, not PB). It is an in-joke to RISC OS users.

Anyway. The purpose is to look back to see what was. We'll start with an advertisement from *College Computers* which was published in *Acorn User* Autumn '91. That's on the left. On the right is a selection of PCs from *Galaxy Computers* (published in *PCPlus*, same time). The *Galaxy* prices are probably fairly standard. I chose their advert because the table form was similar to *College*'s – clear and easy to read.



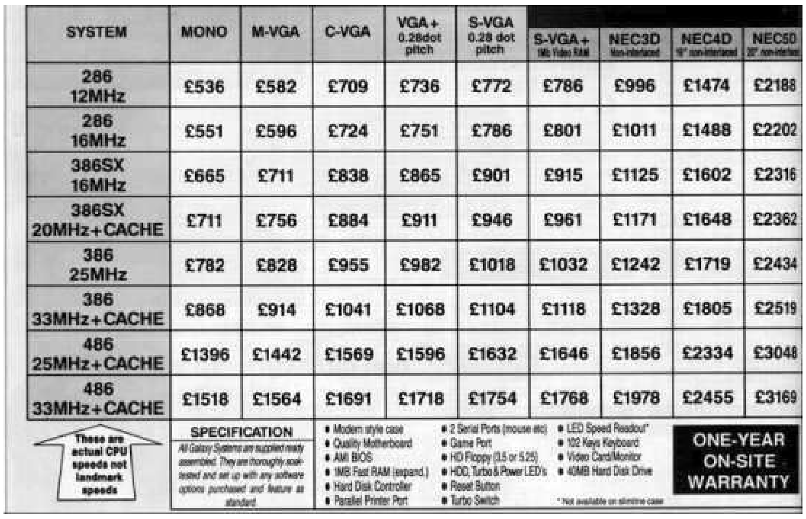
Scanned from *Acorn User* magazine.

ARCHIMEDES

A3000 Computer £599		A3000 + Learning Curve £699		
SYSTEM	ENTRY	MONO	COLOUR	MULTISCAN
410/1	£1099	£1159	£1278	£1448
420/1	£1299	£1359	£1478	£1648
440/1	£1699	£1759	£1878	£2048
540/1	£2995	£3065	£3178	—

College Computers

Galaxy Computers



Scanned from *PCPlus* magazine.

SYSTEM	MONO	M-VGA	C-VGA	VGA+ 0.28dot pitch	S-VGA 0.28 dot pitch	S-VGA+ 1Mb Video RAM	NEC3D Non-interlaced	NEC4D 16" non-interlaced	NEC5D 20" non-interlaced
286 12MHz	£536	£582	£709	£736	£772	£786	£996	£1474	£2188
286 16MHz	£551	£596	£724	£751	£786	£801	£1011	£1488	£2202
386SX 16MHz	£665	£711	£838	£865	£901	£915	£1125	£1602	£2316
386SX 20MHz+CACHE	£711	£756	£884	£911	£946	£961	£1171	£1648	£2362
386 25MHz	£782	£828	£955	£982	£1018	£1032	£1242	£1719	£2434
386 33MHz+CACHE	£868	£914	£1041	£1068	£1104	£1118	£1328	£1805	£2519
486 25MHz+CACHE	£1396	£1442	£1569	£1596	£1632	£1646	£1856	£2334	£3048
486 33MHz+CACHE	£1518	£1564	£1691	£1718	£1754	£1768	£1978	£2455	£3169

These are actual CPU speeds not landmark speeds

SPECIFICATION

All Galaxy Systems are supplied ready assembled. They are thoroughly soak-tested and set up with any software options purchased and feature as standard:

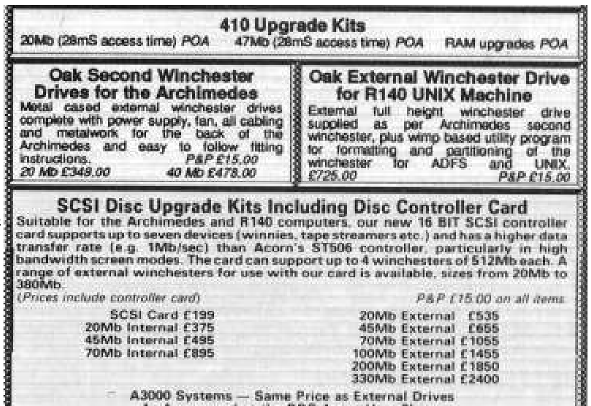
- Modern style case
- Quality Motherboard
- AMI BIOS
- 1MB Fast RAM (expand.)
- Hard Disk Controller
- Parallel Printer Port
- 2 Serial Ports (mouse etc)
- Game Port
- HD Floppy (3.5 or 5.25)
- HDD, Turbo & Power LED's
- Reset Button
- Turbo Switch
- LED Speed Readout*
- 102 Keys Keyboard
- Video Card/Monitor
- 40MB Hard Disk Drive

* Not available on slimline case

ONE-YEAR ON-SITE WARRANTY

It appears that the A3000 (8MHz ARM 2) is on a price par with a 286.
The top-of-the-range A540/1 (there is no price for a multiscan monitor, it is a 'free offer') is about the same as the higher-end 486. Now it is time to look at the smaller print. If I remember correctly, the A540 was supplied with 4Mb RAM and a harddisc around 100Mb. The PC? "1MB Fast RAM" and a 40Mb drive. Wouldn't a 486 machine be crippled with a mere megabyte? I think in these days, thirteen years ago (!) buying an Acorn may have been better value for money – and I'm not just saying that as an Acorn advocate. The Acorn A5000 was released at the beginning of 1992, and the A4 laptop followed shortly after, once they worked out fitting an A5000 circuit board into a PCB about 3½ × 11 inches (!). Guess what I am using *right now* to write this article? With WindowManager 3.98, the 26/32bit SharedCLibrary, and – believe it or not – *OvationPro* (there's practically no memory free). I won't be able to print from this machine, but I *can* write using it. It is still usable despite its 4Mb memory restriction, a 'slow' 24MHz ARM3 processor and a monochrome 640 × 480 display. I think *this* issue of *Frobnicate* should be taken as solid evidence that a computer that is over a decade old still finds application today. Several of these articles were written while I lay in bed. It is hard to be a lazy git when your desktop publishing platform is a two-slice RiscPC with 14" monitor!

My friend has almost a *terra*byte of storage (that's the next step up from a gigabyte). Take that, and even my pitiful 3Gb (combined) storage; I think that I shall leave this segment with part of an advertisement from *Oak Computers*, published in *BBC Acorn User* in October '89 (I was still at school then, having started fifth form some months previously). I like the A410 upgrades all *P*rice *O*n *A*pplication – in my experience, that tends to mean "worryingly expensive", then again a *grand* for less storage than I have inside this laptop? Yikes!
It is a sobering thought...



Scanned from *BBC Acorn User* magazine.

410 Upgrade Kits
20Mb (28mS access time) *POA* 47Mb (28mS access time) *POA* RAM upgrades *POA*

Oak Second Winchester Drives for the Archimedes
Metal cased external winchester drives complete with power supply, fan, all cabling and metalwork for the back of the Archimedes and easy to follow fitting instructions. *P&P £15.00*
20 Mb £349.00 *40 Mb £479.00*

Oak External Winchester Drive for R140 UNIX Machine
External full height winchester drive supplied as per Archimedes second winchester, plus wimp based utility program for formatting and partitioning of the winchester for ADFS and UNIX.
£725.00 *P&P £15.00*

SCSI Disc Upgrade Kits Including Disc Controller Card
Suitable for the Archimedes and R140 computers, our new 16 BIT SCSI controller card supports up to seven devices (winnies, tape streamers etc.) and has a higher data transfer rate (e.g. 1Mb/sec) than Acorn's ST506 controller, particularly in high bandwidth screen modes. The card can support up to 4 winchesters of 512Mb each. A range of external winchesters for use with our card is available, sizes from 20Mb to 380Mb.
(Prices include controller card) *P&P £15.00 on all items*

SCSI Card	£199	20Mb External	£535
20Mb Internal	£375	45Mb External	£655
45Mb Internal	£495	70Mb External	£1055
70Mb Internal	£895	100Mb External	£1455
		200Mb External	£1850
		330Mb External	£2400

A3000 Systems — Same Price as External Drives
As Announced at the BBC Acorn User Show.

Playing with I²C Part 3

In this, the *final* part of our series, we are going to take that grotty PC code and turn it into nice ARM assembler.

So without any further ado...

```
#define LPT_READ  0x379
#define LPT_WRITE 0x37A
```

This, as you can imagine, has no relevance on RISC OS. Actually, I think those locations are data used by the memory management system.
We can get perfectly acceptable results using the `Parallel_Op` SWI call. R0 = 2 to update the control register (our outputs) and R0 = 0 to read the status register (in R2). We *don't* need to bash the hardware directly – the SWI is plenty suffi cient.

The fi rst thing we need to realise is that a state should be held for 4.0s or 4.7µs (depending on what). The best way to handle this is a procedure to 'set' the control lines and provide a generic 5µs delay. Thus:

```
toggle_printer
        ; this allows us to time the bits'n'pieces
        Push     "R0-R2, R14"
        MOV      R0, #2
        MOV      R2, #0
        SWI      XParallel_Op

        ; The clock timings are (supposed to be):
        ;
        ;   Condition                     Time
        ;
        ;   Between a STOP and a START    >= 4.7µs
        ;   START hold time               >= 4.0µs
        ;   Data hold time                   0.0
        ;   STOP hold time                >= 4.7µs
        ;   LOW period of clock           >= 4.7µs
        ;   HIGH period of clock          >= 4.0µs
        ;   SDA/SCL rise time              < 1.0µs
        ;   SDA/SCL fall time              < 300ns
        ;
        ; [I2C bus specification, section 10.0
        ; (original) or section 15.0 (newer, 1992)]
        ;
        ; As we work in 'normal' mode, we determine
        ; the best, and easiest solution is to run
        ; run the time-out to 5µs. As it is a 2MHz
        ; ticker, this requires ten clock ticks.
        SWI      XOS_IntOff
        MOV      R2, #IOC              ; &03200000
        MOV      R0, #&FF
        STRB     R0, [R2, #Timer1WL]  ; &50
        STRB     R0, [R2, #Timer1WH]  ; &54
        STRB     R0, [R2, #Timer1GO]  ; &58

        ; Timer1 is a 16 bit down counter,
        ; clocked at 2MHz.

toggle_printer_delay
        STRB     R0, [R2, #Timer1LO] ; &5C
        LDRB     R0, [R2, #Timer1RL] ; &50
        CMP      R0, #&F5    ; 10 counts (5µS) less
        BGT      toggle_printer_delay
        SWI      XOS_IntOn

        Pull     "R0-R2, PC"
```

What is going on is...
1. We update the printer port. R1 is our port settings.
2. We load &FF into the low byte and high byte of IOC timer 1. It is a 16bit decrementing counter. Note that Timer0 is used by RISC OS itself and *very bad things* can happen if you mess with it. Timer1 is more of a free-for-all, though note that the following *cannot* be used alongside this code:

- Acorn's *TimeCode* module
- Niall Douglas' *Wimp2* module
- anything else that relies upon Timer1

Note also that this code resets Timer1 on entry and runs with IRQs disabled, so it will co-exist with other code performing likewise.
3. We write a value to the GO address. We write &FF for convenience as it is already loaded into R0, but any value will start the timer.
4. Now we'll write (any) value to the LOad address. This will cause the current count value to be latched.
5. We'll read the low byte back and compare it with a constant. Loop until the counter value is less than or equal to our constant.

Here are a few globally-defi ned constants:

```
         GBLA SCL_IN
SCL_IN   SETA 64              ; ¬Ack bit
         GBLA SCL_LOW
```

```
SCL_LOW  SETA 2               ; (¬AutoFeed bit)
         GBLA SCL_HIGH
SCL_HIGH SETA 0
         GBLA SDA_IN
SDA_IN   SETA 128             ; ¬Busy bit
         GBLA SDA_LOW
SDA_LOW  SETA 1               ; (¬Strobe bit)
         GBLA SDA_HIGH
SDA_HIGH SETA 0
```

Now we may begin converting C to assembler...

```
void iic_start(void)
{
   /* Sets up a start condition. */
   if ( ( inp(LPT_READ) & 0x40 ) == 0 )
   {
      /* Clock is low (error?); so set it high. */
      outp(LPT_WRITE, 2); /* SDA high, SCL low  */
      outp(LPT_WRITE, 0); /* SDA high, SCL high */
      /* It could be worth sampling SCL for about
         10µs to ensure we aren't mid-transfer. */
   }

   outp(LPT_WRITE, 1);    /* SDA low, SCL high  */
   outp(LPT_WRITE, 3);    /* SDA low, SCL low   */

   return;
}
```

This becomes...

```
iic_start
        PushLR

        MOV      R0, #0         ; get status reg
        SWI      XParallel_Op
        ; ignore errors
        TST      R2, #SCL_IN    ; SCL high?
        BEQ      iic_start_continue

        ; clock is low, so set it up
        ; nb: SCL floats high - why is it low?
        ;     we should check we aren't mid-transfer
        MOV      R1, #(SDA_HIGH + SCL_LOW)
        BL       toggle_printer

        MOV      R1, #(SDA_HIGH + SCL_HIGH)
        BL       toggle_printer

iic_start_continue
        MOV      R1, #(SDA_LOW + SCL_HIGH)
        ; transition of SDA (high->low), SCL high
        BL       toggle_printer

        MOV      R1, #(SDA_LOW + SCL_LOW)
        BL       toggle_printer

        PullRet
```

Here is the stop code, in C and then assembler:

```
void iic_stop(void)
{
   /* Send a stop condition. */
   outp(LPT_WRITE, 3);    /* SDA low, SCL low   */
   outp(LPT_WRITE, 1);    /* SDA low, SCL high  */
   outp(LPT_WRITE, 0);    /* SDA high, SCL high */

   return;
}
```

```
iic_stop
        PushLR

        MOV      R1, #(SDA_LOW + SCL_LOW)
        BL       toggle_printer

        MOV      R1, #(SDA_LOW + SCL_HIGH)
        BL       toggle_printer

        MOV      R1, #(SDA_HIGH + SCL_HIGH)
        ; transition of SDA (low->high), SCL high
        BL       toggle_printer

        PullRet
```

The read byte code is fairly lengthy, so I will not include a C version. Refer to Frobnicate issue #21 if you would like code to compare.

```
iic_readbyte
        ; Every byte on the I²C bus is 8 bits in
        ; length. The MSB is received first. After
        ; we have received all eight bits, we must
        ; send an Ack bit if there is more data to
        ; follow, but we don't Ack the last byte of
        ; a multi-byte transfer.
        ; [I2C bus specification, section 5.1, 5.2]
        ;
        ; As we are called:
        ;   R0 = Whether or not an Ack should be
        ;        sent for this byte
        ; When we leave:
        ;   R0 = Byte we've read
        ; or otherwise we'll raise an error.
        Push     "R1-R5, R14"

        ; store 'ack it?' value
        MOV      R5, R0

        MOV      R3, #0              ; byte = 0
        MOV      R4, #7              ; bit count

iic_readbyte_loop
        ; set up and clock in bit
        ; allows slave to pull SDA
        MOV      R1, #(SDA_HIGH + SCL_LOW)
        BL       toggle_printer
        ; clock it in
        MOV      R1, #(SDA_HIGH + SCL_HIGH)
        BL       toggle_printer

        ; read the bit
        MOV      R0, #0
        SWI      XParallel_Op
        ; mask out everything else
        AND      R2, R2, #SDA_IN
        MOV      R1, #1
        ; invert it and make it lsb (powerful op!)
        SUB      R2, R1, R2, ASR #7

        ; store the bit into the byte (remember,
        ; the msb comes *first*)
        ADD       R3, R2, R3, LSL #1
        ; byte = ((byte * 2) + (bit AND 1)

        ; clock it out now
        MOV      R1, #(SDA_HIGH + SCL_LOW)
        BL       toggle_printer

        ; more bits?
        SUBS     R4, R4, #1
        BCS      iic_readbyte_loop

        ; set byte to return with
        MOV      R0, R3

        ; do want we an ACK?
        CMP      R5, #1
        Pull     "R1-R5, PC", "NE"  ; no ACK, leave

        ; an ACK is wanted
        MOV      R1, #(SDA_LOW + SCL_LOW)
        BL       toggle_printer
        ; clock a zero bit down the bus
        MOV      R1, #(SDA_LOW + SCL_HIGH)
        BL       toggle_printer
        MOV      R1, #(SDA_LOW + SCL_LOW)
        BL       toggle_printer

        ; byte to return with...
        MOV      R0, R3
        Pull     "R1-R5, PC"
```

Now the write byte code...

```
iic_writebyte
        Push     "R0-R3, R14"

        MOV      R2, R0      ; Pop the byte into R2
        MOV      R1, #&80    ; 2^7 bit mask
iic_writebyte_loop
        ANDS     R0, R2, R1  ; zero if bit is zero
        MOVNE    R0, #1
        BL       iic_sendbit
        MOVS     R1, R1, LSR #1
        BNE      iic_writebyte_loop

        ; Do the fake ninth bit (for ack)
        MOV      R1, #(SDA_HIGH + SCL_LOW)
        BL       toggle_printer
        MOV      R1, #(SDA_HIGH + SCL_HIGH)
        BL       toggle_printer

        ; At this point, SCL should have been pulled
        ; LOW by the 'slave' if it is requesting
        ; a pause. We don't support this, and if we
        ; did - we'd need a timeout in case of some
        ; kind of hardware fault.

        ; Read the Ack state
        MOV      R0, #0
        SWI      XParallel_Op
        ; ignore errors
        AND      R3, R2, #SDA_IN
        ; mask it and stuff it in R3 to skip...

        ; leave bus in 'ready' state
        MOV      R1, #(SDA_HIGH + SCL_LOW)
        BL       toggle_printer
        ; Has 'slave' pulled SDA LOW? (bit set=LOW)
        CMP      R3, #SDA_IN
        BNE      iic_writebyte_noack ; no=Ack failed

        Pull     "R0-R3, PC"


iic_writebyte_noack
        ; clear the I²C bus (it 'floats high')
        MOV      R1, #(SDA_HIGH + SCL_HIGH)
        BL       toggle_printer

        ; abort from EVERYTHING!
        Pull     "R0-R3, R14"
        Pull     "R0-R6, R14"

        ADR      R0, iicnorepmsg ; !!! NOTE !!!
        SetV  ; YOU NEED TO DEFINE THIS ERROR BLOCK

        Return


iic_sendbit
        ; Sends a bit of data
        Push     "R0-R2, R6, R14"

        MOV      R6, R0             ; stash bit in R6
        ; Put the bit out with SCL low
        RSB      R1, R6, #1         ; = (1 - bit)
        ADD      R1, R1, #SCL_LOW
        BL       toggle_printer
        ; With the bit out, bring SCL high
        RSB     R1, R6, #1      ; = (1 - bit)
        ADD     R1, R1, #SCL_HIGH
        BL      toggle_printer

        ; Push SCL low again
        RSB     R1, R6, #1      ; = (1 - bit)
        ADD     R1, R1, #SCL_LOW
        BL      toggle_printer

        Pull    "R0-R2, R6, PC"
```

The exercise for the intrepid reader (yes, I mean *you*!) is to combine this in a useful manner. Or perhaps APCSify it for linking with high level code?

If you would like I²C via your printer port (it could be useful if you use an Acorn A4), but don't want the hassles of doing something with this code right way, you'll find the `PtrIIC` module at:
`http://www.heyrick.co.uk/software/ttx/`

Additionally, you'll find my *!Teletext* software for RISC OS, an early version of a multitasking teletext frame editor, and of course the *Teletext for DOS* which can now work directly with the I²C port of a RISC OS machine if you are using it via *!PC*.

Well, this concludes our look at I²C. We have code to implement it in both C and ARM assembler using our specially wired parallel cable. No, there won't be a BASIC version! Sorry. :-)

You could, fairly easily, convert this into 6502 code to work with a 6522. Here, as with the IOC, you can directly connect to just two pins on the 6522 as they can be inputs or outputs. I would recommend port A as the pins can be outputs but they'll be read as the actual level on the pin – thus acting as an input too.

Where you go from here is up to you. There are many I²C devices from single-chip teletext receivers to DTMF generators, video systems, tuners, and memory. Using the right parts, you can make an I/O board that will excel a 6522/µ7002D combination and the best part? It's a very simple interface. Or perhaps your software wants a simple dongle – you could fit an NVRAM chip into a parallel plug!

Rick, 2004/08/01

You may find the following macros useful:

```
; BSR : Branch to subroutine saving R14 (around it)
        MACRO
$label  BSR     $dest
$label  PushLR
        BL      $dest
        PullLR
        MEND


; Push : Push registers given in reglist
        MACRO
$label  Push   $reglist, $cond
$label  STM$cond.FD R13!, {$reglist}
        MEND


; Pull : Pull registers given in reglist
;         (use hat with care!)
        MACRO
$label  Pull    $reglist, $cond, $hat
$label  LDM$cond.FD R13!, {$reglist}$hat
        MEND


; PushLR : Push LR (optimised for one register)
        MACRO
$label  PushLR    $cond
$label  STR$cond. R14, [R13, #-4]!
        MEND


; PullLR : Pull LR (optimised for one register)
        MACRO
$label  PullLR    $cond
$label  LDR$cond. R14, [R13], #4
        MEND


; Ret : Return from function
        MACRO
$label  Ret     $cond
$label  MOV$cond PC, R14
        MEND

; Return : Return from function (same as Ret)
        MACRO
$label  Return $cond
$label  MOV$cond PC, R14
        MEND


; PullRet : Pull LR and Return (optim. for one reg)
;   This is equivalent to:
;     PullLR
;     Return
;   but saves an instruction as we load PC directly.
;
        MACRO
$label  PullRet    $cond
$label  LDR$cond. PC, [R13], #4
        MEND


; SETV : Set oVerflow flag (on 26 or 32 bit)
;                          ^^^^^^^^^^^^^^^
        MACRO
$label  SetV     ; DOES NOT TAKE A CONDITION
$label  CMP      R0, #1<<31
        CMNVC    R0, #1<<31
        MEND

        END
```

If you have any good macros or tips to share, *please* get in touch at the usual address.

Qu’est-ce que c’est, ça?

Now that it appears RISC OS has a future, thanks to Castle performing the *essential* 32bit work to lift us away from outdated parts, it has come time to turn a critical eye towards something that has languished for so long.

BASIC

While each new operating system patches and tweaks bits of BASIC (i.e. the COLOUR r,g,b in RISC OS 3.5 and later) there are a number of more important alterations that I feel would be necessary for BASIC’s continuance.

The suggestions described here are my personal “wish list” for BASIC. You will, no doubt, have other suggestions. Email them to me, share them!

1. Memory allocation

One of the greatest weaknesses of BASIC, to me, is the memory allocation. You have `DIM` and you have `CLEAR` and... well... that’s about it. Because of this, when allocating memory you either need to read the data twice (once to work out how big it is) or you have to fix some arbitrary size on the arrays (as Gavin did in the BASIC vs Lua example on page 7).
What we need are two more functions – `REDIM` and `FREE` which behave like *realloc()* and *free()* in C. Then we can dispose of those kludgy above-HIMEM memory management routines that everybody invents because BASIC can’t do this essential task for itself.

2. Structures

One of the great things about C is the “structure”. We’ve all had to figure out button clicks in our time, using code like:

```
DEFPROCicon_clickbar
  button% = wimp%!8 : x% = wimp%!0 : y% = wimp%!4
  CASE button% OF
    WHEN 2 : PROCmenu_popup(x%, TRUE)
    WHEN 4 : PROCwindow_open(cfg%)
  ENDCASE
ENDPROC
```

Fairly standard, yes? Let’s see it in C...

```
static void icon_clickbar(void)
{
   if (event->data.mouse.button.data.select)
      window_open(cfg);

   if (event->data.mouse.button.data.menu)
      menu_popup(event->data.mouse.pos.x, -1);

   return;
}
```

Now we have abstracted ourselves from pulling bytes from data blocks. It may not seem the greatest example, so consider the process of ack’ing a message, or updating an icon. These often involve a bunch of `memblk%!<offset> = memblk%!<`*`different`*` offset>` with little clue as to what is *actually* going on. In the C program, though it is more to type, it is obvious what we are doing – message_myref is message_yourref...
But the real power of C’s structs lie within the union. Consider part of my 6502 emulator:

```
typedef struct flagdef
{
   int  carry : 1;
   int  zero  : 1;
   [...etc...]
} flagdef;
typedef struct cpustate
{
   [...etc...]
   usion psr
   {
      unsigned int byte : 8;
      flagdef flags;
   } psr;
} cpustate;

struct cpustate cpu;
```

With this in mind, I can refer to individual flags using a variable such as:

```
cpu.psr.flags.carry
```

or I can save the entire PSR without having to build the byte from individual bits using this variable:

```
cpu.psr.byte
```

All of *cpu.psr.byte* is all of *cpu.psr.flags*, the extra name on the end of the flags specifies which flag. It is this ability to give *the same* things multiple ‘types’ that makes unions and structs so powerful.
Obviously, functions must be able to accept and return structures as parameters!

3. Omittable parameters and return values

Look at the following two lines of code. They are examples and have no special relation to each other.

```
PROCmungestring(input$, 0, key%)
junk% = BGET#fp%
```

In our first instance, what is the zero for? Is it some sort of flag and the programmer was too lazy to type FALSE? Does it specify a string offset?
In our second example, we are reading a byte only to discard it. Must we create a variable in order to achieve this?
Better, would have been:

```
PROCmungestring(input$, , key%)
BGET#fp% : REM discard byte
```

BASIC can fill in the missing “unnecessary” value, like it does for missing parameters to SWI calls.

4. Better string compatibilty

Within BASIC, someplace, the end of a string will be detected using something like:

```
    CMP     Rx, #13
    BEQ     end_of_string
```

This is all very well for &0D terminated strings, but since C programs and much of RISC OS return NULL terminated strings (which BASIC is likely to interact with), it is time BASIC supported these strings too. No, not by all of us users having code to scan the string data and poke &0D over &00 bytes, but by BASIC itself coping with either.

It is easier than it looks. The code above? Make it:

```
    CMP     Rx, #13
    CMPNE   Rx, #0
    BEQ     end_of_string
```

Possibly, this could even expand to:

```
    CMP     Rx, #13
    CMPNE   Rx, #10
    CMPNE   Rx, #0
    BEQ     end_of_string
```

so it can cope with a block-loaded file laid out using RISC OS style line termination.

5. Additional variable types

BASIC should be provided with one additional data type for use in structures, I'm not sure how it would work in syntax terms, but it is a type that can be from 1 to *n* bits in size, allowing easy definitions of bitfields; and structure elements should be able to take a parameter to give a size, like:

```
    STRUCTURE MYSTRUCT
      firstbit%[1]
      secondbit%[1]
      padding%[*]
      ints%(4)
      str$[32]
    ENDSTRUCTURE
```

To break this down:

The structure is called "MYSTRUCT".
It contains a variable "firstbit%" which is 1 bit,
and another "secondbit%" which is also 1 bit.
The '*' in "padding%" means align to next word.
There are four "ints%" integers (0,1,2,3).
Finally, there is a string which is 32 bytes *max*.

I thought long and hard about other variable types, such as *byte*, *halfword*, and *variant* but I'm not sure that these are really as useful as they sound. The mem%?offset idea can cater for bytes perfectly well.

6. Array subscripts

The sharp eyed among you may have spotted an "error" in the above description. ints%(4) would be (0,1,2,3,4) and not (0,1,2,3) like I said. This is because the BASIC language never really figured out if it was counting from zero or counting from one, so it always adds one to allow either method to be used. There should be some sort of keyword that *forces* the default behaviour to count from zero or to count from one, but *not* both. Perhaps it is no surprise that people weaned on BASIC have trouble getting to grips with weird crashes in C after writing to array offsets that simply don't exist!

But this isn't the end of the story. Far from it. Our version of BASIC is pretty sophisticated, so we should be able to expand this sophistication with:

```
    DIM graph_xpos%(-20 ... 20)
```

Which will define an integer array *graph_xpos%* with forty *one* elements counting from -20 through zero and onwards to 20.

We shouldn't need to do this with offset maths in brackets, BASIC should offer us this itself.

There you go. My BASIC wishlist. I've carefully omitted all references to OOP as, frankly, I never saw the point of C++ and the like. My uncle writes educational course books and he sent me a copy of a book written to accompany an introductory course in C++. I've read it several times and each time I keep thinking "But I can do this in C!" and I'll have the added benefit of not having to litter my programs with even more &punctuation<<symbols! With careful use of modular code, headers, and includes, you can even fake concepts that don't exist in C, such as *friend*. Structs with pointers-to-functions can fake *classes*, though it is a seriously grotty way to do it. No, sorry, C++ and the OOP concept has not been sold to me. I just look at C++ programs and think "oh my god! what sicko thought *that* up?" and "Hey! Let's wear out the '&' key!" :-)

The *final* thing that I have to say on this subject is that any newer version of BASIC with some or all of these sorts of features *must* be available for ***all*** machines from RISC OS 3.10 upwards. Changes such as these will introduce coding styles and methods that simply do not exist on current versions of BASIC. If Castle, say, implement this but keep it to RISC OS 5, then they'll be restricting its features to RISC OS 5 which while it may not seem a bad thing how many people do you think will use the newer functionality? Those with older machines won't be able to, and those with an Iyonix might decide that cutting out the rest of the market for the sake of a few functions isn't terribly wise.

Or, another way to look at it, the new C compiler with the 32bit CLib and 26/32 neutral code is as happy on RISC OS 3.10 as it is on an Iyonix. I'm using OvationPro on an Acorn A4 right now (let me tell you, memory is *extremely* tight!). Is this pure chance? Or was it designed in order to get the new CLib accepted?

Rick 2004/08/08

Reader Survey Results

Here are the results of the Frobnicate jury...

The first thing that I wasn't expecting is that most of those who replied are older than me. Does this mean that *Frobnicate* is reaching a more learned audience, or that the younger people are DivXing their lives away in the Microsoft camp?
Sadly, not one single female replied. It's a man's world, evidently. `<sigh!>`
Abilities and capabilities varied enormously, though as a number of people submitted mini-bios, it was clear that they started with Acorn and saw it through. One person, in fact, started with an Acorn Atom!

In general, most of the respondents possess a number of computers. Perhaps I should show this stuff to mom who is forever saying that nobody *needs* twelve computers – no, she just doesn't *get* it, does she?

Many have their computers networked. I think, these days, a network is essential if you actively use more than one computer. The ability to have both machine's files appear on each other with no bother and no fuss is so much better than *any* other way of moving files around.
Most *Frobnicate* readers connect their networked machines together using some form of "router", but they use something different for ADSL broadband – better security than a shared ADSL/router combo.
Few readers bother with modems these days. 512K broadband is the average.

On-line storage is a shocking thing. If you omit the one person who is only a few Gb short of a *terrabyte* of storage (!!!), the average about is around 200Gb.

Many respondents *have* an Iyonix. Typically it was purchased on the grounds of "it has been a while since I last bought a new computer, so...". It seems it's a love/hate/love relationship with the machine. Weird!

The question of Windows vs RISC OS brought a huge variety of different answers. Some said that Windows95 onwards thoroughly kicked RISC OS's ass while others feel that Windows XP is about on equal par with RISC OS 3.10. I guess the subjectiveness depends largely upon what you use as your comparison point. In some cases, Windows always has kicked our asses (and always will), while we excel in other points and they've not caught up in a decade and a half of trying.

Those with Unix experience were much more philosophical about comparisons and pretty much refused to make any comparison on the grounds that Unix and RISC OS are two entirely different entities. One respondent suggested the reason that the Unix GUI system is in such disarray and a klutzy mess of hack upon hack is because it was never designed to be a desktop OS. No, Unix will be quite happy to sit in a dark corner and sort huge amounts of mail without demanding paid holidays or losing five million messages each month...

The best features of RISC OS are the font rendering, the filer, and the three button mouse.

Much of the "what needs to be improved" discussion regarded the friction between the two companies developing RISC OS. Other than that, it was the usual wish list – a web browser that *actually works* (their words, not mine, but...), pre-emptive multitasking, multi-threading... as one put it, "the same things we were asking for back in 1997!".
Time for a long, sad, sigh.

The favourite commercial packages were ArtWorks (the new version) and OvationPro. I don't think I need to explain why these two were chosen.

The most hated packages, this time around, tended to be packages that exhibited some sort of flaw which made it unusable (many of the respondents wondered if it was something 'odd' with their computer), and in a few cases the vendor refused to accept the package back because it had been opened. Sure, the product *could* have been copied, but the other viewpoint is *I paid good money for this and it doesn't bloody work!*
It is quite depressing to see this sort of attitude permeate into the RISC OS community.
For what it is worth, the companies and the software packages are *all* different.

Favourite shareware, pretty unanimously, is *!Thump*. One respondent said that it makes *Powerpoint* look silly in comparison (what, it doesn't already look silly enough on its own?!?).

The favourite "free" software was split between Zap, StrongEd, and the PDF viewer.

Favourite non-computer hardware, generally, is either a nifty colour printer or a nifty digital camera. One person was extremely pleased with a scanner/printer/photocopier thing but admitted the need to use UniPrint because there are no RISC OS drivers for it.

Rather ironically, the Iyonix was voted the suckiest hardware several times due to inconvenient crashes (an oxymoron, I don't think I've *ever* experienced a 'convenient' crash!), lost mouse clicks, and the apparent slowness with which on-board hardware is being supported.
It seemed the idea of including a DVD-ROM drive irked a lot of people since RISC OS has no DVD facilities.

But to make things *even more ironic*, quite a number of those who said the Iyonix was 'sucky' also said it was their *best* purchase! Mainly, quoting the performance increase as a major factor, and everybody but everybody loves the near-silence of the machine, as well as regular system updates and bug fixes from Castle.
In general, then, it seems that "things could have been better" but most seem more than happy with Castle's efforts. I'm sure the Merlin project will cause some amount of drooling over keyboards...

The most regrettable purchase tended to be things that sounded 'good' at the time. One respondent nominated a hand-held scanner that he bought for his A3000 way back when. Having used a hand-held scanner, I know it is totally impossible to get perfect scans, even if you wedge the damn thing between lumps of breezeblock with greased wooden guides – yes, I actually tried this and a part of the scan was *still* wonky!

The best RISC OS computer was *almost* unanimously the Iyonix. Like me, a lot of people are happy that RISC OS is now 32bit so we can be free from outdated StrongARM/ARM7500 machines. One person, however, suggested his A305 (with Arthur 0.1 and half a megabyte of memory) – saying that it was the only machine Acorn put their heart into and everything later suffered from greed. I *hope* he was being facetious!!!

The main RISC OS personality is David Pilling.
In second place, Paul Beverly.
Nods, also, to various members of the Castle team for the Iyonix, and thanks also to Sophie Wilson.
One person nominated a certain Mr. Middleton for a reason that I cannot print. Well, it's a nomination isn't it? :-)

There you have it. An overview of the *Frobnicate* readership:
Mature, intelligent, wise, and decisive. And no, I'm not sucking up.

It isn't too late to send in your answers! I'd especially like to hear from a few women to see if/how a female viewpoint may differ.
As always: heyrick -at- merseymail -dot- com

The RISC OS wars

Whose licence is it anyway?

Following my last report on the "future" of RISC OS... It has been brought to my attention that the RISC OS *Select* scheme is all but over. The new baby is RISC OS *Adjust* (well, let's face it, RISC OS *Menu* would have sounded pretty damn silly!).

But that isn't all. Shall I mention the *troubles* between RISC OS Ltd and Castle? This *appears* to have sorted itself out, for the time being at least. While it *appears* that the problem revolved around VirtualAcorn (not being an actual real existent lump of ARM silicon...), we should feel sorry for the other sub-licencee that discovered one day he had been ordered to cease distribution. *Even if* this is now resolved, I can imagine that sort of thing could agitate the nerves more than a few cups of coffee can sooth, and it makes me wonder what sort of 'protection' can be offered to a sub-licencee that apparently gets caught in the middle of somebody else's effluent discharge.

That *Select* and *Adjust* stuff ... For those not familiar with RISC OS who maybe found the PDF->HTML version of *Frobnicate* on Google, our mice have three buttons and always have (nerr!). The one on the left acts pretty much like it would under Windows, hence it is called `Select` (because you use it to select stuff). The button in the middle is called `Menu` (because it pops open the context-sensitive menus) and it is roughly akin to the right mouse button under W95 to XP except that it is our *only* interface to the application's menus (we don't have a manky menu bar across the top of the window). The button on the right? This modifies selections, so it is called `Adjust`. In brief terms, it is similar in concept to Ctrl-clicking in a directory viewer under Windows, only it is much more capable as – for example – you can Adjust-click on increment buttons to decrement or scroll buttons to scroll the opposite direction. Thus, you can go "both ways" without having to move the mouse around. So the first RISC OS scheme was called Select, and I guess they figured RISC OS Menu was a dumb name, so the new one is RISC OS Adjust. One could muse about potential names for the *next* scheme, RISC OS Doubleclick perhaps?

I won't waste time describing the things Adjust brings to RISC OS – look at ROL's website for the details. Heck, ROL probably wouldn't give me the time of day, I'm with 3.7 and happy. In *actual fact* this very article is being written on an A4 (RISC OS 3.10!).

One of the things that ROL is expecting to produce is a new ATA-based filing system to allow access to really big harddiscs; the kind of drive you need to store your respectable (and growing) MP3 collections...
Meanwhile Castle are looking to have an updated filing system that can pass the large swathes of data required for DVD playback. Yes, people, somebody is looking into DVD playback on RISC OS machines – but you'll need an Iyonix as the older kit just won't cut it.

Makes me wonder – will these development efforts run in parallel or will somebody think to try to merge them to form "a better filing system for the future"? Mmm...

This brings us to Merlin. It is essentially a sort of Select for owners of a shiny new Iyonix. On the way will be all sorts of enhancements to RISC OS 5 (like, for instance, the *fairly recent* release of sound sampling software – why'd it take so long to arrive!?!?), so maybe soon those with an Iyonix will no longer wish for Select to be 32bit. Early days yet, but Iyonixy things are looking up for sure.

Where does that leave *us*?

Well, "us" breaks down into two main groups, "users" and "programmers":

Users
One thing we will have to just *accept* is that RISC OS 4 and 5 are different, just as RISC OS 2 was different to 3, and 3.1x was different to 3.5, and 3.7 was different to 4. The only added complication is that 4 (Select/Adjust) and 5 are made by different companies. It is perhaps worth pointing out that an upgrade path is not possible from 4 to 5 as the hardware is too different, but then again an upgrade path was not possible from 3.11 to 3.5 for exactly the same reason.

With regards major file formats – Drawfiles should remain as Drawfiles, Sprites should remain as Sprites. The things people take for granted should be largely left in peace. However, now that I have said that, I do not see any great hardship in adding extensions to core formats. Here I am talking of sprites with alpha transparencies. After all, JPEGs were added to Drawfiles and sprites went "deep". However as a service to the community (and to aid in getting their extension more widely accepted), those who modify a core format should release a tool for the Library directory that can 'flatten' the format. In the case of the alpha sprite, for example, you can set a threshold and all the transparency over this value will be made "transparent" and everything below will be made "normal" in terms of the on-off transparency of older versions. It shouldn't be too difficult to do and so long as it'll run on any RISC OS machine, people can drop this into their Library and use it as/when they need.

For the end user, so long as their applications work, things should be okay. It'd be nice to have the same perks on both versions of RISC OS, but that is nothing really compared to the culture shock of using RISC OS and XP! For what it is worth, I would suggest Castle talk to Thomas Olssen about using his ProFiler (formerly Filer+) within RISC OS 5. Then, all those people who thought the filer niceness in Select will then realise that Select'll be a sorrowful retrograde step. Why do you think I still use RISC OS 3.7? I wouldn't be without ProFiler – I *really* like it!

Programmers
Things change. It is the inevitable consequence of the passing of time. Call it "entropy", if you are lost for a better description.

RISC OS 3.5 introduced the concept of mode definition files, the sadly flawed concept of claiming chunks of non-application space for data storage (dynamic areas), and at a lower level it changed how to jack in to processor vectors. By and large people coped, just as they coped when a subtle change in RISC OS 3 made the old `SYS "OS_UpdateMEMC", 64, 64` speed-up tweak (much touted in the magazines as a way to get BASIC to run faster, not that I ever noticed much difference on my A3000!) lock the machine solid. The danger that exists today is due to two companies developing *different* versions of RISC OS. It is highly imperative that the two talk (yes, I said *talk* – as in "dialogue" (and I don't mean boxes...)) to each other to ensure the shared parts of the API stay in step. It is no good whatsoever to have, for example, RISC OS 5 to access its serial port using `SYS "Serial_Rx"` and `SYS "Serial_Tx"` while the rest of RISC OS uses `"OS_SerialOp"`. [aside: logically it would make *more* sense for the serial calls to *not* be OS SWIs, but this is a historical naming issue and not something to change overnight – *if it ain't broke*...]

It is of paramount importance for the RISC OS companies to document these changes and maintain some sort of reference, either on their own websites or some sort of central repository (such as *iconbar.com* perhaps?). I have no figures to hand, but I think it'd be safe to say that a number of RISC OS utilities are maintained by people that DON'T have an Iyonix (yet?). While much of RISC OS may be close enough, one or two little things may be different. As a good example, take the Dynamic Areas concept on a 16Mb RiscPC and then see how well it "scaled up". It wasn't a bad idea, just a few things made it difficult. Now your 16Mb RiscPC coder may not fully appreciate this. Make her aware of it.

Take me, for example. I have "inherited" FYEO2 (!) to which I added support for the BMP format (I needed it for something I was doing in the PC domain; BMPtoSpr->ChangeFSI->Paint was too fiddly and using ChangeFSI directly was too slow). Along the way, I 32bitted it, patched in (crufty-but-usable) support for progressive JPEGs, and fixed up a few things. It failed on an Iyonix. Weird error that implied there was no suitable screen mode (though you really needed the source to figure it out, the message it gave wasn't so helpful!). The problem? Additions to what RISC OS returned as a mode descriptor (logically enough) that FYEO2's mode selection code wasn't able to recognise. So now, FYEO2 is RISC OS 5 compatible! Does Castle have a list of changes freely available? I'm not talking about an entire RISC OS 5 PRM, just useful nerdy notes describing the main changes since, say, RISC OS 3.70 (I'm using the PRM set plus PRM5a as the 'base' version of the RISC OS documentation). Then, I/you/we could skim though and pay attention to those facilities that we know we use, and from this, our software will be that much more likely to work correctly under RISC OS 5 the *first* time...

I've said this all before – but since you don't shell out hard earned dosh to me for *Frobnicate*, I'll have no qualms about saying it again! While it is completely valid that each company is a company and money must be made, it is imperative that all rifts are patched and wounds healed. That offers the *best* chance of survival for our small community. Of course, the alternative to dialogue and sharing is to think "bugger it" and do things "Miiiiiiiigh waaaaaaaay!". It'll probably work in the short term, but if the heart of RISC OS and its API starts to differ by too much (in cases where both do the same things with a totally different API), then developers will be forced to choose. Costs may rule out the use of loadable modules to support 'both' versions, so instead they are likely to develop upon whatever is perceived as their largest market (and 'home' developers will probably stick to whatever is closest in API to the machine that they program on). That isn't a better end result. Not at all.

Random musings
As for new hardware, there is one person (no names mentioned) who posted a number of pro-RISC OS messages. Okay, no problem with liking RISC OS (I'll gloss gracefully over the crosspost débâcle). But then he touted the idea of making a new RISC OS computer based around the ARM7500. Why? Why? Whywhywhywhyohgodwhy? Why develop *another* 40/56MHz part? Have ARM made anything cute since the XScale? Probably! That's where a new developer *should* be looking. Today's tech, not yesteryear's tech. He *obviously* doesn't get it.

I have received a number of comments regarding my thoughts of buying a PC instead of an "Acorn" in the future. Some said that it was about time the pro-RISC OS crowd woke up to the outside world, while others screamed about the flood of viri and porn (so that'll be syphilis then?) that you find lurking within the Dread Machines on The Other Side. It is a tough choice, for sure. RISC OS does things the way I like while XP grates on the nerves. But just *look* at all the stuff you can get done on a bog-standard PC. The full internet *experience*, not the cutely crippled version that you see with RISC OS. DVD reading *and* writing. Got a new bit of kit? You know it will come with Windows drivers. I have a little Achiever ADC-65 digital camera. 256 pixels by 256. Works on my Windows 95 box. Sort-of works under !PC. Is there a driver for RISC OS? Nope. Is the spec available so I can make a driver for RISC OS? Nope. Did Achiever bother to reply to my two emails asking about this? What do *you* think!
There is some pretty nifty hardware out there, hardware that has the potential to work with RISC OS via USB, but unless a driver gets sorted out it is a closed door. And the driver is likely to cost money, just as a working web browser costs money.
Now you may point out all sorts of "hidden" costs of the ownership of a PC and all this bundled software adding to the price...

Shut up! Seriously, be quiet.

I did a price comparison from a French magazine, dated March 2004. I can get a good desktop machine (circa 3GHz, 512Mb RAM) ***and*** a 19" CRT (LCD panel would be nice, but hideously expensive) ***and*** a reasonable second hand laptop ***and*** a digital satellite receiver with dish rotator ***and*** a new colour printer ***and*** a new whizzy-bang TV capture card ***and*** I can kit those machines, and mom's, with WiFi ***and*** I'd have some money left over to buy VirtualAcorn... and maybe a few DVDs from BlackStar.

Or I can get an Iyonix.

If I have a good win on the lottery, it'll be an Iyonix for sure. Otherwise, I'll have to shop more wisely. I'm not too worried about Microsoft obsolescence. My (real) PC runs W95 and the Craptop Win3.11. If I only get XP and pull eight years use out of it, I'll be okay...

...of course if somebody can come up with a good RISC OS machine with all the goodies and a comparable price tag...? I don't suppose there's ever likely to be a newer RISC OS laptop (that isn't an emulation)?

What made BASIC sucky?

You read articles and books and they all talk about how *bad* BASIC is. Indeed, Gavin refers to this in his article on Lua though he does acknowledge a point that many others seem to overlook – that BASIC was available for the home computer while many of the ‘better’ languages were not. Indeed, the only C compiler I've seen for the BBC micro was rather restricted *and* it needed a ROM. No – you can't argue that BASIC needs a ROM too, as it is there in every machine Acorn made so it can be considered a “standard” part. :-)

But, besides the abundance of astoundingly *crappy* versions of BASIC, we must also consider the literature produced intended to teach BASIC to people. Because, in many cases, even a good dialect of BASIC is ruined by a book that teaches badness.

Our first attack is the Maplin Elecronics magazine #43 (April-May 1991). Our author, Jeff Scott, says:

> “*The trouble is that it is very easy to write BASIC badly. A BASIC program can be written using 'GOTO's entirely with nary a subroutine in sight.*”

Sound advice, but in the very next instalment we see this example:

```
 10 REM PERMUTATIONS OF 4 DIGITS
 20 PRINT "TYPE ANY 4 DIGITS"
 30 INPUT A(1), A(2), A(3), A(4)
 40 FOR J1 = 1 TO 4
 50 FOR J2 = 1 TO 4
 60 IF J2 = J1 THEN 130
 70 FOR J3 = 1 TO 4
 80 IF J3 = J1 THEN 120
 90 IF J3 = J2 THEN 120
100 LET J4 = 10 -(J1+J2+J3)
110 PRINT A(J1); A(J2); A(J3); A(J4)
120 NEXT J3
130 NEXT J2
140 NEXT J1
150 END
```

Well? What can I say? Jeff's dialect of BASIC includes the `WHILE...WEND` syntax so I can say it isn't BBC BASIC. Obviously it is an inferior BASIC that doesn't support procedures, indenting, descriptive variables with more than two characters in the name, or lower case.

Then again, maybe the author of that program is the one at fault – who knows?

Now for the motherlode. *30 Hour BASIC.* An honourable objective (try learning C that quickly – on my bookshelf are two speed learning books, one will teach me *Italian* in less time than the other will teach me *C*!!! (disclaimer: the Italians have brains and will try to figure out what *Ewna pee-anta dela sitya par fav-oh-ruh* might actually mean; the compiler will just whinge and throw four billion irrelevant errors at you, like a single missing semi-colon upsets its entire universe so much that it'll wet itself and cry a lot, but it is generally too stupid to do something useful like stopping...).

The cover of the *30 Hour BASIC* book contains a variety of pictures of a BBC micro in use. It is written, presumably, to support The Micro Project (that which made the Beeb famous) back in the '80s when music was great and the cool people dressed in black. On the front is the BBC owl logo with BBC and NEC written around it, and it says “The Computer Programme BBC TV” and “National Extension College”. Written by Clive Prigmore, this is a correspondence text for course M027.

Amusingly, inside the introduction is the question “Do I need a microcomputer?”. Well, okay, you don't need a microcomputer for the course, but one could wonder why you wish to learn BASIC if you don't have a computer! There are many examples, laid out in a fairly structured format. Obviously, as this *is* an NEC course. The primary let-down is that it has been made “more accessible” to people *not* using a BBC micro. The introduction states:

> “*There are many versions (dialects) of BASIC, each created by a computer manufacturer modifying the original BASIC. As far as possible we have kept to the common core of most microsoft BASICs.*”

As this book was written in '81, I'm not sure exactly how many BASICs microsoft (sic) actually had then, but I think we can agree on the general crappiness. How on earth did this book – claiming to keep to the core of microsoft BASIC(s) – get the BBC Micro logo on the front? Well, they *do* state:

> “*All the programs will run on the BBC Computer and the course includes special guidance on programming this microcomputer.*”

Anyway, it has been made “more accessible”. In other words, a good education resource has been dumbed down to suit ghastly versions of BASIC;

thus resulting in examples such as that on p216...

```
10 REM**BISECTION SEARCH**
20 CLEAR 100
30 DIM N$(20):DIM T$(20)
40 I=1
50 READ N$(I), T$(I)
60 IF N$(I)="ZZZZ" THEN 100
70 I=I+1:GOTO 50
90 REM******
100 N=1:REM*WE ARE USING ZZZZ THIS TIME*
110 REM******
150 INPUT"QUERY NAME";Q$
200 REM**START OF SEARCH******
210 PRINT" L";TAB(5);" H";TAB(10);" M";TAB(15);"N$(M)"
220 L=1:H=N
230 IF H-L=1 THEN 500
240 M=INT((L+H)/2)
250 PRINT L;TAB(5);H;TAB(10);M;TAB(15);N$(M)
260 IF Q$=N$(M) THEN 320
270 IF Q$<N$(M) THEN 300
280 L=M:GOTO 230
300 H=M:GOTO 230
320 REM**END OF SEARCH******
330 PRINT Q$;"'  S TELE NO. IS ";T$(M)
350 GOTO 600
500 PRINT Q$;" IS NOT IN THE LIST"
600 INPUT"DO YOU WISH TO LOOK FOR ANOTHER NAME"' R$
610 IF R$="YES" THEN 150
620 END
900 DATA AAAA, 0000
910 DATA BENNY, 1234
920 DATA COPPER, 9832
930 DATA DRAPER, 1980
940 DATA EDDIE, 7294
950 DATA GWYNNE, 5821
960 DATA HETTY, 8632
970 DATA MORLEY, 7832
980 DATA PROSSER, 1383
990 DATA SMYTHE, 1147
1000 DATA WEEKS, 5529
1010 DATA WILSON, 9936
1020 DATA ZZZZ, 9999
```

I have switched to the Homerton type style as this is what the book uses for listings. You can see how much harder it is than a nice clear `Corpus`.

The program itself. Well? What can I say!? I have distinct déja vu of having said that already, but the sheer crap in this program is really quite something. Not a procedure in sight and the code is total spaghetti with bucketloads of Parmesan on top. Additionally, we run the gamut of lines from `10` to `1020` in an...erm... *41* line program! There's another lurking Gotcha! for people using this course.

Take time to consider this again. This is from a proper NEC course text for a proper programming course. No big deal, right?
Wrong.
Very wrong.
It *is* a big deal.

You see, instead of saying "this is a good BASIC and this is a bad BASIC", the course is trying to cover everything and ends up repeating the lie that it is *acceptable* to write code littered with GOTOs and, essentially, structureless. Without structure. Disarray.

On p40, it has *this* verbiage to say about GOTO:

> "*[...] we ought to warn you about the dangers of using GOTO. [...] But because GOTO allows us to jump rather at random to any point in the program, it is often used in such a way that the logical structure of the solution is broken up by 'jumps of convenience' to other parts of the program rather than by following the logical structure of the analysis of the problem. We will, therefore, use the GOTO statement sparingly throughout this course. [...] We hope that you will also try to follow our example and use the GOTO statement as little as possible.*"

And the very next example has an implicit GOTO (`IF <blah> THEN <line number>`), as do the inequality examples after, and the flowchart example after that. In fact, the next example *without* some sort of GOTO appears twenty three pages later.
On the topic of PROC or FN or procedures themselves, the book had exactly this to say:

> ""

That's right. *Nothing*. Neither topic was in the index. Subroutines rolled in on page 207 (of a 251 page book) and was implemented using GOSUB ... which is just a GOTO in a different guise!

Please, don't misunderstand me. I'm not trying to state that BASIC is a "perfect" language. It has a number of faults and limitations. I'm not sure I'd go as far as Edsgar Dijkstra in saying that an understanding of BASIC is a great disadvantage:

> *It is practically impossible to teach good programming to students that have had a prior exposure to BASIC: as potential programmers they are mentally mutilated beyond the hope of regenerations.*
> [How do we tell truths that might hurt; 1975]

Every language has good points and bad points; as I said in the last issue, the main factor in determining what makes a *good* language is the question of what you want to ***do*** with it.

Or – on the next page we'll turn this argument around and look at it from another direction.

BASIC is responsible for unstructured code

While 'some' BASICs seem to go out of their way to make life difficult (no procedures, for instance), it is a blatant lie to accuse BASIC of lacking structure and clarity. I've seen some beautiful things expressed as BASIC code, while the *International Obfuscated C Competition* has turned indecipherable C code into an art form.

Some people accuse BASIC of gross abuse of GOTO. Well, this *may* be true but it is forgivable when, for example, courses that should know better use GOTO extensively – even after warning about such use. The answer to anybody whinging about GOTO is that they have *obviously* never cut their teeth on assembler. You don't get nice functions in assembler, you get GOTO and (sometimes) something akin to GOSUB.

BASIC is outdated

Some *languages* are outdated. Ever used TurboPascal 5? It seems horribly ancient. So does QBASIC. Other BASICs are evolving. Our BBC BASIC beat the bunch in the early '90s making a *very nice BASIC indeed*, but it just hasn't progressed so much since then. In more recent times, Microsoft's Visual Basic (which is about as similar to BASIC as, say, C is!) is doing good trade by taking the concept of BASIC and what BASIC is supposed to be and expanding it to do all the sorts of things they think a modern language ought to do.

BASIC is too simplistic

Yet another one of those 'difficult' questions. *Some* BASIC versions are rather restricted while others can do a myriad of things. It might be fair to say that, with its built-in assembler, BBC BASIC V can do *everything* that a C programmer can do. *WebsterXL* is an entire JavaScript-capable web browser written in BASIC. The primary problem that *WebsterXL* suffers is that of speed, since BASIC is interpreted and C is assembled. That aside, *WebsterXL* makes my website look quite nice, thank you.

And on the subject of "simplistic", please remember what the BASIC acronym actually stands for...

...*B*eginner's *A*ll-purpose *S*ymbolic *I*nstruction *C*ode. It wasn't intended as a power tool for hardcore hackers. That such things as *WebsterXL* are possible should stand testament to the versatility of the language and Acorn (Sophie Wilson) for providing such a useful version of it.

GOTO is necessary in BASIC

The only answer to this is that if GOTO is necessary in your dialect (I mean *really* necessary and not just a convenience), then it is high time you looked for a different version of BASIC!

BASIC encourages sit-down-and-type programming

...which, as evil as the practice is, is exactly how I write most of my software, be it is BASIC or C or even assembler. I have an idea in my head and I watch code forming in front of me as I give input focus to *Zap* and start bangin' on the keyboard like a chimpanzee ("that ain't workin', that's the way you do it "). If I had to sit and plan every function and every procedure and make cute little flowcharts, I would write better code and spend less time debugging. But, paradoxically, I would write *no* code. I would think "screw this" long before any code had been written. I'm impatient. I rejoice in the peaceful serenity of a full system crash. You just can't get the same sort of mental exercise by stabbing a pencil at a piece of A4.

BASIC is friendly. BASIC has idiosyncrasies, but then so does *every* other language. Gavin shows us Lua in the first few pages of this issue, and personally I find the ability to reassign 'function' values to be horrendous; but some people may really want such a thing which is why the language can do it. Good, bad, or mostly forgettable, it is still an idiosyncrasy.

Until I felt confident with C, I did most prototyping in BASIC. It was quick and simple. And friendly.

You can assign a new variable anywhere you want it!

One of the things they'll tell you in college (while inflicting the likes of Pascal on you) is that it is seriously bad form to define variables at point of use. They should be at the top of the module (if semi-global), at the start of a function, or maybe buried in some header file someplace.

Well, I have written the following in C:

```
for (int loop = 0; loop < count; loop++)
```

This is just one of the nice features introduced in the *ISO 9899:1990* specification. It looks like BASIC was there *way* ahead of the mighty C! :-)

The conclusion? Some dialects of BASIC are quite awful, but you cannot take a handful of pitiful examples and hold it to be representative of the language itself. Yes, some BASICs suck. Ours is just a little 'old' but it doesn't suck. It was just never fully appreciated.

Rick, 2004/08/10

Diary of a hacker...

I guess the biggest question is why the entries in my notes stop at 2002. I check myself out of the hospital against the advice of hoards of so-called specialists, and head for the library. There, all is revealed. Sopowitz and my fave policeman were engaged in a spectacular gun duel in the middle of town. Sopowitz took the Lloyd's bank while the Good Guys That Aren't So Good Really took WHSmith. Panicking cashier girls ran screaming while the two unloaded several dozen clips at each other. Where a policeman found a semi-automatic is anybody's guess. The policeman ran out of ammo first. Sopowitz, being a cocky son of a bitch, walked right out and carried on firing, taking out a few nosy school children along the way. For his efforts, he got a large hunting knife in the eye, but not before he responded with a hail of bullets. Both died. I'll never get to read a psych report on Sopowitz, but... you have *no* idea how much I *wish* I was there just so's I could kick the living daylights out of Sopowitz's carcass. I'd probably be arrested, but hell, prison time would surely be worth it for that.

The locks on my flat had been changed, but the cheapskate bank (or whoever) used stupid locks that were near enough pickable with a paperclip. To think my expensive hardware was *that* insecure. To think that my expensive hardware is scrap these days. As I entered I saw burn marks up the wall by the stairs. Blatantly arsonish. I mean, there's no electric or gas point *anything* there. What exactly would catch fire if it wasn't put there with the intention of catching fire? If Sopowitz had been clever he'd have simply set a small fire in the cupboard by the door. It is full of junk mail. There is a 150 litre bin bag and I empty it when the stupid advertising circulars no longer fit. It would have burned the place down, but I guess looking in cupboards is something women do regularly while men just pretend they might do it if persuaded often enough.

No electricity. The fuse-board looks okay. The meter is dead. I smash the master fuse socket open with the business end of a mallet and, using insulated pliers, I shove in a Draper spanner. That oughta carry sixty amps.
The phone line is more of a problem. The outside phone lines have been removed and outside my door is a stupid little cableco box with a stupid bunch of terminal blocks in it. It looks like everybody on the street has one. No problems. I'll take my laptop and acoustic modem and I'll set myself up for the telecom broadband.
My laptop is dead. Not only is the battery pack well past it, but the CMOS RAM is history too. Nothing wants to hold a charge.
My main computer is just the same. No charge, no remembering what I set it to. Oh my god, what the hell was my login password anyway?

This isn't fun. Really it isn't.

There's a beer in the fridge. Warm, expired two years ago, but it'll do. I stab a hole in the bottom with a Bic pen and I put this hole to my mouth. I flip the ring-pull and feel the beer pour into my stomach.

I don't feel better. I down another can, and then an entire bottle of stupid beaujolais goes down the hatch.
Some drink to remember and some drink to forget. I can't bloody remember *what* I'm trying to forget.
Then another beer. That's the extent of my alcohol collection. A bottle of crappy wine and cheap beer with fake German words written on the side.

Am I happier? No. Do I feel better? Not really. You know, this getting drunk lark is way overrated.
Anyhoo, I'll just sit here on the lino in the kitchen and let this lot percolate. *Lemsip* doesn't work immediately, neither does *Ritalin* nor *Prozac*. Give it time, yes?

I ztil dun feel zoh goot buh I iz awlouta beer'n'shet.
Yu no wat? Bugr this fuh games a'soljers. Me guna go bed now.
Uh, but ferst pewkes me. Dam.

A Matter of Language

Two of the diffi cult things in learning a language (a ‘real’ language, not a ‘programming’ one!) are the genders and the exceptions to the rule.

I wonder where the concept of gender came from. A frying pan is a poêle in French, a sartén in Spanish, and a padella in Italian. All are *feminine* nouns. How can you grasp the concept that a frying pan is female? A computer is *masculine* (ordinateur – F, computadora – E, computer – I) while programming is *feminine*! (programmation – F, programación – E, programmazione di computer – I) I fi nd this ironic that I, as a man, engage in the act of programming (which is something largely performed by males) on computers called Alyson, Anna, Amy, Angela, Ariana, and Angelique (to name just six of them!). I think of my computers as feminine – I'm sure a psychologist would consider them a ‘female’ substitute or some other psychobabble; but *none* of my computers have been male. Does a female think of her computer as male? Or is she more practical and doesn't assign it a gender or a personality? Why is the thing itself considered male in the Latin-based languages, but programming is female? Who thinks up these things anyway?
Now here comes an interesting thing. A turnip (yes, the yucky vegetable) is a “rapa” in Italian, *feminine*. In Spanish it is “nabo” and it is “navet” in French, both *masculine*. So the genders of things do not necessarily agree. This would imply that perhaps English has not simply ‘lost’ gender over time. The German language confuses things further by having a third gender *neuter*. My Swedish dictionary mentions *neuter* and *common gender* but doesn't say anything about masculine or feminine.
How can I possibly think of a turnip as male (except in Italy)? You could say it has a certain phallic appearance, but so does a carrot and all three Latin languages agree on its carrotty femininity (carotte –F, zanahoria (!) – E, carota – I). For what it is worth, a carrot is “morot” in Swedish, common gender. Go fi gure.

Here is what I think: English, the language that so many English speakers say is easy to learn, has no genders because our heads are full of all the exceptions. You know what? I think English should be like Swedish or Spanish and contain accents. Then maybe the following would be obvious to débutantes:

Plough
Through
Rough

What's the rule? “ough” following a consonant is pronounced *how* exactly? Plough is “plow” and not “pluf”, Through is “threw” and not “thruf” or “thr'ow” (‘ow’ as in ‘ouch’, not as in ‘oh’).

Okay, so that's one exception. Why not explain:

head	rhymes with	bed
heap	rhymes with	seep
lead	rhymes with	proceed
...	as well as with	said

What does “read” rhyme with? Well, that depends upon the tense. If you are reading or about to read then it sounds like “reed” but if it is something you have done, it sounds like “red”, but “red” is a colour and not the past tense of “read”.
“Mean” (m'een) has a past tense form that is different – “meant” (m'ent), and it wasn't until the past tense was mentioned that you knew if I was talking about “mean” as in to intend to do something, or “mean” as in nasty.
Why does “cruel” sound like ‘krool’ and not ‘kroo-el’?

As an English speaker, I may slag off the French language for some weirdness – the pluralisation catches me every time, as does the pronunciation (seriously, does “Coësmes” look like “kwem” to you?) and to me words ending -et and -es and -ey and -é all sound alike. Why do the girls in the supermarket say ‘khr'hem-uh kahr'hamell-uh” which is a hell of a mouthful when “krem kara'mel” is easier. It's not that they are pedantic, they say McDonalds as McDo (the ‘nalds’ bit is just omitted).

So yes, I comment on the French language; but I can never escape the fact that my own native language is full of the seemingly illogical as well. In fact, given the lack of accents, our language might make even *less* sense. My French teacher never got “April” correct (a'pril instead of ay'pril). “apr<vowel>” begins “ay'pr” right? April, apron, apricot? Well, not exactly. Apraxia.
Like I said, English has its own problems.

2004/08/22

:) Go figure! (:

a ramble around Rick's mind...

Well, this issue of *Frobnicate* is unusual for being created on three different computers. An Acorn RiscPC (32Mb, ARM710) was used for the image scanning. Most of this issue was written on an Acorn A4 (4Mb ARM3) laptop. And *this* page? This page has been written on a Windows98 PC using OvationPro for Windows v2.74.
Talk about compatibility!

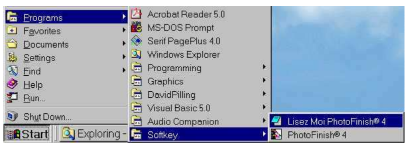
What does now work is the PDA. Well, it sort-of works. You see, on the box it says a 486 with 8Mb RAM and W95 will get you going (but 16Mb RAM and W98 is better). Well, I tried it with *!PC* set to use 24Mb RAM and Windows 95 OSR 2. It was a 33MHz 80486 co-processor. No joy. The next computer I tried was my 'real' PC, a Pentium75 overclocked to 80MHz and 90MHz. Windows 95 OSR 1-and-a-half and 16Mb RAM. Still no joy.

I'm writing this part of *Frobnicate* on a 466MHz Celeron machine running Windows 98 second edition fairly comfortably in 64Mb RAM. The PDA link now (mostly) works. But when it doesn't, it'll just sit and time out like a useless piece of crap instead of doing something worthwhile like saying "sync err #4 pda rec del" which you can translate into "there is a record 'missing' on the PDA and I'm not sure what the hell to do!". Better than 'time out'. And do you know what this Celeron at 466MHz is actually doing? Yeah, you guessed it, sending serial data down the regular serial port with no hardware flow control (it is a three-wire link) at a tedious 9,600bps. My God! I could do that in BASIC on an 8MHz ARM2 *and* make it multitask!

But, on the other hand, this machine can play MP3s (finally!). My RiscPC (ARM710) actually takes almost *exactly* as long to decode MP3s to WAV as it takes to download them in the first place, at about 31kbps. Unfortunately I have a bunch of WAVs on the harddisc. The program that is supposed to make MP3 files actually makes WMAs which is (IIRC) Microsoft's own MP3-alike. It is "supposed" to be better, but since Microsoft is the company that thinks the best way to make a product user friendly is to write "user friendly" on the box...

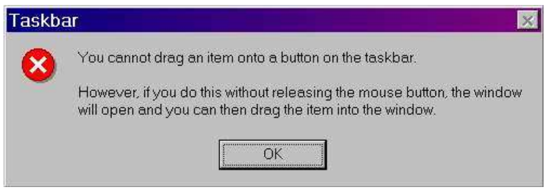
...the whole audio thing is subjective anyway. I know a guy that swears by gold-plated audio plugs. Sounds the same to me, and is possibly a false economy as they don't hook into gold-plated audio sockets and there's copper cable in between those gold plugs; but he swears by them. I'd like to MP3 all of my CDs. Well, all of the tracks I actually like. Then, since this laptop has stereo speakers in the front, I can listen to my music while I program, or write this rubbish. Whatever.

Windows is dangerous. It is official. I said so. :-) There is a logic to the way Windows handles drag and drop. The logic is to do the least expected and potentially most annoying thing by default. Let's take an example. Pictured below, at a nicely rubbish resolution, is the Windows Start menu. We follow through the menu to reveal the "readme" file for PhotoFinish. It is called "lisez moi" as it is in French. Click and hold on that, then drag upwards slightly. Oh! Look! The file has vanished from the menu. It is now on the "Desktop" (pinboard). There was no warning that dragging from the menu would *move* the object and not copy it. In fact, why on earth is it even possible to move things in this way? The right-click-on-Start, Open method exists for modifying the structure of the Start menu. Can we drag the file back? Erm, no. A mouse click closes the menu. We need to either "undo move" then and there, or copy it back the long way. Given that the touchpad on this laptop is sensitive, and two fast clicks could start a copy/move operation, I have to be very careful to ensure my MP3s don't get merged with OvationPro's resources, or silly things like that. A right-click will bring up a "what to do with this object" choices box, like "Move Here", "Copy Here", "Jump on the table and dance the Charleston Here", that sort of thing. There should be an option to make this available by default for *all* copy/move operations.



You can undelete thanks to the Recycle Bin, so why don't we have some protection on copying and moving, now that Microsoft has made it so easy to do those things when you least want...?

But the "task bar" is still *broken*. You can't "install" an application such as Notepad onto the task bar and drag things to it. You can load Notepad and then minimise it so it is just an icon on the taskbar, but dragging files to this icon will cause the following to appear:


Taskbar

You cannot drag an item onto a button on the taskbar.

However, if you do this without releasing the mouse button, the window will open and you can then drag the item into the window.

So let's get this straight. You can drag stuff out of the Start menu which is something you aren't likely to want to do ... you cannot drag a file to an application's icon on the task bar which is something you might think you could do. Rrrrrright.

You have to hand it to Microsoft. The option to "smooth edges of screen fonts" does nothing. Okay, I lie. The large text you see least often – like the current directory name in the MSIE-style file explorer is smoothed. The largish text on the task bar and in the window title bar, and the smaller text used in all the icons... Not smoothed. The standard TrueType® nonsense that looked U.G.L.Y. in Windows95 still looks U.G.L.Y. Perhaps, given the system's insistence on drawing the characters with single-pixel-wide lines, they couldn't figure out a good way to perform the anti-aliasing that we take for granted on RISC OS?

We'll leave Windows alone now. After all, you can't expect them to get it right the umpteenth time. They're only a small organisation with a centre larger than many English towns (and some cities), and a workforce measured in billions... tiny really. They'll get there eventually.

You might wonder where I got this machine. Well, I went to help recover some information from it because the display was broken. Yes, people, I have another PC laptop with a broken display. At least this one has some hope for it! You see, the laptop screen actually works. What has gone is the backlight. I have taken out the backlight exciter circuit and probed it (ooh-err). It *seems* that the main coil is open circuit, but something else may be wrong as no voltage is getting beyond the left inch of the board (it is maybe three inches wide and half an inch deep). Hopefully a replacement will fix the problem. Until then, I have the machine plugged into my 14" monitor. I normally use 800x600 because it causes the least interference to mom's radio. 1024x768 squeals something terrible.
Finally something with a USB port. This was, originally, non-functional but it came back after I reset the CMOS RAM the hard way. Fancy passwords are no match for a screwdriver! Hehe... Anyway, the USB port is useful because all manner of things, these days, connect using USB.

Indeed, pretty much the only thing missing from this machine is 10baseT for ethernet. So I'll quickly ask if anybody has an unwanted PCMCIA ethernet card (10mbit is fine, the rest of my network isn't high tech!), please get in touch!
This laptop claims to accept two type I or type II PCMCIA cards, or one type III card; and/or a ZV card in the upper slot. No idea what that actually means as I've not dealt with pukmukia (PCMCIA) cards before...

This machine has a DVD-ROM drive, and 800x600 is more than enough to watch a movie. I was going to include a screenshot of my watching Nurse Betty while writing this, but it seems the DVD player switches to 256 colours, then pokes the image data (in rather more than 256 colours!) directly into the display memory. All screengrab attempts result in a blank screen. Oh well, you'll just have to take my word for it. I think I'm going to go and brew up some spaghetti and then settle down to watch this movie.

`[hours pass]`

I have tried to make an MP3 of some music I like. This machine has a couple of Gb free on it's harddisc and speakers at the front..why not install some good music and MP3 it? I figure if each MP3 takes about 4Mb, then I can get around two hundred and fifty songs into a gigabyte. It'll sure be better than carrying around large numbers of CDs (between twelve and thirty, at a guess).

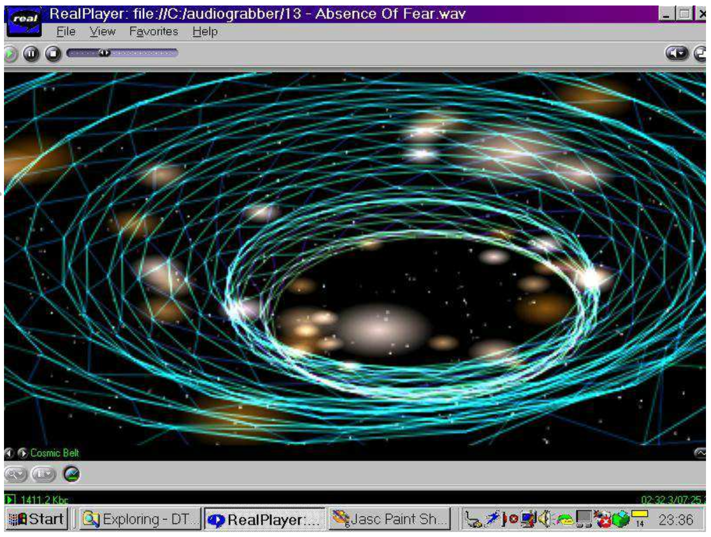
But, problem. Pressing the Play button auto-loads AudioRack32. That doesn't play MP3, I need to use the RealAudio player. I'm sure it can be fixed by poking the system registry, I just can't be bothered right now because...
...problem. The demo software I have been using to extract the songs from CD is called AudioGrabber. It randomly selects, at startup, about half the songs that it'll allow you to manipulate. That's one of the 'annoyances' of it being a demo. I can live with that. What I cannot live with is when I tell it to make an MP3, it makes an WMA What the hell!? How *dare* it claim to make MP3s and then, like, go and make something else. Something that *nothing* on my system will touch. To be fair, Real Player seems to think it knows what it is, but it says I need to upgrade. Seen the file sizes of these "upgrades"?

So I'm listening to a smaller collection of WAVs until I can find an MP3 encoder. Looking in Google, there are a few but at large file size. Good God, can't you show me a DOS executable that'll make an MP3 and can be downloaded in less than a minute? I don't want to have to download 6Mb just for an MP3 encoder!

But, sadly, this bloat seems to be endemic in the Windows world. I have the Learning Edition of Visual Basic. As far as I can figure out, this means I have fewer “controls” available, I can’t make “DLL”s, and *possibly* my compiled EXEs are actually P-code and not native code. It isn’t as if *any* of the documentation I have (about 2,000 pages in paperwork and possibly that much again within CD-ROM books) actually gives a proper run-down of the different versions of VB 5.0.
So I pop over to Microsoft’s web site. I can download an update that makes the whole development suite ‘nicer’, maybe fixes a few bugs. A piddly little 97 Mb.

Anyway, more on VB next time.

For now, back to Real Player. I, as standard, do not like Real because their own formats are proprietary, which means those of us using non-standard computers (i.e. anything that doesn’t run Windows) is not going to have an easy time playing Real files. But I tell you what, though, I could stare at my music playing in Real Player for ages. Because of the *Visualisations*.



One of RealPlayer’s “Visualisations”...

This is the *Cosmic Belt* visualisation. It restfully swirls and tumbles, the diameter of the tube relating to the music. I suspect the colours and movement of the gassy spheres are also controlled by aspects of the music. As I listen, I see the belt swish around the screen with its rotational motion, the spheres bobbing along inside. It’s so beautiful it hurts to look at it.† If it didn’t sound like I was starting to lose my mind, I’d describe the spheres as “benign”. In fact, it is almost scary how you can attribute something akin to a life, perhaps a personality, to a creation that is purely mathematical. It makes you start to wonder – are *we* mathematical algorithms? Sometimes, when I see, like, pieces of mud that has fallen off of a farmer’s plough, I look at them lying on the road. Left, left, centre, left, right. There is a pattern.

†Bonus points if you can identify the misquote...

Our human minds are very good at finding patterns, especially where there aren’t any. But this, left, left, centre, left, right. This is a pattern. Now the question is not “is this a pattern”, but rather “is this a *message*”?

The thing that’ll blow your mind while you are trying to work out if this is any form of communication is “who exactly sent it?”. I know some people who would suggest “little green men” (usually known these days as “greys”, it seems, so I guess they weren’t so green as they were cabbage looking after all!). Okay, fair enough. An alien came down and arranged lumps of mud for me to agonise over. On a scale, that ranks as only slightly more silly than to say “God”, which is the next option on the list. Perhaps God exists, perhaps he/she/it doesn’t. Either way, a “God” of any kind is likely to have more important things to do than to leave lumps of mud in patterns. Besides, the Christian God is apparently more inclined to things you can’t miss; flaming bushes, locusts, flooding the entire planet, that sort of sulk. Not twiddling mud lumps.
Or, in context, it’d be like *you* (you, dear reader) carefully arranging specks of dirt on a clean paving slab as a message for the more inquisitive ants to discover. Can you see yourself doing that, really?

My current theory (of course I have a theory!) is that it *is* a pattern. It *is* intentional, but sadly there is no message in it. The universe is mathematical. Analyse the Golden Mean and the so-called Perfect Number. You’ll find some interesting coincidences. How about if this mud pattern is simply an aliasing problem? You know those funny blocky bits you get in JPEGs when you set the quality a little bit too low (look on the previous page for an example!)? Well what if these patterns were a side effect of that, in terms of our planet?

I’m open to suggestions. It bugs me that I have not figured out how and why we are here. Some take comfort in leaving it all in the hands of a loving father figure. I could shoot that theory down over many pages but I won’t. If you have an idea, no matter how wild, I’d like to hear it. Because I don’t think Left, Left, Centre, Left, Right is an accident.

This is all very far from the subject of Windows, but it is okay, this article is titled “Go figure” which suits this discussion well.

To resume our scheduled Microsoft bashing, I have to ask why Windows 98 seems, half the time, to be so damn glitchy with it’s standby mode. Sometimes it works. Sometimes it doesn’t restart. Sometimes it restarts but you must Ctrl-Alt-Del so it notices all the tasks which are

'Not responding", only as soon as you Cancel they all begin responding.

Why oh why oh why did they swap the messy little INI files for a single point of failure by the name of System Registry? It isn't as if things are suddenly all better now. In fact, it is very much worse. If your registry is corrupt and you cannot restore it from a backup, your only option is to re-install Windows which will create a blank registry. Then, suddenly, you realise half of your applications have forgotten (lost) their settings, and the other half won't run at all. How to fix it is simple. Re-install it *all*. I was talking about this problem to a computer bod not that long ago and he said "format and re-install cures most problems". The worst thing, he said it like it was "common knowledge" and not like he was taking the piddly.

All that, so I can listen to Lene Marlin, huh? :-)

Playing Lene Marlin's CD in a computer.

People that know me know we've recently had a kitten crisis. Our nine-month old (Kibbie) got herself in the family way. All the books said first litters are two or three. Our six-month old (Elsie) wouldn't get pregnant. Too young.

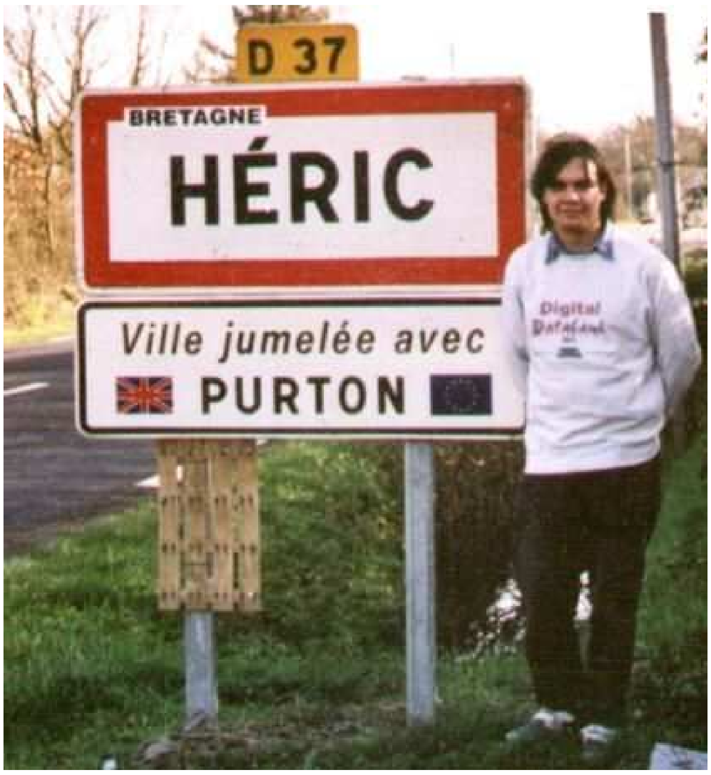
Imagine our surprise when one night Kibbie gave birth to *five*, and lil'Elsie gave birth to *four*! It was okay at first, if a little bit bizarre (more on that another time). The books were wrong, so don't you believe them!

Sadly, though, we seem to be the object of some derision because we take the kittens for a walk.
Yes, and?

My only reply to those sorts of comments is *shame on you* for not being open-minded enough to do things like that yourself. A kitten may be a menace and it may even

be a Weapon of Miniature Destruction, but it *is* a living thing. People take their dogs for walks. We take our kittens for walks. Or would you rather they be ignored, there to pet when *you* feel like petting?

Was out the other day when I saw something on the map. Giving mom (nominated driver) instructions, I simply *had* to have my photo taken by this sign...
(I trust it is obvious *why*!)



And with this eclectic mix of commentary, I'll bring this article to a close.

Until next time, have fun and remember...
...it's only a computer!

2004/10/10

But I'm glad I was there when it happened

by David Norris

Two weeks ago I was sitting here facing my laptop, idly thinking of possibly starting my novel... or maybe even writing someone a letter... or something... When a message suddenly flashed up on the screen – **YOU HAVE PERFORMED AN ILLEGAL ACT**.

It was like being accused of drinking *two* bottles of milk when I was in junior school (I didn't). Like being breathalysed in my Micra (I hadn't). Like being stopped at Customs (I wouldn't). I still blushed.

And I stared.

And as I stared, the screen began to – well, to eat itself. Like those opening sequences in old Ealing films – '***Soon Merrie England began to suffer the insane ravages of one man – psychopathic Duke Nigel...***" – and the explanatory parchment starts to burn up from the corners in.

In an instant my screen had also 'burned up' and apart from a tiny b&w box in the top left-hand corner saying ***REBOOT NOW***, it was black – a total blank. So was I.

Then, suddenly, a female voice announced "***PROBLEMS COMMUNICATING***" in an un-worried, this-is-the-4-minute-warning-but-the-pilot-is-doing-all-he-can tone of voice.

This unconsoling message kept piping up every few seconds – "***PROBLEMS COMMUNICATING***"... It was virtually continuous. By this time ***REBOOT NOW*** had disappeared from the screen and given way to strange momentary images that came and went in flashes. A photo of Hughie Greene... two amorous penguins... the Pope in sunglasses...

Still "***PROBLEMS COMMUNICATING***"... Still "***PROBLEMS COMMUNICATING***"... Then, after the pictures, a succession of unconnected pages of text, apparently snatched from long-ago files and folders... eg "***...Descartes – Usually so rational, he had an almost hysterical phobia when it came to cheese...***" Or a detailed list of the specifications of every ***Ewback*** manufactured between 1927 and the present day... Or a letter (mine ?) offering to adopt a donkey...

But all the time – "***PROBLEMS COMMUNICATING***"...

Desperately I pulled the plug out of the wall. But my laptop just went into battery mode and carried on communicating – problematically.

Now, though, the screen had turned paper white; and for a moment there was nothing on it. Then a small black arrow (my friendly cursor) shimmied into view and started edging across, pointing and heading from left to right. Slowly it reached midway and then – Then it stopped – dead. It turned upwards. Then it turned back in the direction it had started from; then, abruptly, it turned downwards and shot off – into oblivion.

My heart sank with it. This was a significant turning point; something irrevocable has just taken place. Oddly, I felt bereft. Alone. For one thing – mercifully, I suppose – the voice-message had stopped.

But wait a minute – a smell had started.

It began as a sort of small ***reek***, pungent but difficult to identify. Oil ? Metal ? Ashes ? Then – ***phut!*** – a little pale-grey cloud puffed up from the battery side of the keyboard, from inside, somewhere between **W**, **A**, and **S**.

You fight it, but at a certain point you have to admit that the – whatever species of mechanical thing this ***is*** in front of you – that it may actually have gone wrong. This was that moment.

My fingers were still on the keyboard. For some reason I tried to tap out '***I need help***' or '***I surrender***' ... or any cry for help. I don't exactly remember. But the keys felt hot now – very hot. However, hopefully, I pressed '**I**' –

At once flames flickered up from the **ENTER** key – and then from **DELETE** – and then, terribly, from **HOME**. Small flames – but red-and-orange, nasty to the touch, ***real*** flames made of ***real*** fire.

Now look, I've seen pictures of blazing oil fields. I've stood witness at countless Bonfire Nights. I've had my fair share of flaming chip-pans. But this was different.

Somewhere, deep inside this shallow black box – beyond my ken but before my very own eyes, some version of an intelligence was being destroyed. No, it was actually destroying itself. Circuits... connections... systems... logic... memories... all in suicidal meltdown.

Within a couple of minutes I was looking sadly down at a small, smouldering oblong box – a blackened tray, smelling of solder and plastic. Above it, still attached, was a cracked, charred glass screen, now tilting respectfully downwards. As for me, in my hand I held a fire-blanket. I hadn't been able to bring myself – stir myself – to smother my laptop out of its misery in its final moments.

Losing my laptop had been like losing someone special. Someone that doesn't ***seem*** to observe you but is watching you very carefully... Knows you... How you work, how you think... Knows your secrets... your weaknesses... in a way you can't quite ***fathom***... or ***trust***... Clever – yes. Complex – oh yes. Your servant – but really out to control you... Difficult to communicate with... Difficult to like... Difficult to identify...

I called mine ***Blunkett***.

David Norris

Installing a satellite dish on a temporary or semi-permanent basis

by Ewen Cathcart

It *should* go without saying, but Americans may be reading so it'll be said anyway:

This information is provided in good faith and you do these things ***at your own risk***. Okay?

Ed.

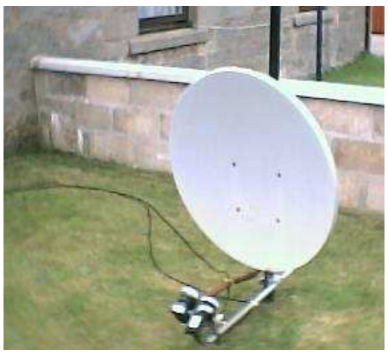
This article concerns the installation of an offset satellite dish as used for reception of television and radio from high-powered broadcast satellites.
My own satellite dish is of the 80 cm solid variety, but it also equally applies to mesh dishes and Sky minidishes.

Original from www.analoguesat.co.uk

There are many solutions to installing a satellite dish on a temporary basis. You can follow Rick's example (top picture) by propping up your dish with heavy objects, such as bricks, concrete blocks, tiles or even fbwerpots! If you have a south-facing wall you can lean the dish against that. The dish could also be placed against a pile of sand or gravel chips. You could even plonk your dish inside the rim of an old tyre! (bottom picture)
{ *Notice the piece of wood holding the dish-in-tyre in place. I'd be tempted to cut notches into the tyre, also, to help keep the dish stable in windy weather. If you look carefully at my ridge-tile arrangement, you will notice the dish fits into the grooves of the narrow end of the tile, to counter slewing that may occur in windy weather. Ed.* }

These basic methods are good way to initially test out the dish and LNB before contemplating a more permanent fi xture.

I have myself tried many of the above. Unfortunately, at this latitude (northern Scotland, roughly on a par with Gothenburg), there is a slight problem: offset dishes that face in the direction of 28.2°E point slightly downwards! Because of this, I am unable to rest the dish on the ground, as the LNB will obviously get in the way.



Ewen Cathcart

{ *The first time Ewen sent me this picture, I was like "dude, what's wrong with your dish!", the curvature of the Earth hadn't occurred to me. Duh! Ed.* }

One novel idea that I didn't try is to turn the dish upside down! In many instances this will allow the (top) rim to be placed on the ground, albeit at an angle. The dish also needs to be tilted back by an amount calculated by adding the offset angle of the dish (roughly 22 degrees) to the elevation angle of the satellite. The further south you go, the more tilt is required. At southerly latitudes of Europe, an upside down dish can almost appear horizontal! At my location the tilt is not too extreme, so it would be quite feasible for me to do it that way. Anyone else might have to look at getting some sort drainage system going – unless they desire a fancy metal bird bath and *no* signal :-)

Some caravaners and travelling folk use a dedicated tripod stand to set up their dish. You can get tripods that will accommodate dish sizes of up to 1.2m.

Original from www.mismatch.co.uk

I have not used one myself, however I have observed a group of travelling folk setting one up. I think you need to splay the legs out as far as they can go to ensure the stability of the dish. There is also the ability to peg the feet down, if the ground allows for this. I gather that dish tripods can be very expensive; often two or three times the cost of the dish itself. You could possibly improvise a tripod by purchasing a rotary washing line (Lidl often have them going cheap) and turning it upside down! { *Far out! Ed.* } The base could be weighted down with bricks/stones/etc. You would also have to take steps to prevent the pole from rotating! I have not tried this method; therefore I have absolutely *no* idea how stable it would be!

If you are seeking a semi-permanent installation you could look out for a metallic "ground spike", such as those used to provide secure anchorage for anything requiring a vertical pole, such as a parasol or indeed a rotary washing line. You either have to drive the spike into the earth with a big mallet or screw it in. Lidl occasionally sell the "screwing" type of spike (and a pretty lethal looking thing it is too – I bet Rick would love to see that being "demonstrated" on The Horror Channel!! { *Mmm, you'd need a wooden one. Ed.* }). Once the spike is in, the idea would be to shove a short length of pole into the socket and attach the dish to that. Adaptors are normally included to accommodate different thicknesses of pole, however I'd recommend using the thickest size possible to minimise any "wobble" in the socket. It also goes without saying that the pole needs to be reasonably plumb, otherwise you might have to compensate by tweaking the LNB skew rather more than would normally be the case. Ideally, the spike should be inserted into hard compacted soil. Unfortunately, the soil in my garden is a bit too soft for the spike to be of any practical use, at least in combination with a large dish.

Ewen Cathcart

You can also obtain patio "ground mounts". These are designed to be fixed to a concrete base, and have a short length of vertical pole. They tend to be more expensive than wall brackets. The Lidl type of bracket is nothing more than a patio mount with a bend in the pole. It might be possible to straighten out the bend using a bit (or possibly a lot) of brute force. Alternatively, saw the curve off and stick a pole in its base.

My own semi-permanent solution is to fix a dish to a washing line pole in the back garden! And why not? It is after all a vertical pole set in concrete! The room where my satellite equipment is located faces away from the main satellites. Were I to affix my dish to this side of the house and aim it towards 28.2°E, it would actually be looking back into the wall. On the other hand, I have a convenient washing line pole with a clear view of the satellites that I need, located straight in front of my window! The dish is bolted as close to the ground as I can get away with in order to minimise the effects of vibration in the wind. There is a washing line running in front of the dish, however this is normally not used for drying clothes. The others lines run to the side of the dish and so can be used for their intended purpose. Cables from the LNBs come into the house through an open window. The window is always left slightly open, although gaps are covered over with insulation during the winter months. I prefer a degree of ventilation anyway, as there is a smoker in the house!

My dish is aimed at 28.2°E. I also have a second LNB on a separate "arm" which is independently attached to the washing line pole with the type of bracket that is normally used to fix TV aerials to masts. The other end of the arm has a DIY holder for the LNB. The LNB receives signals from the Astra 1 constellation of satellites. This is possible because the offset dish produces a series of secondary focal points corresponding to satellites within a limited range. The LNB is positioned on the secondary focal point of 19.2°E, and the signals are fed to a second digital satellite receiver. I have actually had this kind of set up for quite a few years (mainly for analogue reception at 19.2°E and 13.0°E). { *You may have seen these gadgets in magazines for receiving Astra and Hotbird using two LNBs, though now with DiSEqC it is more common to have a lumpy LNB that does both. Ironically the old double-LNB method can be adapted to Astra 1 and Astra 2 as Ewen describes here. The Astra/Hotbird LNB has a fixed offset. Ed.* }

Hopefully, I will have provided a few ideas in this article that you might want to investigate further.

And now for a little something extra...

SAT-HUNTING USING A SIGNAL METER

In the previous issue of *Frobnicate*, Rick told us how to locate 28.2°E using a Digibox. This short article will provide a brief description on how to find any high-powered broadcast satellite using a dedicated satellite meter (also commonly known as a "sat-finder"). For the purpose of this article I will assume that the satellite dish is fixed to a vertical pole, whether it be on the ground, or as part of a wall bracket. I will also be using a basic type of satellite meter that was recently on offer in Lidl UK. The

meter has a 5-LED indicator and an audible tone (the more LEDs that light up and the louder the tone, the stronger the signal), plus a rotary knob for adjusting sensitivity.

This kind of meter is typically a lot less expensive than the traditional "needle" type. You can adapt the advice given here to fit your own circumstances.

First of all, you will need to decide how you are going to power the meter. Of course you can use a satellite receiver to do the job. However, any decent DC power supply capable of delivering 13-18v should be fine. I have a battery pack (10 x AA type) specifically designed for this purpose. I am not too sure how much current your average LNB draws, although in my case the battery pack did provide me with many hours worth of sat finding. Anyway, if you are rigging up your own supply it goes without saying that you need to observe the correct polarity. Also remember that the 22 KHz tone is absent, therefore your LNB will only receive (and respond to) the lower band of frequencies. If you desire both low and high bands (or high band only), then use a receiver with 22 KHz tone-switching. Most new digital and analogue receivers should have this facility.

Before starting anything the dish should be "roughly" pointing in the direction of your intended satellite. As Rick stated in his article, there is no point going for exact measurements, as you are highly unlikely to get it "spot on".
Nuts and bolts should be tightened to the point where the dish is secure but can still be moved by hand.

Make sure the sensitivity knob of the meter is turned so that it is in the least sensitive position. Next, connect up the LNB (using the fly-lead that should come with the unit) and the power supply or receiver. On my signal meter the first LED is always lit (this is normal, as it indicates that the meter is receiving power). Now, adjust the sensitivity knob until the second LED comes on and a tone can be heard. Slowly, move the dish to the left or right until all 5 LEDs light up and the tone is at its loudest. Now, turn the sensitivity down until the second LED comes on (as before) and repeat the procedure.
Eventually, you will come to a point where you will get no more improvement in signal. You can very very slowly move the dish to ensure that the signal is peaking correctly (you may wish to turn the sensitivity up again for this).
Once you are satisfied, tighten the bolts, although please make sure that the process of tightening does not cause the dish to move out of position.
Next, adjust the elevation of the dish. Again, the same procedure needs to be followed, until you can get no more improvement in signal. Now tighten all the bolts carefully. At this point you might want to check the skew of the LNB.
Slowly rotate the LNB left and then right in the holder until you find the best signal. You might need to do this if your dish is not sitting "plumb". Skew shouldn't be too much of a concern, especially if the satellite happens to be lying close to the centre of your particular view of the Clarke Belt.

The above procedure is very similar to the technique for finding a satellite using a needle signal meter. In this case, you start off with the needle in the centre of the scale and move the dish until the needle goes to maximum.
Needle meters tend to be more sensitive and therefore are more suited for homing in on the weaker satellites.

It might be a good idea to use this method of finding the satellite in conjunction with Rick's own Digibox method. That way, not only will you be sure that you are on the correct satellite (!), you will also be able to periodically check the signal strength (independent of the meter) and quality readings. Certain digital satellite receivers (for example, the Comag ones) also have an audible beep that responds to quality level. The louder the beep, the higher the quality. I have tried both signal meter tone and satellite receiver beep together, and with a little bit of trial and error it is quite possible to home in on a satellite by audio means alone!

Ewen Cathcart (2004/10/13)

Ewen wishes to point out that he is not affiliated with Lidl UK. He then sent me a scan of an advert for their Comag receiver and the various other digital reception stuff. Damn! I'm not surprised he mentioned Lidl a few times. If I was in England, I'd have picked up some of that kit too...
Check out, also:

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

Twin Peaks

This article is not about weird goings on in the forest, examined in minute detail by an FBI agent weirder than the assortment of kooks “in town”. Shame, it was a great series.

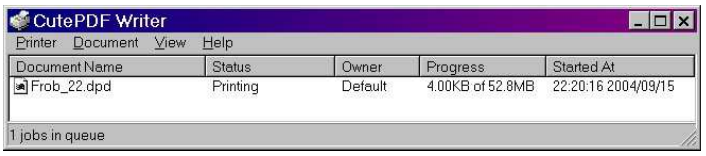
Instead, this article is about the two different *OvationPro*’s. “Twin Towers” would have been too much like jumping on an old bandwagon, and Lynch has more style anyway...

This issue began life on the RiscPC. Gavin suggested I look into Lua and I suggested he write an article comparing it with, say, BASIC. I choose BASIC as the comparison language due to it’s accessibility in Acorn machines (many have C, *everybody* has BASIC...) which wasn’t my preference as Lua has much more in common with C, but it may open the eyes of those who’ve fi gured C is just too much like punctuation abuse.

Due to a rather large electricity bill, mom suggested I cut back on my computer use, at which point I freaked. It is bad enough having *six* modems and no phone line! John Williams came to the rescue and loaned me his A4 laptop. The laptop is an interesting beast indeed, maybe I’ll talk about this next time.
Anyway, I pulled the 80Mb harddisc out of an old (dead) Toshiba laptop and fi tted it into the A4. John hasn’t said whether or not it was okay to format his harddisc, so I took it out and put in my own drive – better safe than sorry! I copied across the important stuff: the C compiler, *OvationPro*, and some other stuff. It *just* fi t, and I set about writing an article for Archive magazine. Then, to my amazement, I was able to tweak the system to get *OvationPro* running (with the nested Wimp) and if I dispensed with the spell-check, there’s around 400K to work in. More than enough to write most of an issue of *Frobnicate*.
As I mentioned before, I went to pull some important data from a PC laptop because the screen had packed in. After doing so, the owner kindly gave me the machine as it wasn’t much use to him what with no screen to look at. Besides, he had a new PC which no doubt operated a heck of a lot faster. The screen problem appears to be a blown coil in the backlight circuit, so I have hooked it up to my 14” CRT. Because of mom’s radio (and my eyes), I work in 800x600 though this machine can do better – the LCD is 1024x768. The modem on this laptop is also not working, I had the clip-in card out earlier and traced the telephone connection back to the coil though ironically it seems to be the modem side that is open circuit. Perhaps there is something ‘weird’ with this machine and wire coils? Maybe it is haunted? Cool.

So here I sit, in front of Windows 98, typing into *OvationPro* to tidy up and fi nish off this issue.

It has been a bizarre experience. For the most part *OvationPro* works as it should, though I have a tendency to close it down when I don’t mean to because of Windows’ inability to ‘install’ applications like on RISC OS. If you want *OvationPro* running, it’ll need to have an *OvationPro* window someplace (even if minimised). The next problem is the font mapping. Very very subtle differences make a lot of trimming and tidying possible. Sadly, it appears *OvationPro* for Windows doesn’t support arbitrary amounts of leading, only single space and double space. This is a great shame as a trick I often employ to squeeze in an extra line or two is to drop the leading to something like 15% (single space is 20%). You won’t see it unless you measure it with a ruler, but it buys you space without your needing to mess with the text size. Another problem is the printing. Those cute ‘fl’ ligatures don’t come out right in PostScript/PDF. They are not part of the standard Windows character set, they reside in some other mapping with a name such as “character-like symbols”. I guess David uses UniCode to access them? Bizarrely, trying to make a PDF of this issue created a *huge* spool fi le! Even worse, trying to PDF one of my poetry booklets caused a crash when the spool fi le filled the harddisc (two and a half gigabytes!)!



Document Name	Status	Owner	Progress	Started At
Frob_22.dpd	Printing	Default	4.00KB of 52.8MB	22:20:16 2004/09/15

1 jobs in queue

Finally, there does not appear to be a way to modify the printer’s margins in Windows. The default margin is slightly smaller than *Frobnicate*’s pages (and the same with the Stylus Color 640 – which works fi ne under RISC OS with larger margins). But these are hardly *OvationPro*’s problems...

No, there are a few minor niggles but on the whole *OvationPro* performs admirably. I had barely used it under Windows before now (why bother running it under *!PC* when I could run it natively?) but now I’m using this laptop I decided to give it a trial run, and it has worked nicely.

But that isn’t the nicest thing. The nicest things is that I can copy the *OvationPro* document *back* to RISC OS and make a PDF that way. Then I know my ‘ﬀ ligatures will look like ‘ﬀ ligatures and not like a ‘•’ or somesuch (this, I feel, is a Windows/PostScript issue as a plain print works).

That is the great thing about *OvationPro*. You don’t have to discard your old RISC OS documents to make the move to Windows. They’re available right away. The user interface is a Windowsifi ed version of the RISC OS version so you’ll feel right at home. Some keys are different (^F to Find, not F4) but I’m sure this can be remapped if you really hate ^F (though many many Windows applications use this keypress). And as I mentioned before, *OvationPro for Windows* is pretty damn slick!

The feature I loathe most:

> The horrible font rendering at certain sizes (this is out of David’s control, it is yet *another* thing Windows can’t do correctly!) [spot a trend? much is due to Win’s own naffness]

The feature I love most:

> That it exists. Oh, feature? Well, aside from Windowsisms like the menu bar and a few different keypresses, it looks and feels like the RISC OS version. Also, this is serious s**t for serious people. It opens with a blank window. Don’t let that fool you. This software assumes that you have a mind of your own and that you don’t need to refer to a “wizard” in order to do anything. And, gloriously, no stupid paperclip!!!

If I ever meet David in person, he’s got a drink on me. *OvationPro* for RISC OS rocks, and *OvationPro for Windows* keeps on rocking.
Oh yes.

Finally it remains for me to thank Gavin Wraith for his detailed introduction to the Lua language – give it a try. Thanks also to Ewen Cathcart for showing us some interesting ideas for mounting satellite dishes (I rather like the upturned washing line!). Finally, thanks to David Norris for an amazingly acute and descriptive examination of the demise of a computer. I’d better hide it from my machines in case *they* get ideas. Yes, computers do seem to attain personalities don’t they? Thank you everybody for your contributions.

Rick, 2004/10/14

The Wrap Party

Yet another issue of *Frobnicate*. It has taken me (and Gavin, David, and Ewen!) longer to write it than it has taken you to read it. I hope you appreciate our tortuous efforts!

Hehe, no, seriously, the more alert of you may have noticed that this issue uses a redirect to download. I am logging your IP addresses as you download this issue so I can get an idea of what my readership *actually* is. You see, a very small number of people emailed me comments about issue 20 and a small number of *different* people talked to me about issue 21. So how many downloads does *Frobnicate* actually have? This is an attempt to figure that out. Am I writing to six people or sixty? It's just a shame that *so few of you bothered to reply to the survey* in the last issue. Yes, I am trying to make you feel guilty about that. :-)

Well, what witicisms to write in this article? I've recently been listening to a news channel and it seems Kerry's idea of an energy policy is to look for green fuel and give incentives to buy hybrid cars. Bush, meanwhile, thinks it is a better policy to make more nuclear power stations and increase the legislation to limit the liability of the operators in case of catastrophic failure, and for the oil situation? No problems. The mighty Americans won't be held ransom to a bunch of Arabs so instead all the wildlife reserves in Alaska will be converted to fields full of nodding donkeys. I guess that last point is an implicit admission that they royally cocked up in Iraq. The country is full of oil, their prime goal (Prime Directive?) was to get their mitts on the oil (under some half-assed "Weapons Of Not-quite-massive-but-large-enough-to-scare-the-sheep Destruction" plot) and after their carefully planned Shock And Awe exercise (which seemed to consist of a hundred tanks marching over *yet another* bunch of desert), the only Shock and Awe that we are left with is the scale of the mess that has been made, which is awesomely shocking.

Need I think of a witicism? Really? Just *look* at the disparity in the proposed energy policy! Perhaps somebody should get Bush to sit down and watch "*The China Syndrome*" some time, but maybe the plot line (shoddy work = big bang potential) would be too much for him to grasp.

Perhaps we should have Teressa Heinz-Kerry run for President? Not only is she better looking than her husband (wouldn't take much), but she must surely be a good person if the best insult the media can dig up is that well-exercised (bored already) "shove it" comment. The only thing is ... she's a rich girl from B*ah*ston (that talks a *lot*) so it is difficult for her to pronounce her name like everybody else... but then again, the illustrious Mr. Powell pronounced his name like that bit of your digestion below the stomach.

Well, this is being written mid-August after watching one of the most bizarre thunderstorms I've seen – just don't mention it to the unfortunate people of Boscastle in Cornwall. With any luck the Americans have kicked Bush out, but that would require the mass population of that country to exercise an *iota* of common sense, and... well... "American" and "common sense" are only things you'd find in a sentence together when it is being disparaging. :-)

This issue of *Frobnicate* is intended to be released on the 31st of October (or as close as possible). Hallowe'en. The fire festival of Samhain. I've heard that said as everything from "sam-hayn" to "so-ween". I, personally, prefer "so-ween" because it is the closest in pronunciation to Hallowe'en. If you can imagine when the Christians 'stole' the festival, they'd have just shoved "hallowed" in front of the name to make it, you know, all religious like, and tell us "e'en" means "evening". Obliterate the old...

This is, traditionally, the time when the barriers between the 'real' world and the 'underworld' are at their weakest and things (both good and evil) could cross over with only a little bit of help from this side (which just screams of "let's pretend we humans are important"). Non-witches used to lock up their houses and carve scary faces in pumpkins, lit by candles inside, to frighten the evil things away. I guess somebody eventually figured out that the sort of evil spirit that takes nubile virgins as a sacrifice is *unlikely* to be worried about pumpkin faces, so this is all now a child's game. Real witches, on the other hand, sometimes use the time to try to contact dead relatives and the like. It isn't a time for fear and worry. Others, meanwhile, take a more Buffy attitude and regard Hallowe'en with a measure of disgust (the Buffy demons avoid Hallowe'en because it is acutely embarrassing!). Yet others go as far as to think "hey, they're dead, let 'em rest in piece!".

It is interesting to read in a witchy text: "*Of all the eight festivals, this is the one where the Book of Shadows insists most emphatically on the Great Rite*". Firstly, there is no "Book of Shadows". You cannot go to WHSmith and order it. What you *can* find is the "Book of Shadows". Confused? Each witch is supposed to have this book, either started by themselves or passed down through generations. It is like a sort of magical diary. What happened, what worked, what didn't. It is like the recipe books of old where people devised their own 'family' recipes and passed the knowledge on. *That* is exactly the same as the Book of Shadows (some call it a "Grimoire", some of the more clueless call their *cat* Grimoire!). Each witch (or family) will have a book that is different from any other. There is no standard "Book of Shadows". The ones you'll find in bookshops are just somebody cashing in on the vaguely-recent "Wicca" craze.

The next point is the one that causes the most contention. "The Great Rite". This means the act of sexual intercourse within the sacred space of the Circle; which can be symbolic (bless the wine, etc) or physical (grunt grunt, etc). Some covens believe wholly in doing this while some witches work in solitary (known as a "hedge witch") in order to escape this 'nonsense'.

While I am here giving you a potted lesson to witchy things, I'll point out that witches can be female *or* male. A "warlock" is an evil practicer of The Craft, to call a male witch a warlock is quite insulting (unless he's into Black Magic, I guess...). Witches cast spells but they hold no special power over people, they cannot turn people into frogs. It could be said that the Christians are exactly the same, the difference being that they refer to their 'rites' as Mass and Prayer. It's just a different way of looking at the same lit candle.

So what does this mean for *you*?

Well, the way I see it, *Samhain* is like Christmas. It is a good excuse to buy more chocolate than you can afford and to pack it all into yourself in one go. After all, that and your birthday add up to three times in a *year*. Surely you can excuse the weight-watching for three days out of three hundred and sixty five (and a quarter)?

If chocolate excess isn't your fancy, then you can always have fun carving cute faces in pumpkins. Or maybe the silhouette of a cannabis leaf? The CND logo? Be inventive! Imagine something!

Either way, have yourself a whole heap of fun and
Happy Hallowe'en!!!

Rick
2004/10/18