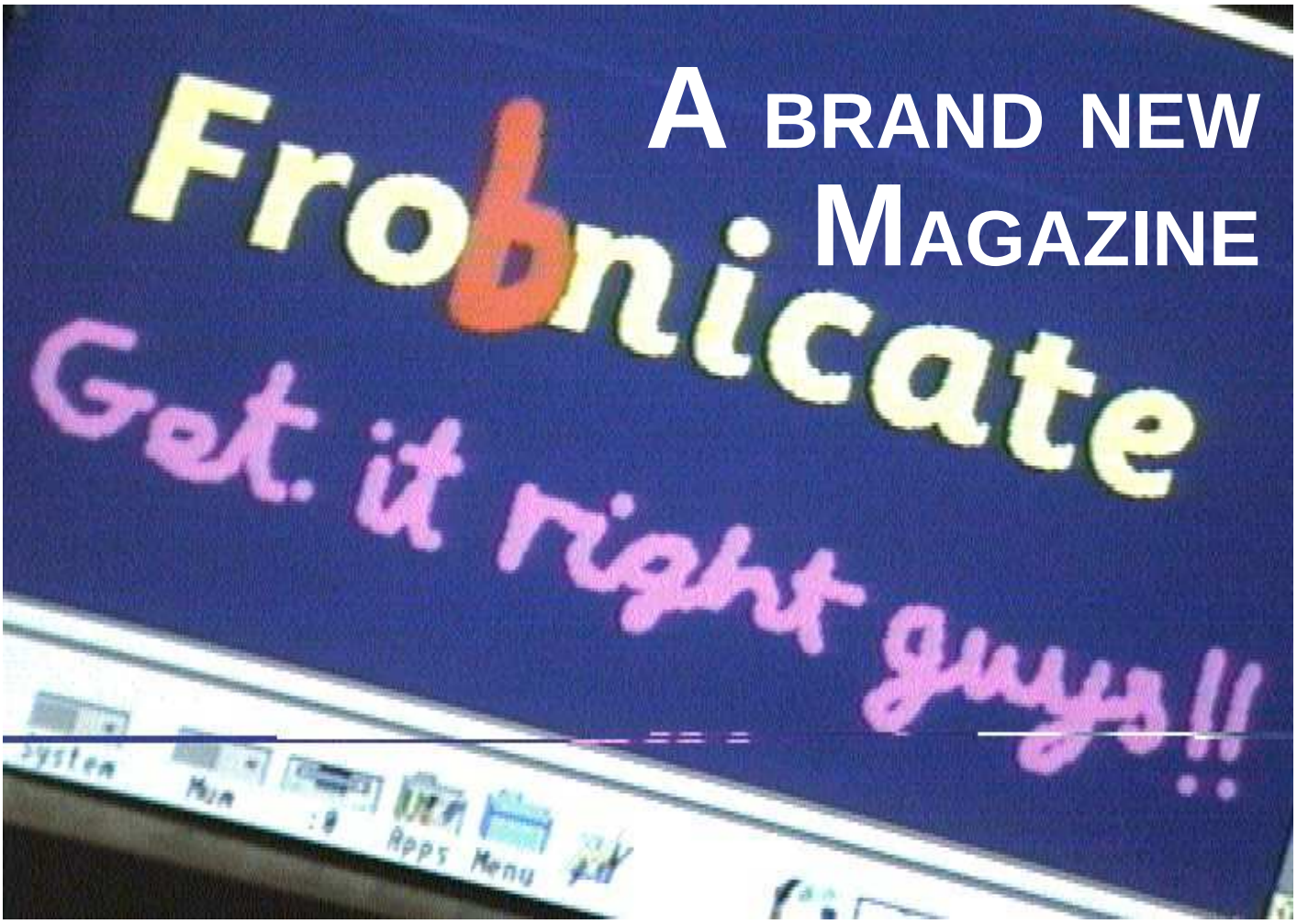
FROBNICATE...

JUNE
1995

Issue 1. The ULTIMATE techie magazine for Acorn Enthusiasts.



In this issue:

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

Published by Your Printer

Index:

Distribution stuff:

Editor . . . . . Richard Murray
Contributors . . Richard Murray
Graphics . . . . Richard Murray

This magazine is distributed locally in hardcopy form and widely on Arcade, ArcTic and Digital Databank BBSs as archived files. You may print the files UNALTERED.
Back issues are available from Encina BBS, as are stylesheets, fonts and logos/graphics.

The editor can be contacted by fido netmail as "Richard Murray" at 2:254/86.1. Feel free to comment or add your own submissions.

HELLO!

Hello and welcome to Frobnicate.

Now what. Erm... Erm...

I suppose I'd best say a few things about Frobnicate and what it is supposed to be.

Right, if you want a nice glossy quality production, then I suggest you take a look at Acorn User. This magazine, however, is a user-creation. That means you are all expected to chip in. It will cover all the stuff Acorn User does not - politics; comms in depth and techie stuff. At this time, I'm not fully sure what will be in this issue, let alone other issues! But a rough look ahead at things I'd like to put in:

RiscUnix, audio mods, speeding up an A5000, RiscPC internals, A5000 internals (techie), DIY serial port for A3000.

Hmm... And more. If you can help, feel free. Details on page 15.

I'd like to offer my commiserations to Ian Kershaw who was planning a production like this, but unfortunately did not receive enough support. I, to date, haven't yet had any real support - but that hasn't stopped me. I hope you will all chip in for next month.

Right. That's filled one column.

Hmm...

I think it is only fair to start at the beginning. Way back in 1987, eight years ago (wow), I used a Master Compact and a word processor called EdWord to churn out weekly episodes of the saga of HappyHak. HappyHak is a character created by Yours Truly, a bit like a cross between WarGames and Pump Up The Volume in character. The teachers loved it...not!

After a while I mellowed out (or ran out of ideas, whichever you prefer) and used AcornDTP and my new A3000 to create a nicer weekly newspaper for my boarding school. It was still a very much 'by myself' production as everybody that promised an article either forgot or handed in their texts a few weeks late. Add to that, most people hadn't grasped the concept of the floppy so most articles had to be retyped. As you can imagine, after 5 weeks I got fed up.

In my final term I got myself Ovation and turned out a computer related magazine for my "computer club", MurrArc.

That persisted for a good 20 issues up until 1993. By them, I'd been away from boarding school for three years and everybody had, sadly, gone there own separate ways. I've spend the last year writing fiction, bringing back HappyHak as a more realistic (!) person in my story The Pupil From Hell (But... part 1).

But I have always wanted to make a more serious attempt at writing something, so up from the proverbial ashes shall rise Frobnicate.

At least - that's how it is supposed to go.

But I'll need YOUR help!

Build yourself a modem to modem link.

This little circuit presented here will allow you to hook two modems together without needing expensive equipment or two telephone lines, which can be just as expensive.

The circuit presented is not ideal. It can be a bit noisy and lossy - but in some ways this is good because it will provide a better emulation of your real telephone line. :-)

Unlike some designs, this one uses a minimum of components. I've not provided a PCB layout. That's up to you. I myself bought two line sockets and superglued them back-to-back, with this circuit (on a spare piece of veroboard) wedged inside.

In the picture on the lower left, you can see the circuit roughly shoved together. Yes, it will work with voice telephones too.

As you can 'blow' your modem from feeding it the wrong kind of signals, I suggest you do like I did and test it with two old GPO/BT rotary dial phones. You can pick these up from parts auctions or your local tip for around 50p to £2 each. They're cool to play with too. Most come complete with a circuit diagram. Ahhh, it was simple in those days. :-)

Telephones are DC. The voltage can swap around depending on various conditions, though this is transparent to the telephone itself.

Floating on-hook line voltage is about 48VDC, and off-hook, the voltage drops to about 5VDC. This circuit provides the 5VDC as a current source. The two resistors limit the current and provide power. The capacitor will pass the AC waveform of the speech/data 'on the line', but not pass the DC.

When the phone rings, there is about 90VAC present on the line. The frequency is about 40Hz.

It is advisable to use a fairly meaty rechargeable battery for performance. I have had this working off a PP3, but the battery didn't last that long.

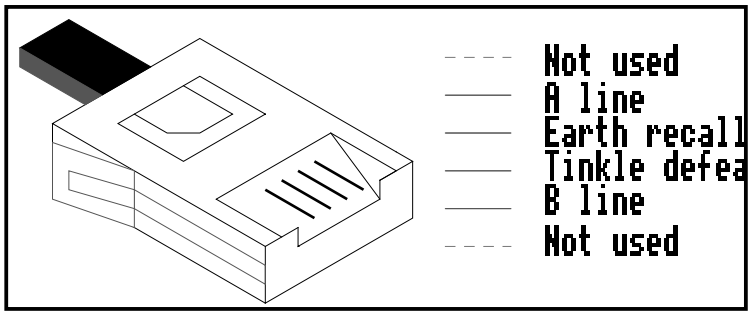
As you can see from above, it is advisable to hook up lots of monitoring whilst setting up this circuit. Some component values may need to be changed, but they *should* be okay.

How to wire up:

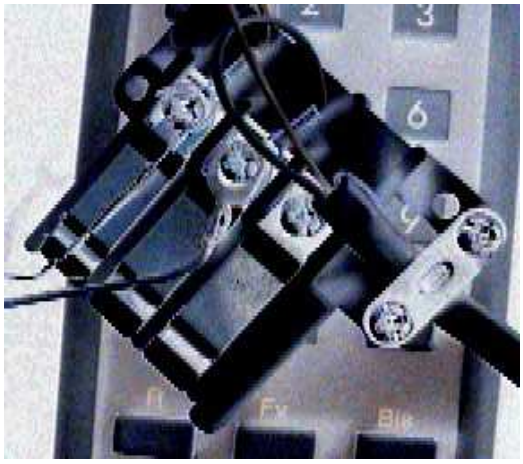
This is given a Maplin difficulty rating of 1. Stick the battery across the battery terminals. The + and - polarity isn't vastly important. Then wire up A to pin 2 of the phone plug, and B to pin 5 of the plug. Again, getting this right isn't vastly important. As long as it is roughly correct, pick up the handsets on the two test phones. If something smokes, rip out the battery, throw this magazine in the bin and give up all hope of understanding anything even slightly technical.

If all seems okay, hold the speaker on one handset to the microphone of the other. You should hear a loud whistling. If you don't, place one handset to your head whilst tapping the other on a nearby table.
If you still hear nothing, check the voltage across both A/B's and the capacitor. Again, if you stick this circuit together as shown, it should work first time.

Next, leave the two phones off-hook for a while. Test the two phones, is anything hot? Do the previous tests still work? Is the voltage fluctuating? If all is still okay, then you can hook in your two modems.



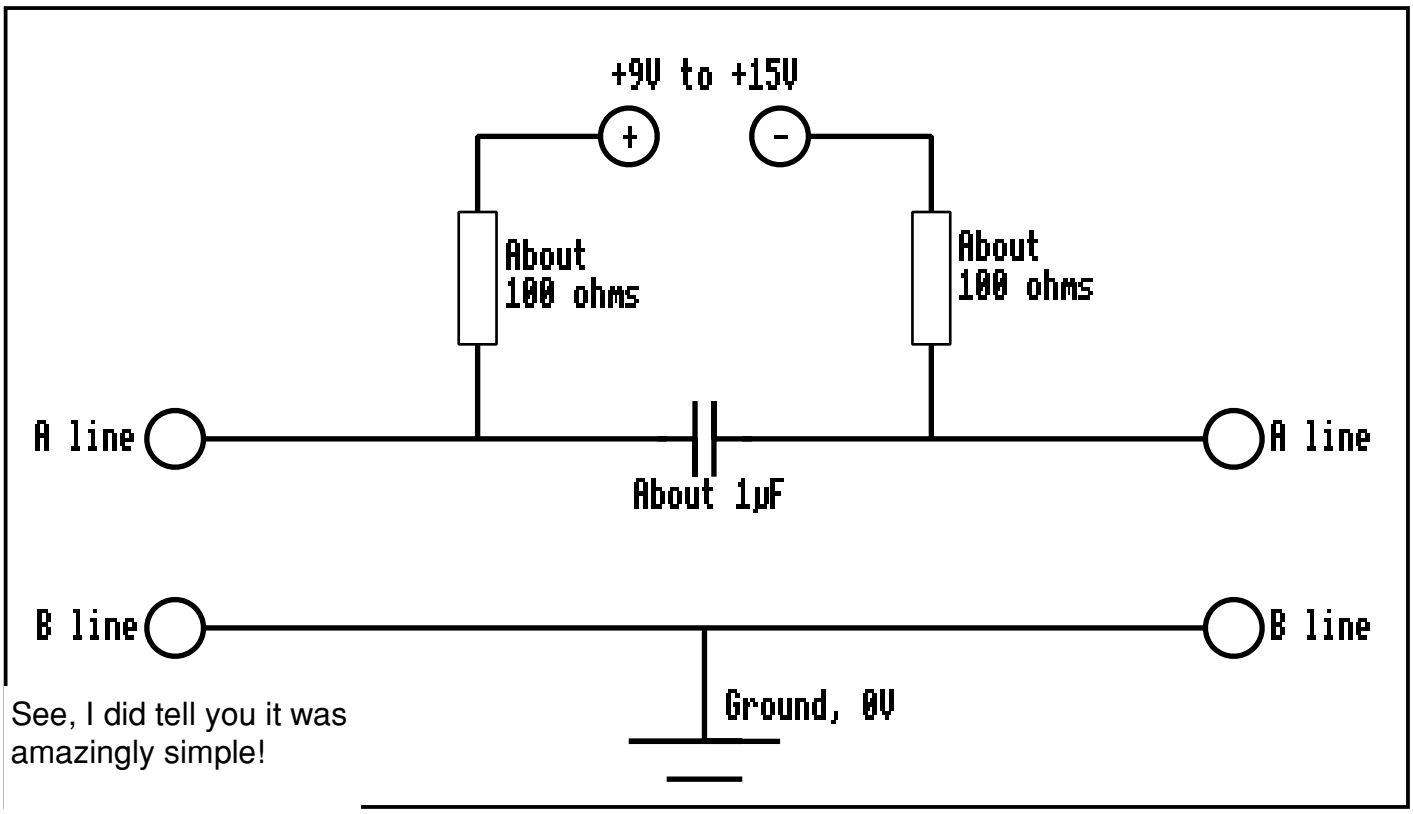
There are six grooves on all phone plugs, but usually only four metal contacts. Use the outer two, on a four-contact plug. Sockets are marked 1, 2, 3, 4, 5 and 6. Use 2 and 5.



In the image above, you can see the two telephones hardwired together on a France Telecom phone plug.

To make a connection between the modems, you will need to follow to on-hook connect procedure. This usually means typing ATA on one modem and ATO on the other. Using this technique, I have gotten MTerm (on the A3000) to connect to ArcBBS (on the A5000).

You even could hook a fax modem to a facsimile machine for a quick'n'simple scanner. :-)



See, I did tell you it was amazingly simple!

What you'll need:

2 x 100Ω resistor, ¼W.

1 x 1µF non-polarised capacitor.

1 x Power source.

Some wire.

You may also need:

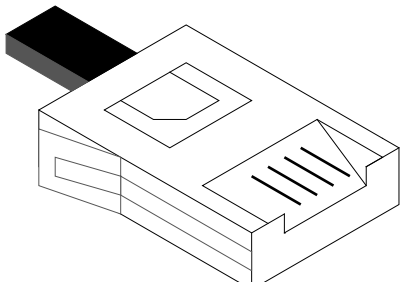
2 x Phone sockets.

2 x Crimp sockets for battery terminals.

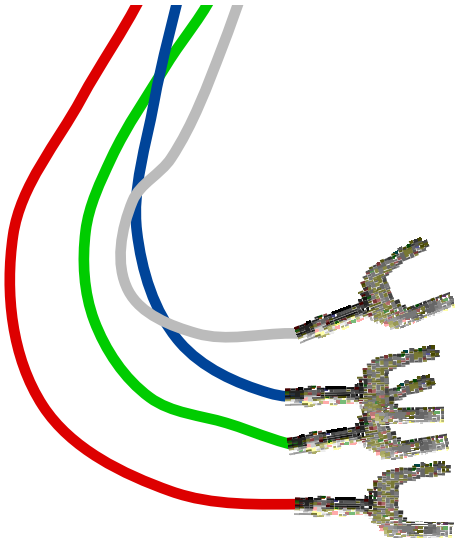
Remember, you do this at your own risk!

Richard Murray.
14th May 1995 @ 18:53

The British Phone system enters your house on two wires, the ***A*** and the ***B***. From your Master LineBox (or equivalent), this is split into three wires. The third wire is the ***Ringer***. This carries the AC ring signal, seperated from the ***A*** and ***B***. The fourth wire is no longer used. The outer two wires are undefined, hence many telephone plugs don't even have them. Some do, they are often used in data transmission between a PABX and a 'featurephone'. They are also a good way to bug somebody. Stick a tiny microphone in a phone socket, hook the mic up to the other wires (1 and 6) and tap the signal off somewhere down the line. :-)

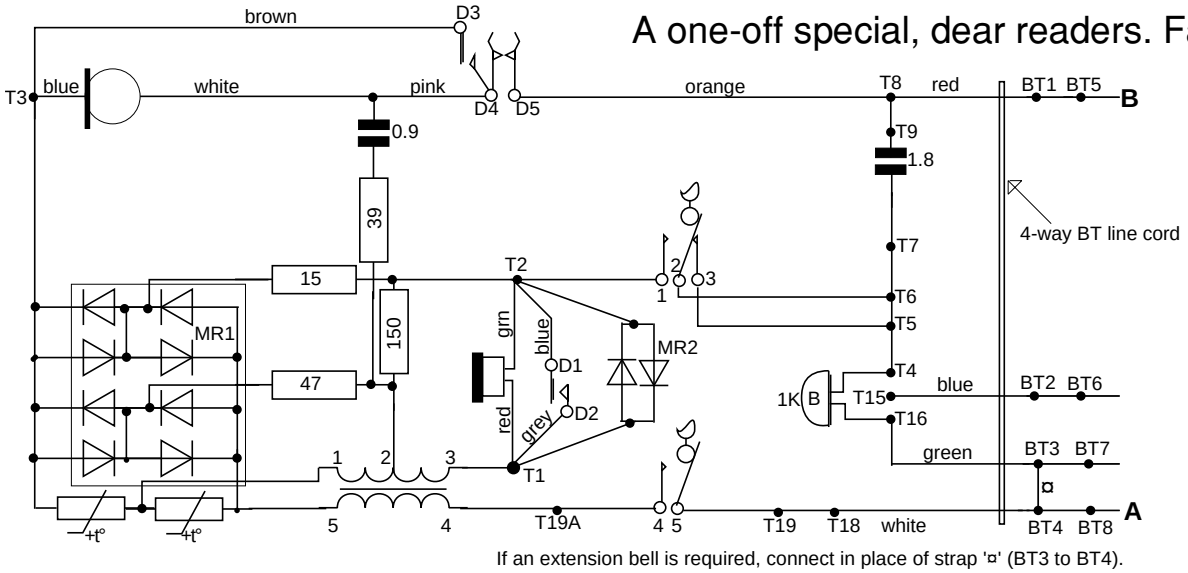


Here is that BT plug again.



If you are using an old GPO/BT rotary-dial telephone for test purposes - here is how to wire that up; followed by the 'standard' domestic cable colours:

White wire	connector 5
Blue wire	connector 3
Green wire	connector 4
Red wire	connector 2
Green, white rings	connector 1
Blue, white rings	connector 2
Orange, white rings	connector 3;
White, orange rings	connector 4.
White, blue rings	connector 5.
White, green rings	connector 6.



A one-off special, dear readers. Fancy building a telephone? ;-)

If an extension bell is required, connect in place of strap '¤' (BT3 to BT4).

Our routine is:

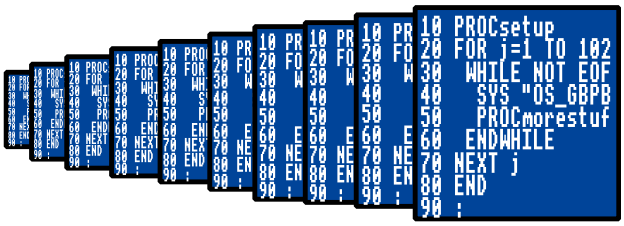
MDMALLOC A multi-dimensional malloc()

It will compile and link with Acorn C release 4 in native (Acorn) mode.

```
/*
 * MDMALLOC - Multi-Dimensional malloc()
 *
 * Allocates a multidimensional array dynamically, at runtime, so that
 *   1: its elements can be accessed using multiple indirection
 *   2: it can be deallocated using a call to the standard free() function
 * Note: On PC's the max array size is 64K
 *
 * Paul Schlyter, 1992-02-09.  Released to the public domain.
 *
 * Converted by Richard Murray, April 1995 for Frobnicate.
 *
 */

#include <stdlib.h>
#include <stdarg.h>
#include <string.h>

#define MAXDIMS  5              /* Defines the maximum number of dimensions */
#define MAXSIZE  ((size_t) -1L) /* Maximum size of array */
#define TEST     1              /* Compile in the test program routine */

void *mdmalloc( int esiz, void *initval, int dims, ... )
/*
 *  Input:   esiz     size of each array elements, as given by sizeof
 *           initval  pointer to initial value. NULL ==> zero fill
 *           dims     number of dimensions: 1..MAXDIMS (5)
 *           ...      number of elements in each dimension (int's)
 *
 *  Returns:  NULL    error: out of memory, or illegal parameters
 *                    otherwise base pointer to array
 */

{
      unsigned int dim[MAXDIMS], accdim[MAXDIMS];
      va_list ap;
      int i, j;
      long int totsiz;
      void **q;
      char *p, *r, *s;

      if (dims < 1  ||  dims > MAXDIMS)
            return NULL;

      memset(dim, 0, sizeof(dim));          /* Read dimension numbers */
      memset(accdim, 0, sizeof(accdim));
      va_start(ap, dims);
      dim[0] = accdim[0] = va_arg(ap,int);
      for (i = 1; i < dims; i++)
      {
            dim[i] = va_arg(ap,int);
            accdim[i] = accdim[i-1] * dim[i];
      }
      va_end(ap);
```

```
                                        /* Compute total array size */
      totsiz = esiz * accdim[dims-1];   /* Data size */

      for (i = 0; i < dims - 1; i++ )   /* Add space for pointers */
            totsiz += sizeof(void *) * accdim[i];

      if (totsiz > MAXSIZE)             /* Exit if totsiz too large */
            return NULL;

      p = malloc((size_t) totsiz);      /* Allocate memory */
      if (p == NULL)                    /* Out-of-memory   */
            return NULL;
      memset(p, 0, (unsigned int) totsiz);  /* Zero out allocated memory */
      q = (void **) p;

      if (dims == 1)
            r = (char *) q + esiz * accdim[0];

      for (i = 1; i < dims; i++)        /* Fill in pointers */
      {
            int siz;
            int accd = accdim[i-1], d = dim[i];

            siz =  i == dims-1 ? esiz : sizeof(void *);

            r = (char *) q + sizeof(void *) * accd;
            for (j = 0; j < accd; j++)
            {
                  *q++ = r;
                  r += siz * d;
            }
      }

      if (initval != NULL)
      {
            for (s = (char *) q; s < r; s += esiz)
                  memcpy(s, initval, esiz);
      }

      return p;

}  /* mdmalloc */


#ifdef TEST   /* Test program */
#include <stdio.h>
main()
{
      static char init_d[8] = { 0x01,0x23,0x45,0x67,0x89,0xAB,0xCD,0xEF };
      int init_i = 0x1111;
      double *a   = mdmalloc( sizeof(double), init_d, 1, 4 );
      double **b  = mdmalloc( sizeof(double), init_d, 2, 4, 5 );
      double ***c = mdmalloc( sizeof(double), init_d, 3, 4, 5, 6 );
      int ***d = mdmalloc( sizeof(int), &init_i, 3, 4, 5, 6 );
      int i, j, k;

      for (i = 0; i < 4; i++)
            for (j = 0; j < 5; j++ )
                  for (k = 0; k < 6; k++ )
                        d[i][j][k] = (i * 256) + (j * 16) + k;

      a = a, b = b, c = c;

      return 0;
}
#endif
```

Next time - 'fuzzy logic' string search routines...



Hacking BASIC

One of the delights of BASIC, to a hacker, is that it is an interpreted language.

Whilst a programmer may delight you with numerous bells and whistles - the source code *is* needed, and will always be present.

Firstly, we shall take a look at the various encryption methods available:

1: Converting to application.

This is a rather common method of locking up a BASIC program. the main advantage here isn't speed - indeed the 'app' part is only a little bit to load in the code and sling it to BASIC. No, the main advantage is you can run Squeeze on it, which will provide runtime decompression of 12-bit LZW. This can, in some cases, give up to 50% savings in storage size.

2: Crunching/Compressing.

This method is, in my opinion, a bit nasty. You can load the code right into !Edit, but all the variables have been renamed, SYS calls are now SWI numbers and so on. This is performed because it is possible to crunch a 200% speed increase out of a program (though most will have to make do with approx. 70%!).

3. Other.

Other. Very general category that. It pertains mainly to encryption techniques, like my own system (created by Ricky Sarge), or AutoRun etc etc...

These encryptions, whilst good and useful for locking up a program from novices, can be smashed in 30 seconds [Robin Abecasis claims 5, and I think I believe him :-)].

EXAMINING THE CODE

The first step is to examine the code. Because most people 'app' there code (encryptors 'app' it too), there is a possibility that your program may not be BASIC at all!

1. Are there any libraries?

A dead giveaway is the presence of a BASIC library file (!Imagery, !SnakeDoor). A C app can't call a BASIC library directly, and a hand-crafted ARM code app won't call BASIC on principle. Be sure it is a library (a collection of functions/procedures) and not some support program.

2. Does it say "Shared C Library" anywhere in the jumble you are looking at? If so, it may be a C program.

3. Does the very end of the stuff you are looking at read something like "rcc 4.00"? (the numbers will usually be between 3.01 and 5.40). If so, unsqueeze the application and try again! Unsqueeze is supplied in the !Patches directory of the RiscOS3.10 Support disc. It is a module.

4. Can you find "Basic -Quit" in the stuff you are looking at? If you can - you've got life easy.

POST-HACK PROTECTION

Some programmers will add post-hacking protection. One of the most common is to get the program to access something set up by the app loader, something that would be lost when you strip off the app loader. This can be a preset value, or maybe a CRC check?

A real pain-in-the-butt programmer could stick the "Wimp_Initialise" in the header and the rest in BASIC - but I've yet to see this.

Turn over....

A hackin' we shall go!!!

A HACKING WE SHALL GO!!!!!

Here are some methods:

1. If you find "Basic -Quit", change it to "Basic -Load". If all goes well, a command window will pop up, and you should be able to type "LIST" and "SAVE". Hey presto!

2. If you find "%Basic -Quit", you can use the technique above, but forget the simple route of aliasing 'Basic'.

3. Also forget using *Save at quit or !Zap to read the app's workspace. Too much hassle.

4. A way that is almost foolproof is to call the application (if a Wimp app) outside of the Wimp. It should crash upon "Wimp_Initialise" with the error "Wimp already running" or similar. Some, like SysOpChat, check the value of Wimp$State, and do nasty things (like erasing itself?) if Wimp$State is 'commands'. Try using the TaskWindow.

AND FOR THAT *REALLY* HARD-TO-HACK ONE.

Those simple methods will get you into most programs. Sadly, however, some still won't budge. A great example here is !Imagery.

Case history

Used to be written in BASIC. Can't imagine a total rewrite. Some parts too slow to be assembler. The 'Overlays' subdir contains a bunch of BASIC function files (libraries).
Has a fancy loader that calls a non-existant * command.

WHATEVER NEXT?

The loader was called with "Basic -Chain" in the !Run file, and "STOP" was inserted in various places. Here are the last few lines of the loader:

```
  OSCLI("LOAD <Imagery$Dir>.Load"+STR$~DAT%)
  OSCLI("LOAD <Imagery$Dir>.!CodeImage"+STR$~
(DAT%+LEX%))
  A%=DAT%+LEX%:B%=DAT%+LEX%+REX%:CALL DAT%
  *Imagery
```

As you can see, it is a bit odd - but there is BASIC there somewhere.

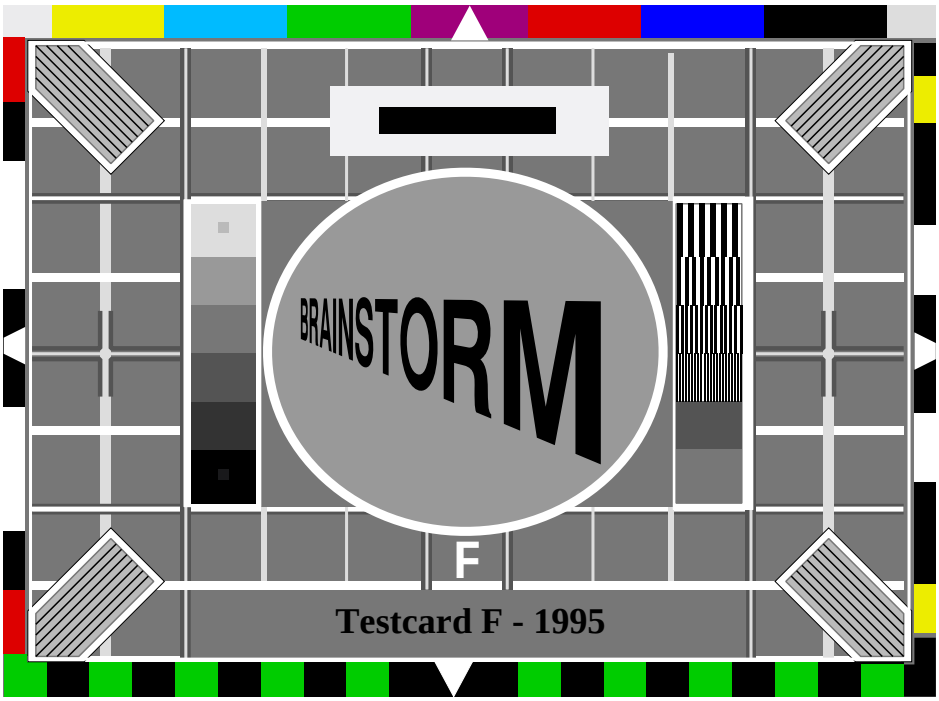
FOUND IT!

Place "CALL DAT%" on it's own line. Put a "STOP" command before it. When the program stops, enter "CALL DAT%", then enter "PAGE=DAT%+LEX%". Type "LIST".
Yippee! That had me scratching my head for, oh, 2 minutes tops... :-)

[legality of hacking a commercial application]

Well, I don't know about that guys. I got Imagery from an AW coverdisc!

But I agree, it isn't nice to hack into commercial applications, though those that are written in BASIC are fun to play with (X-Stitch demo, !Vision...). It also allows you to remove bloody stupid persistent 'register me' windows from programs like the one in ToolKit (AW June 1995), an app that - well, let's say you can get better for free. And that 'register' window gives a little util all the appeal of ZAnsi. Yes, you guessed it. It's still on the AW disc. I'm writing my own! :-)



DOORBELL?

"Doorbell! Doorbell!", announces the computer in a voice that sounds like Darth Vader crossed with a mid-Atlantic telephone operator.

Using a simple circuit, a bi-directional parallel port, a little BASIC proggy and Superior Speech, the computer will announce:

"DoorBell"
"Intruder, front door"
"Intruder, back door"
"Intruder, living room"
"Intruder, garage"
"Intruder, kitchen"

And there are two data lines left unused. I am simply hooking a switch to various 'make' contacts/relays.

I could hook up a smoke detector. I could start getting complex with logic gates to provide more inputs (6 bits and two control bits would give 6<<4 or 24 inputs). I could get fancy with I^2C and have up to 128 inputs?

This is just to whet your appetite, actual construction and implementation is left to you.

BRAINSTORM?

The idea of this article is to give some ideas for you if you are bored or have a case of *programmer's block*. :-)

Try these...

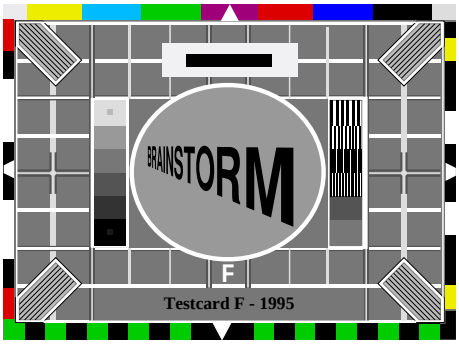
DOOR

A cryptic heading? Not really. I'm on about BBS doors. Most SysOp's out there are looking for a good door to make their BBS better than the rest. The de-facto for Acorn doors is the ArcBBS protocol. It may not be the best, but it is the most widely supported. ArcBBS and its userbase supports it (obviously), ArmBBS supports it (and it did so before supporting it's own door format). Forget about RiscBBS. Serious RiscBBS people have been shown the joys of ArmBBS. Archiboard, well... ArcBBS doors should work - but don't seem to. You'll need to get a copy of Richard Murray's DoorDocs file to see what supports what. The latest version is V5.

Here are some ideas:

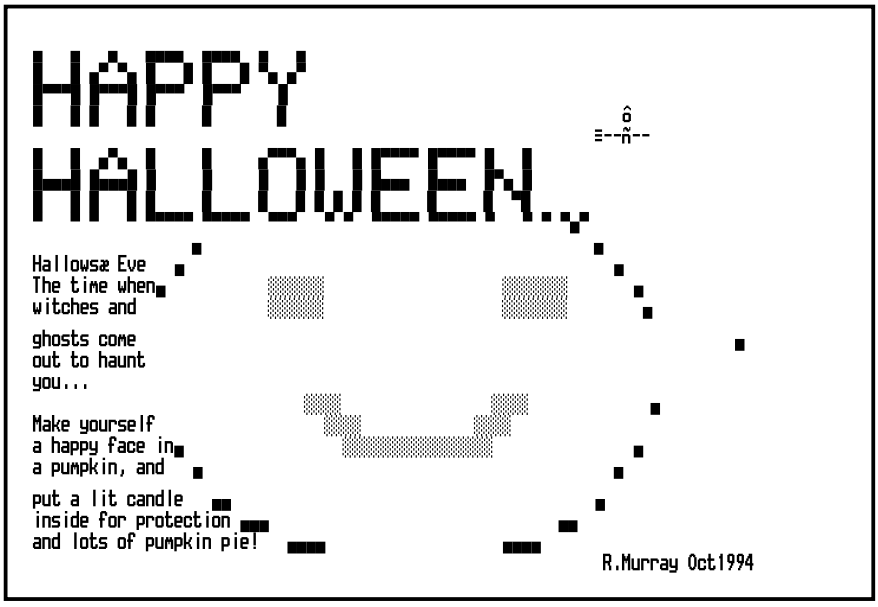
National Lottery predictor.
Download Queue Manager.
Graphical User Interface for BBS.
User-User chat system.
Games.
System Management.

Some doors already exist (VirtualCafe, UserEditor etc). Some don't. :-)



SPRITE-ASCII/ANSI

This, you take in a sprite. Resize it keeping the same aspect ratio (if ANSI) to fit, then start scanning the pixels and output an ANSI/ASCII representative...



I hope you will be able to come up with some plans and ideas for this page. Anything (within reason, of course) is acceptable.

Well, time to dig up the old excuse of "Sorry guv'nor. This is the first edition and I've had no feedback, so no more ideas for now".

Never mind. You can always use this page to get your fire going next winter.

This page - quality fire starter! :-)

Reader's Letters

Send your letters to:
"Richard Murray" at 2:254/86.1 (FidoNet) or "rmurray@digibank.demon.co.uk".

When will this thing be released? Will it be free?

Amy Hopkins; 2:254/86.86

Good news!
It will be released this week (15th May to 21st May). The hardcopies will cost you, the data copies won't.

That's it for the letters. So I thought I'd say a little about how this issue was created...

That is me! With Bert (a budgie) on my shoulder.

I am entirely wholesomely responsible for this issue. That's gotta change. :-)

Okay. The first thing I did was to sift through my 1Gb harddisc and countless floppies, looking for things that might just be interesting. I hope I found something to interest you.

The next step was to hook up my Canon A2Hi video camera to the digitiser, and grab a few prerecorded images. Above, that is Polly Page (Lisa Geoghan) from The Bill. I just grabbed it live from Meridian TV...

Finally, after editing and tweaking, I sat down with Ovation and began to compile the magazine, starting at the beginning, ending in a few pages. Not the way Acorn User would do it, but I'm not them. :-)

All in all it is a simple thing to pull off DTP. What is hard is getting it interesting, and making it look good. No doubt many of you think this little production isn't too wonderful. Well... Lend a hand!

For budding DTPers, here is a rundown of what I used:

Software:
- !Paint
- !Imagery
- !Ovation
- !Fonts, supplied with RO3/Ovation or PD
- !Draw & !Drw3Dptch
- My archives

Hardware:
- Acorn A5000 with 4Mb RAM
- AKF12 monitor (!)
- 1Gb HD (smaller can be used. Hehehe!)
- Betamax VCR
- Canon A2Hi vcamera
- Vision Digitiser
- Various lenses/filters

And finally...
- Everything you see in the magazine (like 2 phones)...

Oh, yeah... Mustn't forget... This was completed (start to finish) in about 12 hours. :-)

Feetch, Feetch!

HELP WANTED
ENQUIRE WITHIN

This magazine with put together by Richard Murray to help start the ball rolling. Acorn User is not likely to delve into overly techie stuff, and it isn't commercially viable to make a brand new magazine. After promising Acorn User and Acorn Computing will both be run in parallel (after one got taken over by the publishers of the other), guess what - they are now one magazine. We have seen the demise of Risc Developments too. Times are hard, Acorn appears to be wandering away from computers. What is there to satisfy your average Joe Techie?

This? Well, it's up to you now. I've put in my part. I don't want to contribute more than 30% to the next issue (not including compilation/editing). What could gave you a better feeling than to share your immense knowledge with others. Bring an insight into the ways things tick, kick your Acorn in the balls yet again... Has anybody ever found a dead BBC micro with a fault that wasn't traceable to the video ULA? Such problems are hard to find. The BBC micro was a sturdy old machine, and in the same spirit, your Acorn RiscOS lump o' sil'con can take a bashing. How far are YOU prepared to go? Tell us what, where, when, how and who is invited. Do you hate PCs? Say so. As long as it's not legally suspect, I'll put it in...

I would appreciate ready-made Ovation pages, but life is rarely so nice. I can accept:

Acorn TextFile, DOS TextFile, Simple RTF, 1stWord+ files, AcornDTP files, Impression (directories), Style (files), Ovation files, TechWriter files, WordPerfect 5.1, WordPerfect 6, XYWrite, DrawFiles and Wordz.

For graphics, I'd appreciate 8bit Acorn spritefiles, but the following can also be accepted:

RiscOS 16/24/32bit sprites, ProArt/ProArt compressed, Clear, TIFF, IFF ILBM, .PIC, ColoRIX 8bpp, GIF up to 8bpp, MacPict2 8/24/32bpp, PCX, BMP, IMG, Unix RLE, TGA, MTV, QRT, JPEG/JFIF, PBM and Imagery.

I, the editor, reserve the right to amend anything sent.

Upload the stuff for me on Arcade, DigiBank or ArcTic, or FREQ it to me on ArcTic...
Hardcopy people, give me the files on a disc, or a manuscript copy.

Everything sent will be looked at, and replied.

SUBMISSION FORM:

Please cut out and send...

Name: _____
Age : ___ years
Fido: __:__/__.__
Article:_____
Detail:

{1b}[p;i;z;z;a

Advertisment feature...

ANSI codes:

Note, this isn't always STRICTLY ANSI. The 'ANSI' terminal appears at the moment to be a mishmash of ANSI, VT100 and bolt-on ideas. These commands should be tested on at least two comms programs before you implement them.

The sequence **ESC** doesn't mean 'E S C', it means **ESC**ape, character 27, &1B.
So '**ESC**[11A' is '<&1B> [1 1 A'.

Keystrokes:

Pressing a particular key in a comms program will send a code, here are the special codes:

Left cursor key	**ESC**[D
Right cursor key	**ESC**[C
Up cursor key	**ESC**[A
Down cursor key	**ESC**[B
Home key	**ESC**[H
End key	**ESC**[K
Ctrl + Home key	**ESC**[L
Ctrl + Page Up key	**ESC**[M
Function key 1	**ESC**OP
Function key 2	**ESC**OQ
Function key 3	**ESC**Ow
Function key 4	**ESC**Ox

* - Not all terminals send function key codes. Maybe this is VT100?

ASCII codes that affect the terminal:

&7	^G	Beep
&8	^H	Destructive backspace
&9	^I	Tab
&A	^J	Line feed, scroll if necessary
&C	^L	Clear screen, home cursor
&D	^M	Return, cursor to column 1.

It should be noted that to 'continue on the next line properly', you must send a CHR$13 and a CHR$10 (return cursor, move down a line).

Scroll up

ESCD

The screen is scrolled up one line. The bottom line is filled with spaces coloured according to the current attributes.
Note, there is no left bracket '[' after the **ESC**.

Scroll down

ESCM

The screen is scrolled down one line. The top line is filled with spaces coloured according to the current attributes.
Note, there is no left bracket '[' after the **ESC**.

Reset terminal

ESCc

This command resets the terminal to its defaults. Useful after a door or ANSI animation (both known for playing with the terminal).

This is a brief example
of the DoorDocs file.
Download it from a BBS near you!