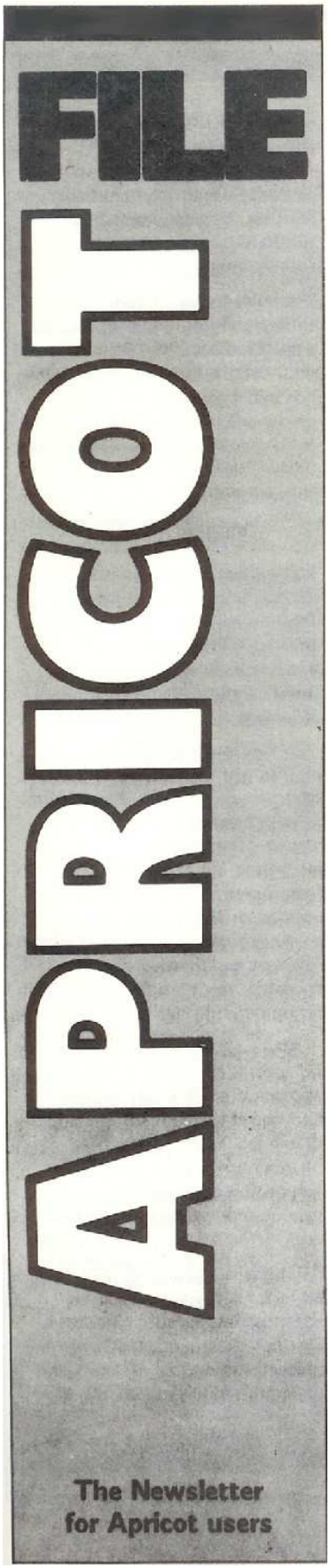
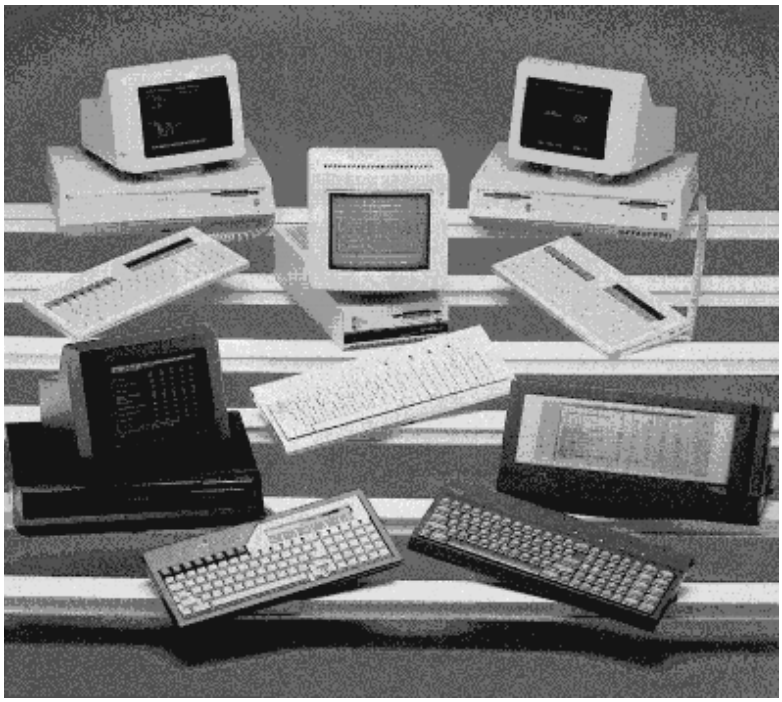


Apricot File 2_10



Scans courtesy of www.actapricot.org

Copyrighted content republished with thanks and permission from:

> **Dennis Jarrett of TP Group**
> **Stuart Bunt**
> **Eddie Bromhead**
> **Mark Whitehorn**

Many additional thanks to Eddie Bromhead for scanning all of Release 3, 4 and extra inserts.

Release 2.10

FILE APRICOT

The Newsletter for Apricot users

July 1987

Apricot loses out in Wordplex chase

£14 million bid beaten

Apricot has lost out in its £14.1 million bid for the loss-making office automation group Wordplex Information Systems.

The proposal was mooted just as Apricot reported a return to profitability for the year to 31 March 1987 (just as we forecast). Profits were £4.04 million on sales of £71.16 million, compared with losses of over £15 million on £90.6 million sales the year before.

Wordplex is in dire straits because the whole office automation market has become increasingly integrated with the corporate computing business; Wordplex failed to move with the times and continued its traditional concentration on single-function word processing systems. As a result, last year the company lost over £2 million on a turnover just below £47 million.

Apricot offered 13 shares for every 10 Wordplex, with 130p cash as an alternative; the idea was to finance this through a new share issue to existing shareholders. The Wordplex board initially rejected Apricot's offer in favour of a refinancing package organised by management services concern Octagon Industries, which would produce £3.6 million in cash through a rights issue of its own and a new share placement plus a £4 million bank loan.

That just failed to win the support of shareholders, so the Wordplex board ended up reluctantly recommending the Apricot offer "in the absence of anything better".

Chase Securities, which holds a fifth of Wordplex, was reportedly in favour of the Apricot proposition; but in the event Apricot had received less than 13 per cent acceptances when its offer closed. The key factor was the last-minute arrival of an alternative bid from a somewhat unlikely source, the Norwegian minicomputer group Norsk Data, which was prepared to pay 155p per Wordplex share against Apricot's 130p.

In the event Apricot had already decided to withdraw if it could not declare its offer unconditional — "you have to avoid letting your heart rule your head" commented Roger Foster. The market appeared to agree; Apricot shares immediately jumped 5p to 111p on the news, Wordplex held steady at 155p.

Apricot apparently wanted the whole caboodle; Roger Foster said the company believed that Wordplex's maintenance operation, sales force and customer base would complement Apricot's. And indeed it probably would, particularly in terms of offering desktop publishing to Wordplex's eminently upgradable WP users.

On the other hand, Wordplex could have been a major millstone around Apricot's financial neck — it has debts of £12 million, and any buyer would have to spend some £5 million in write-offs and restructuring.

Norsk Data has that kind of beef, though its UK presence has hitherto been muted. The acquisition doubles the Scandanavian group's revenues and staff in Britain. ◆

In this issue

The Workshop: Batch File Recipes 2
Notebook: Caveat Emptor 5
The Workshop: Safer Spreadsheets 6
Queries from Readers: The Q&A Clinic 8
Doodles by David: Tips and Tricks 11
Artificial Intelligence: Reflections on AI languages 12
DEALS FOR READERS Update 14
The Complete Apricot Escape Codes — part two 16
Readers' Writes: letters from subscribers 20
The Workshop: The Apricot 'Control Device' 22
Free Small Ads 27

Batch File Recipes:

Change without pain, Format without fuss

By David Lewis

As a follow-on to the recent *Cookbook* series all about batch files, we're inviting readers to send in their favourites. For starters, here are two from David Lewis that illustrate the kind of thing we're looking for — handy tools for dealing with day-to-day requirements. You might not need the specific applications they cover, but there should be enough ideas and techniques in the recipes for you to recycle.

If you have a hard disk, you'll know the difficulties of directory-hopping.

You're in subdirectory \CLIENTS\SMITH; you'd like to move into subdirectory \ACCOUNTS\1987 and run SuperCalc. Then you'd like to go into yet another subdirectory called \TAX and edit a file called TAXMAN.LET using SuperWriter; after that you want to get to the root directory and run EDLIN to edit a batch file called LOOKFOR.BAT.

This is what you would have to do:

```
CD \ACCOUNTS\1987
SC
CD \TAX
SW TAXMAN.LET
CD \
EDLIN LOOKFOR.BAT
```

That's a total of 66 characters, including six Returns and four backslashes. If you're a touch-typist like me, you'll probably miss the backslash key at least once and instead type a '1', or a tab, or the HELP key.

What's more, you can't type the third line until you've exited from SuperCalc, nor the fifth line until you've come out of SuperWriter.

CHANGE.BAT avoids all those backslashes (and the Returns) and lets you accomplish the entire exercise in a single command line which is almost like a directly-entered batch file. For all the operations in the example you would type this:

```
CHANGE ACCOUNTS 1987 RUN SC CHANGE TAX RUN SW
TAXMAN.LET CHANGE RUN EDLIN LOOKFOR.BAT
```

The newsletter's column-width is too narrow to show it properly, but the only Return is at the end of the input line — you type 85 characters, but no backslashes. And execution is automatic as soon as it's entered.

CHANGE.BAT

Line numbers are included for reference — don't enter them when keying in this batch file.

```
1    ECHO OFF
2    REM CHANGE.BAT
3    IF NOT %1#==# GOTO ROOTDIR
4    ECHO Directory not specified
5    GOTO END
6    :ROOTDIR
7    CD \
8    IF %1==CHANGE SHIFT
9    IF %1==change SHIFT
10   :TEST
11   IF %1#==# GOTO END
12   IF %1==ROOT SHIFT
13   IF %1==root SHIFT
14   IF %1==RUN GOTO RUNTEST
15   IF %1==run GOTO RUNTEST
16   IF %1==CHANGE GOTO ROOTDIR
17   IF %1==change GOTO ROOTDIR
18   CD %1
19   SHIFT
20   GOTO TEST
21   :RUNTEST
22   IF %2#==# GOTO NOPROG
23   IF %3#==# GOTO DOIT
24   IF %3==RUN GOTO DOIT
25   IF %3==run GOTO DOIT
26   IF %3==CHANGE GOTO DOIT
27   IF %3==change GOTO DOIT
28   GOTO PARAMET
29   :DOIT
30   %2
31   SHIFT
32   SHIFT
33   GOTO TEST
34   :PARAMET
35   %2 %3
36   SHIFT
37   SHIFT
38   SHIFT
39   GOTO TEST
40   :NOPROG
41   ECHO Program not specified
42   :END
```

Syntax for CHANGE.BAT

The CHANGE.BAT command line consists of 'phrases'. Phrases, and the words comprising them, are separated by spaces.

There are two kinds of phrases. Anything beginning with CHANGE makes a particular subdirectory the current one; a phrase beginning with RUN invokes a program or MS-DOS command.

The command line *must* begin with a CHANGE phrase. Such a phrase may take the following forms:

CHANGE

... (with no parameters) for the root directory; this cannot be the sole phrase in a command line (you can't enter just CHANGE on its own).

CHANGE ROOT

... for the root directory.

```
CHANGE directory
```

... the directory named must be a subdirectory of the root directory.

```
CHANGE directory1 directory2 etc
```

... the first directory named must be a subdirectory of the root directory, the second must be a subdirectory of *directory1*, and so on.

A RUN phrase should be followed by a command with or without a single argument.

So in the example given, we see the following phrases:

CHANGE ACCOUNTS 1987	CHANGEs to the 1987 subdirectory of the ACCOUNTS directory, which itself is a subdirectory of the root
RUN SC	loads and runs SuperCalc
CHANGE TAX	switches to the TAX subdirectory of the root
RUN SW TAXMAN.LET	loads and runs SuperWriter with the file TAXMAN.LET as an argument (so that document will be loaded automatically)
CHANGE	changes back to the root directory
RUN EDLIN LOOKFOR.BAT	loads and runs EDLIN with the file LOOKFOR.BAT.

Building the batch file

CHANGE.BAT contains examples of advanced MS-DOS batch file programming — in particular the commands IF, GOTO and SHIFT. And before going any further, I should explain about 'arguments' (or parameters) and the way they are referred to in batch files.

In the example given, CHANGE and each of the words following it is an 'argument' of that command line. These 'arguments' (preceded by the percentage symbol) are numbered from %0 to %9: it is only by virtue of the SHIFT command that arguments in excess of ten can be given effect. (As you'll have guessed, CHANGE itself is treated as the argument %0.)

All this means is that if in a batch file you refer in a particular line to %1, say, the text of the argument in question is substituted when that line is executed.

In the example, these are the arguments and their numbers initially assigned to the command line:

```
%0      CHANGE
%1      ACCOUNTS
%2      1987
%3      RUN
%4      SC
%5      CHANGE
%6      TAX
%7      RUN
%8      SW
%9      TAXMAN.LET
```

For the time being the last four words of the command — CHANGE, RUN, EDLIN and LOOKFOR.BAT are ignored.

For the purposes of discussion, each line of CHANGE.BAT has been numbered (obviously you shouldn't type in these line numbers when you come to try entering the thing). Line 3 of the batch file ascertains whether or not any arguments at all have been assigned.

Note the odd syntax, with the pair of equals signs and the hash mark. (In principle any character could be used instead of '#' but it's better practice to use a non-alphanumeric character.) If you simply entered CHANGE on its own, parameter %1 would be a null string: %1# would therefore equal '#' and program control would pass to lines 4, 5 and 42. In our example, though, argument %1 is ACCOUNTS; then %1# does *not* equal '#' (since 'ACCOUNTS#' is not the same as '#') and control passes to lines 6 onwards.

Incidentally, if line 3 omitted the two hash symbols like this ...

```
IF NOT %1== GOTO ROOTDIR
```

... the batch file would be aborted with a syntax error message. That's because the right side of the equation (and the left too, if no arguments have been entered) would be a null string.

The words ROOTDIR, TEST, RUNTEST, DOIT, PARAMET, NOPROG and END are all labels. In the lines which they label they are preceded by a colon, but they are used just as they are following the GOTO command in the lines passing control. Labels have no function other than as the destination for a GOTO command, and are not otherwise executed.

SHIFTing

The SHIFT command simply strips off the argument hitherto known as %0, renumbers %1 as %0, and renumbers any subsequent arguments accordingly. Using our example, the effect of the first SHIFT executed would be to renumber the arguments as follows:

```
%0      ACCOUNTS
%1      1987
%2      RUN
%3      SC
%4      CHANGE
%5      TAX
%6      RUN
%7      SW
%8      TAXMAN.LET
%9      CHANGE
```

Note that the last percent-sign parameter has now picked up the first of the 'spare' words in the

original command string, leaving three currently unassigned. They'll be added by successive SHIFTs, and the effect of the last SHIFT executed before the last phrase would be thus:

```
%0      CHANGE
%1      RUN
%2      EDLIN
%3      LOOKFOR.BAT
```

There is a twofold purpose to this use of SHIFT. First, it means a command line with more than ten arguments (including the command itself) can be fully executed.

The second purpose is to enable each phrase to be identified and dealt with in a consistent manner and one at a time. It is easy to program for a phrase whose first word can be identified as %1, whose second is %2 and whose third is %3: but it's jolly hard to do so if the first word of the phrase could be any identifier from %0 to %9.

Most lines in a batch file starting with IF will have the form ...

```
IF a==b GOTO LABEL
```

But the command following the condition need not be a GOTO. For instance, in lines 8 and 9 I have used the form ...

```
IF a==b SHIFT
```

(Note that there is no 'THEN' command; this is implied immediately following the equation.)

Wrap-up

To give some more detail about the batch file itself, lines 8 and 9 make sure that any CHANGE found in the middle of the command line is stripped off. Line 11 terminates the batch file when all arguments have been executed (so that there is no %1 left). Lines 12 and 13 strip off the argument ROOT when its job has been done. Line 18 is the business line which actually changes the directory; and it is followed by line 19 which immediately brings into play the next directory, or CHANGE or RUN.

Lines 22 to 27, after the label RUNTEST, ascertain whether or not there is an argument or parameter to be executed after the command to be run, or indeed whether there is in fact any command following RUN. Line 30 runs the command without any parameter, and lines 31 and 32 effect two subsequent shifts — the first to strip off the RUN and the second to strip off the command name.

Line 35 runs the command (%2) with its parameter (%3); and this time there are three SHIFTs (lines 36 to 38) since the parameter must also be stripped off.

NEWDISK.BAT

Here's another demo of some advanced MS-DOS commands, a batch file that is designed to make it a little easier to format floppy disks.

NEWDISK.BAT

Line numbers are included for reference — don't enter them when keying in this batch file.

```
1    ECHO OFF
2    REM NEWDISK.BAT
3    IF #==%1# GOTO ERROR
4    IF #==%2# GOTO ERROR
5    ECHO F > FLOPPY.DOC
6    IF %1==D GOTO DOUBLE
7    IF %1==d GOTO DOUBLE
8    IF %1==S GOTO SINGLE
9    IF %1==s GOTO SINGLE
10   :ERROR
11   REM ECHO in next line to be followed by
     <TAB> <SPACE> <CR>
12   ECHO
13   ECHO  Syntax (S=single-sided, D=double-sided
           disk):
14   ECHO  NEWDISK S VOLUMENAME or
15   ECHO  NEWDISK D VOLUMENAME SYSTEM (to add
           system tracks)
16   GOTO END
17   :DOUBLE
18   IF #==%3# GOTO DBLDATA
19   IF %3==SYSTEM GOTO DBLSYS
20   IF %3==system GOTO DBLSYS
21   :DBLDATA
22   FORMAT B:%2 /M1 < FLOPPY.DOC
23   GOTO VOLUME
24   :SINGLE
25   IF #==%3# GOTO SNGDATA
26   IF %3==SYSTEM GOTO SNGSYS
27   IF %3==system GOTO SNGSYS
28   :SNGDATA
29   FORMAT B:%2 /M0 < FLOPPY.DOC
30   GOTO VOLUME
31   :DBLSYS
32   FORMAT B:%2 /M1 /SA: < FLOPPY.DOC
33   GOTO VOLUME
34   :SNGSYS
35   FORMAT B:%2 /M0 /SA: < FLOPPY.DOC
36   :VOLUME
37   VOL B:
38   :END
```

The MS-DOS command FORMAT can be complicated to use. For example, to reformat a double-sided disk in drive B with System tracks and the label 'MAINSYSTEM', you would need to enter ...

```
FORMAT B: MAINSYSTEM /M1 /SA:
```

Using NEWDISK.BAT you would enter ...

```
NEWDISK D MAINSYSTEM SYSTEM
```

The first word of the command line is always 'NEWDISK', of course. The second (and compulsory) word is 's' or 'S' (to format a single-sided disk) or 'd' or 'D' (for double-sided). The third parameter — also compulsory — is the label-name you want to give to the newly formatted disk; it should not exceed 11 characters, and must contain no spaces.

The fourth (optional) word is 'SYSTEM' or 'system', used if you want to add System tracks.

NEWDISK assumes the disk to be formatted will be in drive B; MS-DOS of course should be in the

current directory, or the current path, of the hard or floppy disk in drive A.

Many of the usages in this batch file will be comprehensible from what I have said in introducing CHANGE.BAT, but there are some new ideas too.

Line 5 is interesting. Normally the command ECHO followed by some text will echo those characters on to the screen. But this output can be redirected: for example, ECHO HELLO >PRN will print the word 'HELLO' to an on-line printer.

Line 5 has the effect of creating a file named FLOPPY.DOC which contains the single letter 'F'. Why? Because the FORMAT command pauses at one stage to invite you to press 'F' if it's a floppy disk you want formatted. It *will* be a floppy that you want to format, but NEWDISK.BAT saves you having to press that 'F'.

You will notice that lines 22, 29, 32 and 35 all take redirected input to the FORMAT command from FLOPPY.DOC. The effect is that FORMAT.EXE, instead of stopping to invite you to press 'F', will take the character 'F' from FLOPPY.DOC and continue accordingly. (Note that the redirection arrow is pointing *left* in lines 22, 29, 32 and 35 to signify redirected *input*, as opposed to *right* for redirected *output* in line 5.

What about lines 11 and 12? The purpose is simply to enable a blank line to be echoed to the screen. When transcribing line 12, you should type 'ECHO', then the tab key, then the space bar, and then the Return key. (Simply typing 'ECHO' and Return will result in an unwanted and untidy line telling you that ECHO is off.) ◆

Contribute!

Why not share your own useful utilities? Let us have your own home-brewed batch files, and we'll write them up for this *Recipe* series. Just drop us a line at APRICOT FILE, PO Box 509, London N1 1YL.

Notebook

Caveat Emptor

Here's a salutary tale for anyone contemplating comms. APRICOT FILE subscriber **Norman Mash** wanted to buy a modem for his Apricot. Not knowing anything about such matters he contacted the **Compulink** user group, who suggested the **Miracle Technology WS4000** modem on which Compulink has a discount offer. Mr Mash looked at the bumf and decided this was the modem for him.

So in December 1986 he bought and received his WS4000. All well and good — until Mr Mash discovered his modem didn't do the things he wanted it to.

Being a new user, not unreasonably Mr Mash would have liked everything to run smoothly. It didn't. Mr Mash was using the modem with the One-to-One e-mail service, the ASYNC software from Apricot and TRANSEND from Messiter Software. The problems included these:

- The modem didn't recognise split baud rates while in Terminal mode. At these speeds the Hayes ATD command did not appear on the screen.
- The modem didn't accept Prestel graphics at 1200/1200 baud with either of the obvious parameter combinations (8 data bits, 1 stop bit, no parity; or 7 bits, 1 stop bit, even parity).
- The programmable internal memory store was lost whenever the modem was disconnected (and there's no battery back-up).
- The redial command didn't work.
- The modem didn't seem to operate at 75 baud at all.
- When dialing a remote service, the delay between the 'Connected' message and actual on-screen connection was ridiculously long.

There were many other faults, problems and inadequacies; and the manual was incomplete. Still, Miracle Technology has a customer Help line. So Mr Mash and various people from his office called it ... and called it, and called it. There was never any reply.

Mr Mash got fed up and decided to return the WS4000. But Miracle's documentation said he couldn't return the modem without ringing first for authorisation; and when he called again, he still got no reply.

Mr Mash sent Miracle Technology a letter. He sent copies to us, to Compulink and to the Trades Description authorities. It is not a short letter, and Mr Mash did not mince words: "It is not that [the Help line] is engaged and your staff are busy, it is just that there is *no* reply. Either the staff have become fed up with having to answer the many queries on your defective products or you do not employ enough staff to deal with the many problems". He threatened legal action under the Trades Descriptions Act and on the basis of customer rights (he couldn't return the product because he couldn't phone up).

Eventually Miracle got in touch with Mr Mash and explained that what he really needed was a more expensive WS3000 modem. Mr Mash was persuaded to part with several hundred pounds. He now has his 'professional' modem, but finds there are problems with the phone lines and switchboard in his office. His modem lies in its expensive box and waits for the men from Telecom; meanwhile he can't really check that all the problems have been solved.

Some people have all the luck. ◆

Safer Spreadsheets

How to avoid problems

By Simon Potter

■■■ Identification ■■■

- **Identify the spreadsheet.** In cell A1 you should always have a line of text stating as clearly as possible what the model actually is.
- **Identify the responsible parties.** All spreadsheets should carry a text note saying who actually produced it; and there might also be a note of who's amended it subsequently.
- **Date the spreadsheet.** That means including the original date of creation, the dates for any amendment, and probably the date of the last recalculation (the TODAY function in SuperCalc and most other decent spreadsheets allows you to do this automatically, but don't use it for the date of last amendment — many users habitually recalculate every time they load a worksheet, whether or not they're going to alter it). It makes sense to combine dates and names in an 'amendment record' somewhere in a spare corner of the worksheet.
- **Identify the spreadsheet's users.** If the spreadsheet is passed on to someone else for consolidation, indicate the destination and the date.

■■■ Explanation ■■■

- **Annotate the spreadsheet.** Sensible use of titles and headings will help you and anyone else understand exactly what's going on, but unless you have absolutely no room left it also makes sense to include notes about what a particular formula means and how it is assembled — spreadsheet formulas can look pretty cryptic to the uninitiated, and while it may have been clear to you at the time you might well have forgotten precisely how you'd decided the spreadsheet should arrive at that particular figure. Comments like this can often be 'hidden' or placed well out of the way so that they won't be printed when the figures are produced in hardcopy form.
- **Standardise on spreadsheet layout.** For instance, you might always reserve the top 20 rows for identification and amendment records; column A, set to a decent width, could always contain explanatory narrative for the formulas in particular cells.

■■■ Management ■■■

- **Plan ahead.** As with many desktop computer applications, there's a tendency to jump straight in — switch on the computer, load the spreadsheet package, start entering data, add the formulas and headings later. Immediacy is one advantage of microcomputers over larger machines; and that also applies to spreadsheeting by comparison with 'true' financial modelling. But it still makes sense to think ahead, to map out at least the row and column headings and the calculations before you start entering data. At the very least, this will save you having to insert or move too much information later — those changes can really mess up formula calculations.
- **Match existing practice as closely as possible.** Provided the existing forms and report layouts are adequate, your computerised version should follow them as closely as the spreadsheet will allow. This will help with data entry, and the printed end-results will be more immediately comprehensible to the recipients. At the very least you should utilise the same headings (expanded if necessary) and the same general order in which information is presented.
- **Audit any critical spreadsheets.** Check the data sources, the formulas and the results. At the very least, this means adequate (more is better than less) explanation and comment for all cells containing any formula the function and operation of which isn't immediately obvious. Any key spreadsheet should be accompanied by an annotated printout of cell contents — with 1-2-3's /Print Options Other Cell-formulas command, for example, or SuperCalc's /Output Contents option.

■■■ Spreadsheeting ■■■

- **Don't SUM from first to last.** If you want to add up the entries in cells D6 to D9, the obvious method is SUM(D6:D9). But subsequently adding a row at D6 means the SUM won't now be accurate — the formula will have changed to SUM(D7:D10) after the /Insert Row command. The solution is to include something like a repeating rule above and below the values to be added:

```
     |   D    |
4        SALES
5    ---------
6     12663.55
7      1039.60
8     37173.11
9       450.00
10   ---------
```

Now you can have D11 contain the formula SUM(D5:D10), and the result will still be an accurate adding-up of the values — text like repeating dashes isn't counted in the SUM. But you'll be able to add subsequent rows anywhere within the rules, and the SUM will accommodate the new entries.

This also applies to SUMming rows. Put a delimiter mark of some kind in the cells on either side of the values, and have your SUM adding up everything inside the marked area — including the marker text. Then you can safely add extra columns within the SUM range.

The same procedure can be used for several other functions, including COUNT and AVERAGE.

- **Write the formulas before entering data.** That way you can test the worksheet with dummy data to check formulas before you commit yourself to entering any values. To pick a simple example, you can test a row or column SUM by entering '1' in each cell; if you have ten cells, obviously the SUM should produce '10'.

- **Use column widths to verify results.** In some cases, you can set a column width to provide an instant check. For example, if the largest number you should get from a calculation is 999.99, set that column's width to '7' and make sure it's formatted for no more than two decimal places. With most spreadsheet packages, any number larger than 999.99 will appear as a row of asterisks or less-than signs in that cell when you recalculate. You might have to do some dummy runs to find out the maximum column width for a particular number with decimal places, percentage signs, commas, currency symbols and the like.

- **Cross-check formulas.** As a general rule it makes sense to provide cross-foot totalling whenever you can. It may be the case that you SUM a series of rows and then calculate a Grand Total by adding up the SUMs in that column. But if you also SUM each of the data columns, you can cross-check by adding up the row containing the crossfoot totals.

- **Use manual recalculation.** It's much quicker than having the spreadsheet recalculate automatically after each amendment. But include a note at the top of the spreadsheet warning other users that manual recalculation is required when any alteration is done; or reset the spreadsheet to automatic recalculation before you save it and distribute it.

- **Use names for key data and formulas.** Most up-to-date spreadsheets allow this; it reduces the risk of your inadvertently referencing the wrong cell. And get into the habit of specifying 'absolute' cell references for key data; G11, for instance, means you won't get any unwanted 'adjustment' of that cell when you use /Copy or /Replicate.

■■■ Last words ■■■

- **Buy more memory.** If you're keeping your spreadsheets short and cryptic to fit them into 256KB, consider shelling out on an upgrade. You'll be able to run larger spreadsheets and you'll have more space for explanatory text.

- **Buy an updated version of the spreadsheet package.** If your spreadsheet software was a free gift from a kind friend in 1983, consider an upgrade. You'll get more functions, a better manual, and — most important — more ways to validate your work. ◆

Contributors in this issue

Dennis Jarrett is Editor of the newsletter. He had edited several magazines and written half a dozen books about computing.

Garrett Keogh is the staff writer for TP Group; since coming to TP he has acquired more knowledge about desktop micros and envelope-stuffing than he ever thought possible — at the grand old age of nineteen he is adding to his computer education through a long process of trial and error(s).

David Langford is a full-time author and computer addict, whose latest book *EARTHDOOM!* — a spoof of disaster novels, written with John Grant and scorned by Kingsley Amis — is "by no means" about Apricot BIOS variants. (The inclusion of a permanently drunk character called Apricot should not be misinterpreted.) He is inordinately proud of being a guest of honour at the 45th World SF Convention at Brighton in August 1987).

David Lewis is a solicitor for a London borough that happens to be a keen user of new technology; he started computing himself six years ago with a ZX-81, but says his *annus mirabilis* was 1986 "when I bought an Apricot Xi and a Miracle Technology WS4000 modem — the Apricot took a fair slice of my savings, and the modem has taken care of my salary ever since". His work has made him "fairly conversant" with the Data Protection Act. Interests apart from law and computers include astronomy, chess, languages and international politics.

Simon Potter is a business systems consultant with a speciality in word processors, file managers and spreadsheets. He says he'd like to like computers but can't find one that likes him enough to warrant that.

David St. John-Wallis is a sort of psychologist who runs an export marketing consultancy and writes *enormously* thick marketing books about the developing world. He's a chastened man since he first leapt to journalistic prominence in these pages; vituperation, vilification, abuse, rejection and guffaws have nearly broken his spirit. So he now runs a sort of computer software company which doesn't do very well. He says that all contributions will be gracefully acknowledged.

Mark Whitehorn started life as a Cytogenicist but the Siren Song of the computer was too strong and he is now working as a programmer at a Scottish university, particularly on the application of computers (especially Apricots) to education. He says his hobbies include penguins, building 1960s racing cars on the cheap, and indoor underwater powerboat-snooker (of which only the last is apparently untrue).

Nick Wilson wrote his first article when he was 13 and first appeared in print when he was 17. He helped write the first Apricot BIOS, and as such now wears a convincing disguise. Since the Dark Days at Apricot, as he calls them, he has worked for a Swiss software company writing PC programs; he now writes the software that sits inside petrol pumps.

The Q&A Clinic:

Questions by subscribers, answers by APRICOT FILE

Between us, the staff of APRICOT FILE and our contributors offer an unusually wide range of expertise and experience. As many subscribers will know, we handle a lot of queries each month by mail or by phone; and every so often we run the Q&A page to present what we feel are the more interesting and/or useful of these. If you want to make use of the service, just drop us a line ...

Q. Memory missing

A while ago I purchased a 256KB memory expansion board for use in my 512KB Apricot Portable. It doesn't work; the switch-on memory indicator still shows 512K. The board works ok in a friend's 256KB Portable. Any idea why?

A. This could be down to a variety of causes. The obvious (but remote) possibility is a faulty connection to or in the expansion slot; can you try another board in there?

The more likely alternative is that the start address for the additional RAM is set to 256K, meaning that it's set up so that the Apricot detects it as starting at that address. This would account for the correct operation in a 256K machine; if there's already some memory starting at 256K (as would be the case in a 512K machine) the Apricot will simply ignore the duplication when you add the RAM board.

So DIP-switch settings are the most likely candidate, if the board has them — most add-on memory expansions do have to be configured via DIP switches to indicate the start and end of the expansion as 256, 384 or 512K (a few use short lengths of wire called 'straps' or 'jumper leads' to do the same thing). Set the switches to start the extra memory at 512KB instead of 256KB.

A few of the boards we've encountered weren't customised particularly well for the Apricot; their base addressing assumed that they would be sitting on top of 256KB of existing memory — no more and no less (this was the case, for instance, with the RAM boards we sold originally; our current DEALS FOR READERS products don't have this restriction). If this is the case here, there isn't much you can do about it except return the board to the supplier and ask for a refund or a replacement ...

Q. PRN in batch files

I have a 50-line batch file which sends a sorted directory listing to a disk file, sets up the printer by COPYing files containing hex codes to device PRN, copies the directory listing file to PRN, and then resets the printer by COPYing more hex-code files. It uses up to five parameters which determine the contents of the directory list (for instance, *.BAT) and the printer set-up codes.

There are usually no problems the first time this is run. But second and subsequent shots will bring up 'File Creation Error' on the DIR and inside the batch file, and sometimes we get an 'EXEC failure'. Each subsequent command that uses redirection then produce the 'File Creation Error' message.

The syntax of individual lines is perfect — they work ok when tested in smaller batch files — so the bug appears to have something to do with PRN not being recognised as a valid 'file'.

A. We tried to reproduce this with a simpler arrangement:

```
DIR|SORT > SORTED
ECHO [some ESC codes] > PRN
COPY SORTED > PRN
```

There was no problem with this, even when it was run several times. Maybe the additional complexity of the five parameters is getting MS-DOS into a tangle?

Or it could be that the long batch file ends up with too many files open — try FILES=50 in the CONFIG.SYS. That should get over the 'file creation error' problem ...

Q. SuperCalc roll-forward

I use a spreadsheet to monitor the performance of subcontractors. My problem revolves around the need to update the monthly figures. In brief, I have one column for value of work done up to the start of the current period (column A, say): a second column for work done in the current period (column B): and a third that sums these two to produce value of work to date (C). At the end of a monthly period, the value in column C should replace that in column A (because it represents the new total to date) with B and C being zeroed. How can I do this?

A. The answer is simply to Save the *values* from column C and not the formulae. For a start, set up a template worksheet called say MONTH.CAL with the calculation being done in column C; columns A and B would be empty, with A waiting for the totals to date and B ready for input of the current month's data.

Now here's the simplest and least elegant method. Load MONTH.CAL and use it to enter the current month in column B. Since there's nothing to add from

column A, the calculations in C should duplicate the entries in B. Save the file to disk as usual using a name like JUL87.CAL — then save it again with a new name (say JUL87V.CAL) but using the Values format. The syntax would be ...

```
/Save, JUL87V.CAL, Values, All
```

When you start a new month, Load JUL87V.CAL and Delete columns A and B. Or you can load Part of JUL87V.CAL, specifying column C as the target (range C1:C999 or whatever) and cell A1 as the destination. Either way, since the *values* and not the *formulae* in the original column C have been saved, you won't get any errors from this. You can then Load in MONTH.CAL over the top of what's on the screen, and the worksheet will use the new contents of column A — the data formerly in column C of JUL87V.CAL — in its recalculation.

Follow this procedure each succeeding month and you'll have a running total carried forward. Of course, the more elegant solution is to automate all this with an .XQT file.

Q. Automatic answers

The 'automatic answers' idea in Part One of the *Batch File Cookbook* (APRICOT FILE Release 2.6) didn't work on my Xi-HD with BIOS 3.1 — it returned a 'file not found' message. Why?

A. Well, the obvious possibility is that you haven't created the 'answer' file from which input is redirected. If that's too simple, perhaps you inadvertently saved it in another subdirectory, one that isn't the current directory and one that isn't referenced by a current PATH command.

Q. Oki printer drivers

I have an Oki 192 printer and I'd like to be able to use it to produce Greek characters. Do you know of an Apricot printer driver that will handle the Oki?

A. No, but if the Oki can be programmed to print graphics characters it shouldn't be too difficult.

You can probably **download** the required characters: check the manual for the relevant character codes, put the corresponding ESCape sequences into a text file, and send it to the printer with a command like ...

```
COPY NEWCHARS.TXT > PRN
```

Thereafter the printer should perform appropriately when it spots the Greek character codes arriving from the Apricot.

Or you should be able to **substitute** the Greek font characters *en route* to the printer by previously loading a 'printer intercept program' that does the conversion on the fly — Ansible's APP3.COM is recommended for this. Along with much else, it's on the ACOLYTE disk (available from DEALS FOR READERS as AS035 at £57.50).

Q. LQ1500 and PRINT

When I press PRINT on the Apricot, my Epson LQ1500 throws out the piece of A4 I have just loaded and then prints the screen. Can this be avoided?

A. Yes — though you'll have to patch the BIOS. This isn't as devilish as it sounds, and David Langford's *Disinformation Column* of APRICOT FILE Release 1.9 gives the blow-by-blow detail.

Briefly, you'll have to get at IO.SYS wherein the BIOS lives — and because that's a hidden system file, you'll first have to 'unhide' and 'un-system' it using something like the DEALS FOR READERS utilities ALTER or CHMOD. Then DEBUG IO.SYS, Search for the byte pair B0 0C, change the 0C to 00, and Write IO.SYS back to disk. Use ALTER or CHMOD to return it to hidden system status, reboot, and the formfeed should have disappeared next time you use the PRINT key.

To summarise, this is what you should get with BIOS 3.1 and using CHMOD (your input is shaded):

```
A>  CHMOD IO.SYS /SY
A>  DEBUG IO.SYS
    -  S 100 FFFF B0 0C
    XXXX:1B1D
    -  E 1B1D 0
    -  W
A>  CHMOD IO.SYS /HX
```

Q. Updating SuperWriter

I have version D1.09 of SuperWriter. Have there been updates to the program? If yes, where and how can I get the latest version? And we have some PC clones in the office; is SuperWriter available in IBM format? If yes, where and how can I get it?

A. SuperWriter version 1.03 is later than D1.09. The changes aren't huge, but may be worth having; the main improvements are a much wider range of configurable printers (via SW.DEF) and slightly better dictionary maintenance functions.

We don't think you can actually buy the update any more, but many Apricot dealers have it and will slip you an upgrade on production of your D1.09 master disk.

And as far as we know, SuperWriter isn't available in IBM format. But the Apricot version can be transferred on to an IBM disk and it will work under the Ansible APE Apricot emulator — see the back-page item in APRICOT FILE Release 2.9.

Q. Speeding SuperCalc

Is there any way of speeding up the recalculation in SuperCalc 3? Will FAST (or HARE, or QUIK) help?

A. The slowness of SuperCalc 3 is probably its biggest drawback. None of the speed-up utilities affect anything that will accelerate SuperCalc 3's recalculation speed, though there may be an *apparent* improvement (because the utilities improve the speed with which the display is rewritten). An 8087 maths co-processor chip won't benefit you, either. Martin Filmer's mods to the standard Apricot (see APRICOT FILE Release 2.8) *will* help, but at a hefty price.

The only palatable solutions are pragmatic — don't use automatic recalculation, minimise the number of evaluations that have to be done, and try to cut down the size of your spreadsheets.

Q. DETOUR.BAS

I have entered DETOUR.BAS from Release 2.7, and it works ... except when I tried to incorporate it into a batch file:

```
CACHE
MSBASIC.COM
LOAD "DETOUR.BAS",R
SC3
```

This stops after the second line with 'bad file or command'. Why?

A. Two reasons — you don't use the suffix when loading a .COM file. That's the reason for the 'bad file' message. But even if you omitted the .COM, it wouldn't work: LOAD is a Basic command, which won't be recognised in a batch file. The way you've written it here, the batch file processor is looking for a program called LOAD.COM or LOAD.EXE and not finding it.

The correct way to load MSBASIC and run DETOUR.BAS from a batch file is simply ...

```
MSBASIC DETOUR.BAS
```

In this case, you could rewrite the batch file to read ...

```
CACHE
MSBASIC DETOUR.BAS
SC3
```

Q. Colour on the Portable

My Portable set-up includes a Microvitec Cub 653 colour monitor. I nearly fell off my chair when the IANKEY disk I bought from you sprang into life with full colour — apart from the SKETCH program running under the atrocious ACTIVITY, I have never been able to achieve anything other than white on black. Any hints for the ignorant?

A. Obviously your software must support colour, otherwise you'll just get a white on black display. But it's been possible to get colour on Apricots in a standard way since BIOS 2.7. You have to use escape sequences to write to the screen, and it is a little complex; it's documented in the *Technical Reference Guide*, but with Apricot's usual clarity ... So see Part Two of David Langford's *Complete Escape Codes* in this issue.

Basically, there are 16 colours available; and if you are using 80 columns you can have up to four of them on the screen at one time. They have the following codes:

Black	0
Blue	1
Green	2
Cyan	3
Red	4
Magenta	5
Brown	6
Light Grey	7
Dark Grey	8
Light Blue	9
Light Green	:
Light Cyan	;
Light Red	<
Light Magenta	=
Yellow	>
White	?

Unfortunately you have to send the colour codes as an index, rather than directly — you have to set colours for index codes 1, 2, 3 and 4 (as foreground or as background). The defaults are pretty useless:

1	Light Blue
2	White
3	Black
4	Light Red

If you don't want the defaults, you must tell the palette which colours you want to use for which index number. To select yellow on blue (usually a readable combination) instead of white on black, you need to send the following string to the screen to set the palette (ESC of course means the Escape code, which in ASCII is 27 decimal or 1B Hex):

```
ESC ] 1 > ESC ] 2 1
```

Or if you are doing it in a Basic program ...

```
PRINT  CHR$(27) + "]" + "1" +
       ">" + CHR$(27) + "]" +
       "2" + "1"
```

In both cases the command line sets the palette (with the ESC] or CHR$(27) + "]" sequences) so that index "1" is colour ">" (yellow from the list above); and then sets the palette again so that index "2" is colour "1" (blue).

If you are using a program like WordStar, you could use the package's installation program to put these codes in as an initialisation string. That will set things up to use your chosen display colours.

And you can send the string to print your colours in the foreground and background:

```
ESC 5 1 ESC 6 2
```

This sets the foreground (ESC 5) to be our index colour "1", and the background (with ESC 6) to be our index colour "2". Again, the Basic equivalent would be ...

```
PRINT  CHR$(27) + "5" + "1" +
       CHR$(27) + "6" + "2"
```

You must send a string like this every time you want to change the screen colours. If you are installing WordStar you might use one colour combinations for starting highlight and another for ending it (returning to normal).

Don't forget that you can only have four colours on the screen at once, which is quite limiting. In addition it may be a good idea to start your program by saving the current environment and resetting the colour palette to the default ...

```
ESC ' ESC xG
```

Or ...

```
PRINT  CHR$(27) + CHR$(96) +
       CHR$(27) + CHR$(120) +
       CHR$(71)
```

And when you're done, you can then restore the environment to what it had been before with ESC a or PRINT CHR$(27) + CHR$(97).

You can have a lot of fun experimenting with persuading your software to use different colours sensibly. But avoid colour combinations which can be easy to program but hard to read — such as white on white! ◆

Doodles by David

Tips and Tricks

The occasionally essential column

By David St.John Wallis

To wrinkly subscribers and devotees of this highly contrived column, this will be older than blocking hats (yes, Woody Allen does make me cry in the Tube) ... But I think it is nice to have a clean root directory — preferably with nothing in it at all. At present mine has this:

```
Volume ID for drive A: is OMR-LIMITED
Directory of  A:\
PROGRAMS      <DIR>       9/04/87   18:21
HUNGARY       <DIR>      22/05/87    1:10
SCANNERS      <DIR>      27/05/87    0:36
1        BAT       256  23/05/87    0:00
 4 File(s)    5750784 bytes free
```

"Why?" I hear you cry. To which I answer, "Ignore most pundits and magic things you don't need to look at — like COMMAND.COM and AUTOEXEC.BAT and the legendary CONFIG.SYS sitting next to Rajiv Gandhi in row G.

A chap said faintly ...

... to our answering machine: "What's all this RAM stuff, and BIOS whatsits, and do I need it, and call me back at 0101 892 463255". The likes of D Langford and R Gann will know *all* about this kind of phone call. This chap also won over my heart completely by quoting one of our two-year-old ads and announcing in a brown ale voice that he no longer subscribed to APRICOT FILE.

I wanted to say "RAM is how you drive it, BIOS is how it's driven and if you want to know more send fivers quickish to APRICOT FILE since chaps who won't pay £650 a year to read my stuff aren't worth a barrel of Dutch herrings". But I thought it was sunny enough to plant grass instead.

Punters please note: if you want support from your dealer, friend, supplier or whatever, please be a bit more precise — and give us some reason to want to deal with your enquiry properly. Not all dealers are crooks, cheats, bankrupts and/or convicted of bestiality with Hoovers. But they do have to try to pay for their phone calls ...

On missprints

These devastating gems go to APRICOT FILE via Telecom Gold e-mail and using (would you believe) Communiqué (but any comms program will do). Here's the sequence of things.

Type COMMUNIQ. Type '1' and '1', then Return. (This assumes you're properly parametered into Telecom and that you've numbered it '1'). Sit about for a day or two. At the 'Send, Scan or Read' message, go 'S' and Return. The remote computer asks you who to send your wisdom to. You go, turgidly, '81:JET767' — that's the APRICOT FILE mailbox number. In the manner of *The Prime of Miss Jean Brodie*, you're asked the subject. You can type in Thingy. Then you get to send the text.

Now, if you've written the text using Superwriter, make sure you've saved the magical hieroglyphs in SYSTEM format (if WordStar, use Non-Document format). So at the 'Text' prompt you thud the Load key on the MicroScreen and the program asks you for a file name. You type that and away it goes; at the end you type 'S' to send it. The satisfaction is too much ...

A whimper ...

... if not a snivel was made by my hard disk on the eve of a thunderstorm and just after the computer's home was transferred a few miles. Could it be the clouts it got? Could it be the incipient storm?

Swift release of the lid told me all the engines were in place but one was loose and I was low on brake fluid. Yes, t'was the hard disk controller card (that's the one with the grey squashed worms entrailing all over the carcase). The removal of same, the squirting of the socket with WD40 and the replacement was but the work of a moment; and on the seventh day one saw what one had done and was mighty smug. Later the storm broke in all its fury and my duck drowned.

The other readership survey

To that man who said less of St.John Wallis, I say "a drat upon your roof tiles". I can take any criticism just so long as it's obsequious and adulatory. You try churning this stuff out for an occasional lunch down the *Old Bell and Bucket* which you can't attend because of pressing matters with the silage effluent.

Wont to give a helping hand to all and sundry, when *Practical Computing* requested 'panel members' one thought one would volunteer. What a snare and a delusion! Now while APRICOT FILE has got it all wrong by selecting readers who criticise me in a negative and unconstructive fashion, *Practical Computing* has got it a lot wronger.

Their idea is to send out a free magazine and a questionnaire. The bits they want you to read, they say, are marked pink — but they chose to stick great big pink labels right over the very text upon which one was keen to opine. The error was grievous and one marvels at how the statisticians are going to wade through all those 'extremely interested' to 'bored me stiff' scores in ignorance that the chaps filling in the forms were guessing at whether the articles were any good or not.

The date

I was charmed to receive the Official Apricot Date Upgrade Disk with NEW_BIOS2.7. But lo! It failed to work and I failed to suss it. Back to the reliable old nine-finger exercise ... The moral in this story: don't get in an uproar and don't send the money *(except to* APRICOT FILE. *As indicated in Release 2.9, we have a fixed version of the fix available to anyone who cares to mail us a fiver, a disk, and an SAE — Ed).*

Incidentally, for those prone to notice a certain green-welly tone in this article, send your abuse direct to my new address — **The Old Stores, Arford, Hampshire, GU35 8BU.** Or maybe not. ◆

Reflections on AI

Why use a special language?

By Mark Whitehorn

In APRICOT FILE Release 2.6 I tried to define Artificial Intelligence (AI) and Expert Systems (ES) in user-friendly terms. I suggested that systems which display AI often have one or more of the following characteristics:

- they behave like an intelligent human
- they use heuristics (learning based on past experience — an approach to problem-solving which, while not guaranteed to solved it, will generally make a solution more likely)
- they learn from experience
- they have large databases

To most programmers it must be clear that all of these characteristics can be produced in conventional programming languages such as Basic or Pascal. Take for example that most elusive of characteristics — behaving like an intelligent human. Many of you will be familiar with ELIZA; originally written by Joseph Weizenbaum in 1966 at MIT, this program supposedly mimics the responses of a Rogerian psychotherapist — ELIZA is capable of receiving input such as "I feel very depressed" and responding with something like "Why do you feel depressed?".

I make no pretence that this program could fool anyone for very long. ELIZA is incapable of understanding complex sentences; and given any input which it fails to understand, the program tries to bluff its way with a response like "I see" or "Tell me more".

Still, it's clear that ELIZA can mimic human responses in a limited fashion; and on the surface it would appear that a little development work could improve ELIZA until it was capable of fooling people.

Which language?

Versions of this program have appeared in a variety of languages — I know of at least two in Basic and one in Pascal. So why have new programming languages been developed for AI when we already have languages which could do the job?

There are two answers to this. First, AI *could* be developed in Basic; but in practice this would be very difficult because the language was not designed with the above-mentioned requirements in mind. For example, getting Basic to learn from experience is possible but unwieldy — the language has no commands specifically for this purpose.

I said earlier that ELIZA could apparently be improved to a stage at which the program might fool people into thinking that it was a real person. In practice, any attempt to improve ELIZA runs into the law of diminishing returns: ELIZA gets bigger and bigger and bigger (and slower and slower) without getting very much better.

The second answer is that once the decision is made to design a language specifically for the development of AI, new features (like learning) can be built in; and the whole way in which programming is performed can be adapted to suit the rather specialised task required of the language.

So you're convinced that new languages are, if not essential, at least a good idea for producing expert systems. The next question is — which language?

There are many ways of creating an expert system. They can be written in a language such as PROLOG or LISP, or they can be constructed using an expert system generator such as ESP Advisor or KEE. For the sake of simplicity I'll restrict this article to the languages; and as I know more about PROLOG, I shall completely ignore LISP from now on.

This is hopelessly narrow-minded, I know. LISP programmers out there are grinding their teeth and howling (in fact LISP programmers seem to do this a great deal — it probably has something to do with all the brackets they have to contend with). In fairness, LISP has been used to construct more expert systems than PROLOG has; and whole ranges of computers have been built whose architecture is designed solely to run LISP at impressive speeds. Without doubt, LISP is an excellent language.

I can however defend myself by pointing out that PROLOG is the prototype language selected by the Japanese for their Fifth Generation Computer Project. What's more, PROLOG is available for Apricots while (as far as I know) LISP is not.

PROLOG stands for PROgramming in LOGic; it was developed by Alain Colmerauer and the Groupe d'Intelligence Artificielle around 1970. Its design was an attempt to allow the programmer to specify the task in logic rather than worrying about telling the computer exactly what it should do at each stage.

Why are AI languages so different?

Languages such as Pascal can be described as *procedural*: the programmer writes out a detailed set of instructions which describe exactly which steps must be followed in order to solve the current problem. In his *Introduction to Knowledge-Based Systems*, R A Frost puts this very well: "the flow of control through the program statements is expressed explicitly by the programmer". (I rather wish I had written that, but since I didn't I thought I'd quote it instead.)

On the other hand, languages like PROLOG are *declarative*. Instead of specifying the way in which the problem is solved, the programmer enters a set of facts and rules (collectively known as 'clauses') which describe both the problem and the information which will be required to solve it. The *language* then solves the problem. The actual way in which this is done, that is "the flow of control through the program statements", remains under the control of the language.

This is not as sinister as it sounds. The language will use the clauses in a way which is well defined, but the programmer is to a large extent freed from the worry of exactly how the program needs to be structured.

A quick example may help to clarify the situation. Suppose I wish to construct a family tree of the Whitehorn clan. I could regard the following as facts which the program might require (examples of the PROLOG syntax are shown in each case):

The sex of each member of the family:

```
male(mark).
male(simon).
male(john).
male(james).
female(grace).
```

The parents of a group of siblings:

```
parent(mark,john,grace).
parent(simon,john,grace).
parent(james,john,grace).
```

The astute will have noticed that these last three lines contain more information than at first meets the eye. Taken together, they contain the information that Mark has two parents; that his father's name is John; and that his mother's is Grace. Further, since Mark, Simon and James have the same parents, the three of us must be siblings. In addition these rules provide information about other people, for example, my mother. We can see that she has a husband called John, three children, and so on.

PROLOG can be used to recall both the obvious and the less obvious facts from its program. For instance, I could ask PROLOG for the name of my mother using the command ...

```
?-parent(mark,john,X).
```

PROLOG will respond with:

```
X=grace
```

We can however write rules in PROLOG to link facts and extract the less obvious information. These rules are often written in a form reminiscent of the logic commonly used by people: as long as I know that person X is male and that persons X and Y share the same parents, I will also know that person X is the brother of person Y. So if I want PROLOG to find my brothers, I write a rule telling PROLOG to

- find a person who is a male
- look to see if that male has the same parents as myself
- and to make sure that it hasn't found myself as my own brother.

In PROLOG this would look like this:

```
brother(X,mark) :- male(X),
        parent(mark,Father,Mother),
        parent(X,Father,Mother),
        not X==mark.
```

I can rewrite this more generally to find anyone's brother:

```
brother(X,Y):- male(X),
        parent(Y,Father,Mother),
        parent(X,Father,Mother),
        not X==Y.
```

The strength of the language becomes apparent when you realise that rules can be built up from rules and the whole structure can rapidly become very complex, while the programmer need only think at any one time about fairly simple concepts. For example, the following rules are all perfectly valid — and yet the later ones all depend upon the early ones:

- a grandparent is a parent of a parent of someone
- a grandfather is a male grandparent
- a great-grandfather is a male parent of a grandparent

... and so on.

Having done all this I can ask such questions as 'Who are the grandfathers of my cousins?' and expect a sensible reply. Note that the system has never been given this specific information. Instead it can derive the information from the given clauses. This means that:

- once the rules have been written, they can be applied to any generation (so I can ask for my great-grandfather's cousin's uncle's sons);
- to add extra people I only need enter their sex, and the names of their parents. There is no need to specify whose second cousin twice removed they happen to be. Extra people can simply be added as they are discovered (or born) without upsetting the program structure.

One further characteristic of PROLOG is that, to a large extent, the programmer can include these clauses wherever he/she likes within the program. For a programmer used to procedural languages, this is rather unnerving at first; it feels as if you're relinquishing part of your control over the program, as indeed you are. After a while, though, you get used to it.

Are these languages any better?

To be 'better', they must be able to produce programs displaying the four given characteristics more easily than a conventional language.

In the main, this is true. Intelligent behaviour can be more easily programmed in PROLOG (as can heuristic approaches to problems) because of the very complex responses which can be produced. This complexity is based on the ability to write rules which depend on rules which depend on rules ...

Learning from experience is easy to program in PROLOG because you can write rules which in turn write facts and rules into the program itself (under specific circumstances). Thus the program can grow and evolve with time, gradually acquiring experience.

Finally, a PROLOG program is itself a large database — the very nature of the program is a collection of facts and rules that make up a database. With this type of language, in fact, the distinction between program and database is difficult to make. Are the lines of code quoted in this article program or database information?

These declarative languages are very different from procedural languages, but don't let anyone tell you that they are fundamentally any more difficult to learn. The time invested in getting to grips with the language are amply repaid by the discovery that you have seventeen first-cousins-once-removed ... ◆

Mail-Order Bargains — Update

New from Deals For Readers

ACOLYTE prices

Due to circumstances beyond our control (the sub-editor's hangover, probably) the most recent APRICOT FILE DEALS FOR READERS mailer does not give the correct price for Ansible's excellent ACOLYTE disk. The correct details are as follows:

ACOLYTE	
Order no:	AS035
Price:	£57.50

Cable connections

Male-to-Male Gender Changer	
Order no:	AC001
Price:	£10.00

Solid, simple, fully-shielded adapter that plugs into the RS-232 serial port before you attach the cable — thus allowing a female RS-232 connector to plug into a female socket.

Female-to-Female Gender Changer	
Order no:	AC002
Price:	£10.00

Allows a male RS-232 connector to plug into a male serial port socket.

RS-232 Mini-Patch	
Order no:	AC003
Price:	£27.50

One of the cheapest, safest break-out boxes we've found. Plug-in 'jumper' wires allow you to connect any combination of incoming and outgoing RS-232 lines; male connector at one end and female at the other, so there's no problems with attachment. Use it with AC004 and you'll have the equivalent of a full-scale break-out box at modest price.

RS-232 Data Tracker	
Order no:	AC004
Price:	£28.00

Simple plug-in unit with eight pairs of LED indicators to monitor negative and positive signals on each of the most commonly used RS-232 lines — 2 to 6, 8, 11 and 20. Use it by itself or with AC003.

RS-232 Surge Suppressor	
Order no:	AC005
Price:	£28.00

Voltage fluctuations and interference caused by static build-up, wobbly mains, power supply failure or even lighting can damage equipment or garble RS-232 transmissions. This neat, robust and simple plug-in unit suppresses all voltage spikes above 26V.

Hardware

8087	
Order no:	AH007
Price:	£120.00

The genuine Intel maths co-processor to speed up your Apricot PC/Xi, F-Series, even the Portable! Easily fitted — it just plugs in to a vacant socket. Comes pre-fitted to a socket mount to avoid damage during installation. Make sure your software is capable of addressing it; MS-Fortran, Turbo Pascal, etc.

Dataflex Stradcom Modem	
Order no:	AH008
Price:	£200.00

For the Xen-i — the only IBM-style Hayes-compatible internal modem that fits. Offers 300/300 (V21) and 1200/1200 (V22) speeds, choice of pulse or tone dialling, phone socket, auto-dial/answer, monitoring speaker. Comes complete with special customised version of Procomm called Baudwalk.

FD5AI 5.25in Drive	
Order no:	AH009
Price:	£350.00

An external 5.25in IBM-format floppy disk drive with interface card for the Apricot PC/Xi — enables you to transfer data (read/write and format) to and from IBM PCs. From Plus 5. Normal RRP £402.50

FD5XI 5.25in Drive	
Order no:	AH010
Price:	£295.00

As AH009, but for the Xen — and not the Xen-i. RRP £339.25

FD3XA 3.5in Drive	
Order no:	AH011
Price:	£295.00

An external 3.5in disk drive for the IBM-compatible Xen-i. Can read and write the old and new Apricot 3.5in disk formats, too. RRP £339.25

FD3A 3.5in Drive	
Order no:	AH012
Price:	£250.00

Add an external 3.5in drive to the Apricot Portable. Not suitable for F-Series. RRP £287.50

Books

Our prices for books include postage within Great Britain. For overseas orders, contact us first to confirm the total price.

Using WordPerfect	
Order no:	AB004
Price:	£19.00

Beefy paperback for newcomers to WordPerfect — based on Release 4.0 and so an ideal starter for users of any version. IBM-oriented, of course, and despite the whopping price we don't make much money on it; still, it's produced by the people at Que Corp — probably the best documenters of software outside Microsoft Press — and is widely regarded as the very best intro to what must be the most classy WP package now available on the Apricot.

WordPerfect Tips, Tricks and Traps	
Order no:	AB005
Price:	£19.00

Masterly follow-on to the above, again from Que — hundreds of tips and lots of advice in quick-reference style for more advanced users. Based on WordPerfect 4.2 — and on American usage (especially in the applications examples), but it's still a mine of information.

Turbo Pascal Tips, Tricks and Traps	
Order no:	AB006
Price:	£19.00

Another massively competent super-size paperback from Que; widely regarded as the best toolkit of Turbo Pascal advice currently available.

Running MS-DOS	
Order no:	AB007
Price:	£21.00

Van Wolverton's Microsoft Press all-purpose introduction to MS-DOS is the best we've seen. It is certainly the clearest; as much care has been taken in the style and presentation as with the content — which is pretty comprehensive. Covers MS-DOS versions up to 3.1, indicates most of the functions and facilities specific to IBM PC-DOS. If you want only one book about MS-DOS, buy this one — the

average Apricot user simply won't need another MS-DOS reference.

Advanced MS-DOS	
Order no:	AB008
Price:	£21.00

Another hefty number from Microsoft Press, aimed at the more experienced Assembler or C programmer. Explores the functions and special features of MS-DOS in detail, without the dense and/or terse style that so often characterises the knowledgeable author; "a book that was written to be read" as our review in Release 2.9 puts it. Includes a complete reference to DOS interrupts.

Modem Connections Bible	
Order no:	AB009
Price:	£17.00

Sensible, pragmatic explanation and survey of connection between computers and modems — one of the best introductions we know, in fact. Includes a section giving detailed and very clear illustrations showing how to connect popular micros to some of the better-known US modems; it's a shame that no-one seems to have done the same for British products (but such a project is under way at COMMS FILE right now).

RS-232 Made Easy	
Order no:	AB010
Price:	£21.00

Very nearly lives up to its title — covers the how and why of attaching printers, terminals and modems while assuming no prior knowledge of the RS-232 interface. Includes lots of explanation, details of real-world products (not all of which are US-only), and numerous reference summaries.

Supercalc Prompt	
Order no:	AB011
Price:	£8.50

The latest edition of Randall McMullan's excellent intro to SuperCalc and its commands — and just for a change, it's a recommendable British book. Based on SuperCalc 3.

Apricot Extras

MicroScreen keystrip	
Order no:	AH016
Price:	£25

Problems with a duff MicroScreen keypad? The likely cause is deterioration of the el cheapo contact-keypad strip — and we can offer a quick 'n' easy DIY replacement, with simple-to-follow fitting instructions.

Apricot-format software

WorkMate	
Order no:	AS050
Price:	£55

The only SideKick-style memory-resident 'desktop utilities' package we know of for the Apricot — includes Calendar with Appointments Diary, alarm clock, and 'to-do' list: Address Book with automatic dialling via modem for voice communications: ASCII code summary.

SpellBinder 5.4	
Order no:	AS100
Price:	£290.00

A very powerful word processor, one of our favourites, boasting many features — easily customised for any printer, includes spelling checker. RRP £396.75

WordTech DBXL	
Order no:	AS101
Price:	£125.00

A dBASE III+ clone for a quarter of the price! File structures, syntax, commands and indexing are fully compatible with dBASE III+; they run identically, too. Features menu-driven options and 'Tutor' line for novices. Brilliant value.

IBM-format software

SpellBinder 5.4	
Order no:	SI200
Price:	£290.00

As AS100 above, but in IBM-format. Specify disk format as 5.25in or 3.5in when ordering.

WordTech DBXL	
Order no:	SI201
Price:	£125.00

As AS101 above, but in IBM-format. Specify disk format as 5.25in or 3.5in when ordering.

Flight Simulator	
Order no:	SI154
Price:	£50.00

The latest version (2.13) of the original — and still the best! Not yet available in 3.5in format.

Psion Chess	
Order no:	SI039
Price:	£25.00

As the Apricot version, but now in IBM-format. Not yet available in 3.5in format.

Norton Utilities 4.0	
Order no:	SI155
Price:	£80.00

The latest version of the best disk editor and housekeeper for the Xen-i user — includes the essential undelete facility. Will also run on old BIOS 3.1 Apricots under the IBM emulator; phone for details.

CardBox Plus 3.5	
Order no:	SI044
Price:	£280.00

IBM-format version of AS044.

WordStar 4	
Order no:	SI159
Price:	£300.00

Latest (and much-improved) version of the old favourite in IBM format.

Borland — IBM format

Turbo Lighting	
Order no:	SI150
Price:	£69.00

Turbo Basic	
Order no:	SI151
Price:	£69.00

Turbo C	
Order no:	SI152
Price:	£69.00

Turbo Prolog	
Order no:	SI153
Price:	£69.00

Turbo Pascal	
Order no:	SI048
Price:	£69.00

All our prices include VAT and despatch (except for overseas mailing of books: see the relevant entries). Your order will arrive with a VAT invoice/receipt. There are three ways to pay:

- **Girobank transfer:** Our National Girobank account number is **59 028 1909**
- **Cash with order:** A cheque drawn on a UK bank, a Sterling money or postal order, or a Eurocheque can accompany your order (make cheques payable to "TP GROUP" and cross them)
- **Credit card:** We can accept **Diners Club, Access** (*aka* **Mastercard**), and **American Express** (*but not Barclaycard/Visa*). For a payment we need a signed letter that includes your name as it appears on the card, your card number, and its expiry date.

Escape plans (continued)

All about ESCape codes: Part Two

By David Langford

The proposition that 'sending an ESC sequence to the screen' is merely a legal fiction can be clearly seen from the fact that keyboard and printer functions may also be activated in this way at the whim of the BIOS.

Keyboard specials

Taking the keyboard first, we'll include all those ESC sequences which pop characters into the keyboard buffer to be read by your program as though you'd typed them in yourself.

ESCape	Dec	Hex	Function
ESC $	36	[24]	put character at cursor position into keyboard buffer — ie read from screen
ESC 4anm	52	[34]	program a key — a four-character sequence where a is '1', '2' or '3' to select normal, shift or control setting; n is the key code number; and m is the new character
ESC Z	90	[5A]	'identify as VT52 terminal' by putting the enigmatic sequence ESC / K into the keyboard buffer so that programs can read it
ESC \n	92	[5C]	put character n into keyboard buffer
ESC n	110	[6E]	report current cursor position to keyboard buffer in the format ESC Y yx (compare the 'move cursor' sequence above) where y is row+32, x is column+32

(As before, the underlined parameters are variables for you to enter — anything else should be typed as it appears.) ESC 4 has been found particularly difficult by users. Two minor pitfalls: the first parameter a is not ASCII *code* value 1, 2 or 3 but ASCII *character* '1', '2' or '3' (codes 49, 50 or 51 decimal). And the new character m may *not* be a control character in the ASCII range 0 to 31 [0 to 1F].

The *major* pitfall is the obscurity of Apricot's key numbering, which makes sense only on the Xi with MicroScreen. For the purposes of this command the keys are numbered as follows:

- The eight labelled function keys on the Xi are numbered consecutively from 0 to 7. So 0 to 3 are HELP, UNDO, REPEAT and CALC (alias F1 to F4 on newer machines); 4 to 7 are PRINT, INTR, MENU and FINISH, alias F6 to F9. What happened to F5? See below ...
- The MicroScreen keys are numbered left to right from 8 to 13.
- The top row of the keyboard proper goes from 14 to 32 (the backslash is 14, the calculator's plus sign is 32).
- The next row is numbered 33 to 50 (TAB to the calculator keypad's '9').
- The third row down is 51 to 68 (CAPS LOCK to '6' — the Return key is counted in this group).
- The fourth row is 69 to 85 (SHIFT to '3').
- The bottom row goes from 86 to 95 (ESC to ENTER).

The F5 or VOICE key on later machines, though logically one of the function keys, had to be squeezed in at the end of this numbering as key 96: TIME/DATE, where present, is not part of the logical keyboard and cannot be programmed.

So to program the SHIFT position of the ENTER key to generate, say, an asterisk, you'd send ESC, then 4, then 2, then ASCII character 95 (underline) to indicate key number 95, and finally an asterisk character:

```
ESC 42_*
```

To check the current cursor position with ESC n, you first send that sequence to the screen and then immediately read four characters from the keyboard buffer. The first two of these (ESC Y) can be discarded; subtract 32 from each of the third and fourth to give row and column numbers respectively — counting from 0,0 at top left.

Printer specials

The printer has fewer complications:

ESCape	Dec	Hex	Function
ESC &	38	[26]	send whole screen to printer
ESC '	39	[27]	send current screen line to printer

Notes:

- ESC & is generated by the famous Apricot PRINT key, which should be used with SHIFT in BIOS 3.1 and later versions. Every character on the screen goes to the printer ... but annoyingly, a formfeed character is sent first, so that if you're hand-feeding the printer, the paper will scroll out before printing begins. Hang on to it grimly, or try one of the BIOS patches suggested in my column of APRICOT FILE Release 1.9.

■■ ESC ' omits the formfeed and automatically starts a new line on the printer after sending the text .

Cursor appearance, environment flags, and twiddly bits

ESCape	Dec	Hex	Function
ESC 2	50	[32]	set flashing cursor on Xi (not F series)
ESC 3	51	[33]	turn off Xi flashing cursor
ESC ‘	96	[60]	save 'environment'
ESC a	97	[61]	restore saved environment
ESC x*a*	120	[78]	set environment flag *a*, where *a* is as follows:
G	71	[47]	reset screen palette (F series only)
$	36	[24]	set Apricot-compatible mode (F series only)
1	49	[31]	enable status line for ESC Y access
4	52	[34]	set block cursor (Xi only)
5	53	[35]	turn cursor off
8	56	[38]	set auto LF on receipt of CR
9	57	[39]	set auto CR on receipt of LF
D	68	[44]	smooth screen scrolling (F series only)
E	69	[45]	LCD contrast up (Portable)
F	70	[46]	bell volume up (F series only)
ESC y*a*	121	[79]	reset environment flag *a*, where *a* is as follows:
1	49	[31]	disable status line for ESC Y access
4	52	[34]	set underline cursor (Xi prior to BIOS 3.1)
5	53	[35]	cursor on (cancels flashing cursor, if any)
8	56	[38]	cancel auto LF on receipt of CR
9	57	[39]	cancel auto CR on receipt of LF
D	68	[44]	fast screen scrolling (F series only)
E	69	[45]	LCD contrast down (Portable)
F	70	[46]	bell volume down (F series only)

Notes:

■■ On machines where it works at all, ESC 2 has the side effect of turning the cursor on should it have been off.

■■ ESC ‘ is supposed to save the first three environment flags for later restoration by ESC a. In theory these are Screen Palette, Xi Compatibility Status (F series only: see under 'Colour') and Status Line Enable/Disable (all machines): this is why ESC xG for the palette appears oddly out of sequence in the above, because that's where Apricot's documentation puts it. In practice I find that some details of cursor appearance are also saved (whether it's on or off, and its 'flashing' or 'underline' status on the Xi prior to BIOS 3.1); and that the information is, unusually, preserved even through an ESC z screen reset.

■■ ESC a can produce odd effects if invoked before ESC ‘ has been issued: the cursor may unexpectedly vanish, or become a flashing underline cursor where the machine and BIOS allow this ...

■■ The ESC x8 and ESC x9 sequences are occasionally useful when listing text files in offbeat formats (for instance, lines separated only with linefeeds and paragraph ends marked with solitary Returns, as in SuperWriter). By forcing the screen to insert an automatic linefeed when it receives a Return, or *vice versa*, we can be sure of getting both — a properly set-up new line, in other words — even if only one of them is present.

■■ ESC xD allows you to take advantage of the F series machines' graphics display for particularly posh and leisurely effects: with 'smooth scrolling' the screen will scroll by much tinier vertical steps, individual graphics lines rather than whole text lines. Horizontal scrolling isn't affected.

■■ ESC xE and ESC yE are produced by the Portable key combinations which let you adjust the liquid crystal display contrast from the keyboard.

■■ ESC xF and ESC yF don't seem to make the slightest change to the bell volume of my F10; I can't help wondering whether this facility is restricted to, say, the Portable.

The Calculator and MicroScreen

ESCape	Dec	Hex	Function
ESC /*n*	47	[2F]	turn on MicroScreen LED lights in pattern *n*
ESC <	60	[3C]	display date and time on MicroScreen
ESC ?	63	[3F]	activate built-in calculator
ESC U	85	[55]	turn on dual output to MicroScreen
ESC V	86	[56]	turn off dual output to MicroScreen
ESC W	87	[57]	send display output to MicroScreen only
ESC c	99	[63]	disable MicroScreen scrolling
ESC d	100	[64]	enable MicroScreen scrolling
ESC e	101	[65]	turn MicroScreen cursor on
ESC f	102	[66]	turn MicroScreen cursor off
ESC g	103	[67]	disable MicroScreen time/date display
ESC r	114	[72]	enable MicroScreen echoing of main display output
ESC s	115	[73]	disable MicroScreen echoing (restores date/time)

Notes:

■■ ESC /*n* turns on a pattern of LEDs corresponding to the binary representation of the character *n*. For example, ESC /? will turn all six lights on: the question mark (ASCII character 63) is 111111 in binary. ESC /@ will turn them all off again:

character 64 (the '@' mark) is 1000000, and only the last six binary digits are important. The rightmost bit controls the leftmost LEDs.

■■ ESC ? is generated by the CALC key (SHIFTed position in later BIOS versions). Apricot has thoughtfully not provided an explicit ESC sequence for turning the calculator off again.

■■ ESC U and ESC r slow main screen handling disastrously.

The F-series Colour Supplement

To begin with, here's the basic table of colours. Although ASCII decimal and hex codes are given, these aren't escape sequences — just values which can be used in ESC 5, 6 and] sequences below. The palette and colour codes may look just the same, but read on ...

Default palette index	Colour	Colour code	ASCII value	Hex value
0	Light Blue	9	57	[39]
1	White	?	63	[3F]
2	Black	0	48	[30]
3	Light Red	<	60	[3C]
4	Light Green	:	58	[3A]
5	Yellow	>	62	[3E]
6	Light Cyan	;	59	[3B]
7	Light Magenta	=	61	[3D]
8	Blue	1	49	[31]
9	Light Grey	7	55	[37]
:	Dark Grey	8	56	[38]
;	Red	4	52	[34]
<	Green	2	50	[32]
=	Brown	6	54	[36]
>	Cyan	3	51	[33]
?	Magenta	5	53	[35]

In ESC]ac below, 'c' for 'colour' can be any of the 16 available colour codes from the table. But there is a complication. 'Set foreground colour' and 'set background colour' do not directly set a colour code: they set a palette index, as shown on the left of the table. The foreground colour is normally palette code 1 and the background is palette code 0.

The codes for colour values are reliable, but the palette index values are dodgy in the extreme. They can be reset as explained below; different background and foreground defaults may well have been read from your boot disk's 'label' sector; and the sixteen apparently available palette codes are in fact telescoped down to the number of colours currently allowed.

For example, in the normal 80-column two-colour display (Xi-compatible) with black and white as palette codes 0 and 1 (background and foreground), then attempting to use palette codes 2, 4, 6 and so on will give black while 3, 5, 7 etc give white.

You can change the palette setting with ESC] but you're limited to two on-screen colours... or more if multi-colour display is set with ESC 71.

The overall message: don't rely on default palette codes, and always use ESC] to ensure that the colours you want are available.

ESCape	Dec	Hex	Function
ESC 5a	53	[35]	set foreground colour to palette code a
ESC 6a	54	[36]	set background colour to palette code a
ESC 7a	55	[37]	set screen colour mode according to a, as follows:
0	48	[30]	80-column two-colour display on F machines, 80 column liquid crystal display on Portable, both with Xi monochrome compatibility
1	49	[31]	80-column four-colour display on F machines, eight-colour on Portable
2	50	[32]	40-column four-colour display on F machines, 40-column LCD on Portable
3	51	[33]	80-column two-colour display as mode 0 on F machines, 80-column two-colour display on Portable
ESC]ac	93	[5D]	set palette code a — an ASCII code from 0 (ASCII 48 [30]) to ? (63 [3F]) — to colour value c. For example, in a default two-colour screen ESC]01 could set a blue background, ESC]18 a light grey foreground — and so on, depending on the number of colours available in the palette

Notes:

■■ As with Attribute settings, the foreground and background selected with ESC 5 and ESC 6 will be the default for future character output until further notice. Changing foreground and background this way requires the setting of multi-colour mode with ESC 71: it's more reliable to control foreground and background by changing palette indexes 0 and 1 with ESC]. Example: ESC]04 sets a red background by changing palette colour 0 to red.

■■ ESC 7 can apparently cope with Portables having two monitors, both LCD and colour — it seems that ESC 70 selects LCD and turns off the colour display cursor, ESC 72 selects 40-column LCD similarly, and ESC 73 allows a dual LCD and colour display with the LCD cursor turned off.

■■ ESC 7 commands also reset screen windowing and home the cursor.

■■ Note that the ESC 71 multi-colour display setting disables the Xi-compatible attribute controls: underline, bright/dim, inverse video, etc.

■■ Interesting effects can be obtained from 'grey-scale' colour selection on many monochrome F series

machines. It's fun to experiment with this: pretend it's a colour monitor and see what effects colour change controls produce. You can find background/foreground combinations which are easier on the eye than the default setting.

■■ ESC] palette changes will retrospectively alter the screen colour appearance, very quickly: without changing the palette codes which have been set in various parts of the screen (like character attributes), it lets you instantly change their meaning.

■■ ESC xG — from 'Environment' above — instantly restores a standard default palette.

■■ ESC z screen resetting doesn't change the palette, nor the current screen colour mode as set with ESC 7. But it does reset the background and foreground colours to those set in palette codes 0 and 1 (if they've been changed with ESC 5 and/or ESC 6).

Miscellany

ESCape	Dec	Hex	Function
ESC [	91	[5B]	lead-in for ANSI escape sequence
ESC |	124	[7C]	not officially used, but it turns on the compressed text mode of the Black Hole 132-column screen driver when this is installed on the Xi

ANSI escape codes are not covered here because the article is too long already. Essentially they are cumbersome (having been devised by an American committee) ways of doing some of the things the VT52 codes do more efficiently. For example, 'clear screen' translates from ESC E to the ANSI sequence ESC [2J.

More recent versions of the Apricot BIOS support ANSI codes with a vague eye to IBM-compatibility; but Peter Norton's authoritative books on IBM programming strongly discourage use of these codes because (a) such programs will only run when the optional ANSI.SYS device driver has been installed in CONFIG.SYS, an extra and intimidating setting-up step for inexperienced users; and (b) ANSI.SYS operations are terribly slow.

IBM programmers tend to use specialist BIOS interrupts or direct hardware access even for simple things like clearing the screen. That accounts for the general non-portability of major IBM programs to Apricot format, and the very limited success of Apricot's 'IBM emulator'. (On the other hand, it's quite possible to write an Apricot emulator for IBM-compatibles: I know this because I've written one.)

Finally, some snares, delusions and officially inoperative sequences:

ESCape	Dec	Hex	Function
ESC #	35	[23]	once documented as 'send screen to keyboard buffer'
ESC %	37	[25]	originally documented for Xi as 'send current line', but has never worked
ESC X	88	[58]	Sirius line-moving command, never available for the Apricot: swapped the current screen line for one held in an internal buffer
ESC]	93	[5D]	originally documented for Xi as 'send status line' but never actually worked (see under 'Colour' above)
ESC _	95	[5F]	originally documented for Xi as 'toggle debug mode' whereby 'hex codes for print stream' were supposed to appear as if by magic on the status line — but see 'Attributes' above for what it actually did
ESC m1nm	109	[6D]	originally documented for Xi as 'set width for individual character cells', with n-32 and m-32. Never seems to have worked
ESC m2nm	109	[6D]	originally documented for Xi as 'set top and bottom lines for the screen', with n-32 and m-32. Never seems to have worked, though this function is available for windowing in the form of ESC ,
ESC {	123	[7B]	originally documented for Xi as 'disable keyboard'
ESC }	125	[7D]	originally documented for Xi as 'enable keyboard'
ESC £	156	[9C]	representative of several ESC n codes (where n was a character with an ASCII value above 127 [7F]) which used to crash the Xi: as of BIOS 3.1 they tend merely to make the cursor vanish until another character is sent

Notes:

■■ In earlier, bug-ridden versions of David Langford, ESC # was said to crash the Xi. While checking ESC sequences for this article, I found that this must have been one of those common keyboard/printer confusions: the black spot should have been handed to ESC £.

■■ 'Originally documented for Xi' codes are often Sirius controls which never got implemented on the Apricot. Inoperative escape sequences are generally harmless, and this also applies to those not applicable to the current machine: for example, you can try to program a MicroScreen key on an F series machine and there will be no effects or complaints. ◆

Next month — the third and final part of this opus, a quick-reference tabular summary of the Complete ESCape Codes.

Readers' Writes

Your chance to have a say

Write On

The Readers' Writes page is your chance to share your thoughts with other subscribers — we'll print any and all comments, queries, complaints, compliments, tips that we feel may be of general interest. Letters might be edited for grammar and length, but in general we print more or less exactly what we receive.

Send your letter to us via electronic mail (Telecom Gold 81:JET767) or by land mail to:

APRICOT FILE
TP Group
PO Box 509
London N1 1YL

SuperCalc consolidation

The SuperCalc manual states that "a blank spreadsheet cell and a non-blank disk file result in the contents of the disk cell"; this does not appear to be the case. But there is a simple remedy. The first of four company Profit & Loss accounts loaded to a Consolidation file did not have one item included in the Group Company. A common matrix was used and the option +, after specifying the cells for Part-loading, for the other three companies — including the Group, which was loaded last. No value appeared in respect of this item but the totals were correct as these were in the matrix.

The solution is to insert a zero in any account on the matrix of the company loaded without an amount. The statement is correct if a blank spreadsheet cell has been activated.

E A A Kimber
Pinner, Middlesex

SuperCalc RAP Planner

Pension premiums, commonly known as 'retirement annuity premiums', carry tax deductions at the highest rate of income tax and have the added benefit of providing life assurance with relief. It can be a complex and time-consuming calculation for accountants to ensure that clients not only have full provision for income tax in their retirement but take every advantage of the tax concessions available in the premiums they make.

A recent advert in *Taxation Practitioner* showed a program for 'pension contribution planning at the touch of a key' at £201.25 inc VAT. I felt sure that SuperCalc could be used to create a Retirement Annuity Premium Planner and proceeded to prove this.

The RAP Planner presented here is a simple-to-use program that will carry out these calculations and then produce a smart report. I have checked it with data from fifty of our own clients and have found that it saves a considerable amount of time.

The Planner will calculate the maximum contributions that those who are self-employed or in non-pensionable employment may make in any one year, based on the net relevant earnings (as defined by section 227(5) of the Taxes Act 1970) and the optimum carry-forward of unused relief or relief of the unused amount against an earlier year rather than a later one. The table will carry unused relief back over the previous six years. This will enable you to see how much premium can be paid in the current year, and avoid paying excess premiums that carry no tax-relief benefit.

Whether you offer investment advice on personal pension plans or tax planning advice on pension contributions, the RAP Planner will cut down drastically on the time you spend on this exercise. You can even give the job to a junior member of staff to input the details of clients' relevant earnings; the program will automatically do the rest, and you can be assured of accurate results.

RAP Planner Worksheet

Row 1	heading — name
Row 2	repeating rule
F4	rate of relief in force
Row 9 to 13	net relevant earnings
B14 to G14	total earnings for each year — SUM (B9:B13)
R16 to G16	maximum relief available for each year — Row 14 * F4
Row 17	heading — name of insurance company
Row 18 to 21	premiums paid in each year
B22 to G22	total premiums paid — SUM (B18:B22)
C23	carry-back from 87/88 — B30
D24	carry-back from 86/87 — C30
E25	carry-back from 85/86 — D24
F26	carry-back from 84/85 — E30
G27	carry-back from 83/84 — F30
B28 to G28	relief available — SUM (B22:B27)
B29 to G29	relief used in year — IF(B16<B28,B16,B28)
B30 to G30	relief not used carried back to previous year — IF(B28<B29,0,B28-B29)

Your readers can either create their own RAP Planner based on this, send me a formatted disk and £10 for a copy of this worksheet, or pay £175.00 plus VAT to Butterworth and Co.

Norman D Mash
N D Mash and Co, Sheffield

GSX and GW-Basic

Contrary to popular belief (and to DEALS FOR READERS disk AP005), it is perfectly (well almost) possible to access all GSX functions from GW-BASIC.

The key lies in an upgraded version of GSXBAS.COM of 4,750 bytes against the 384-byte MS-BASIC version. This program was made availible in 1985 on a 'Black Hole' disk issued to upgrade all the educational F1s, and was accompanied by an unusually helpful booklet called GSX *Graphics on the Apricot Family*, subtitled 'an introduction to graphics using Basic by the ACT (UK) Technical Support Department'.

Unfortunately all the examples are in MS-BASIC, but there are no real problems in translation. Perhaps I can offer a few tips:

- To interpret the same program via both GW-BASIC and MS-BASIC, it must be saved in ASCII format
- Use any text editor to disable or replace any escape sequences in MS-BASIC PRINT statements; GW-BASIC displays all characters as literals but offers conversion functions for most MS-BASIC screen-handling sequences, such as CLS for PRINT CHR$(27)"E" — the main exception being the high/low intensity video toggles, which are unusable in GW-BASIC.
- Although many GSX driver functions are availible via GW-BASIC statements, use of the second (stroke) font and other device drivers such as printers and plotters require GSX calls as documented in the Digital Research GSX manual or the ACT booklet.
- The GW-BASIC interpreter calls some GSX functions directly and works entirely within a graphics environment, an OPENed workstation. So calling GSX from within GW-BASIC opens a workstation within a workstation! Unless you close the workstation within Basic in a tidy manner, some confusion results, notably with the GW-BASIC screen editor and cursor addressing. Before closing the workstation, you must reset the bar fill pattern to solid, reset the font, and clear the workstation, before any editing or direct mode use of GW-BASIC can occur.
- Any program calling GSX should include an ON ERROR statement to carry out the above functions. Where a device driver other than the normal display screen has been used, the display should also be reinstated. This prevents 'hanging' due to a programming bug leaving control in an output-only device such as a printer.
- Ensure that only sensible demands are put on device drivers to avoid silly results. Perversely, the ACT-provided drivers for screens support ten fill patterns and continuous text rotation and height, while the Epson printer drivers support only six fill patterns, provide text rotation in 90-degree steps and restrict maximum character height.
- Apricot's GSX release 1.6.3 (dated 8 March 1985) is considerably more extensive than earlier versions. Some escape functions unique to Apricot are provided, together with user-defined fill and line styles. Suprisingly the documentation is a reasonably comprehensible .DOC file of 22KB.
- The GW-BASIC LINE statement accepts values for the style parameter from -32767 to +32767 (i.e a bewildering 65,535 possible styles). Useful values are -1 (solid line) and 0 (no line or line in background colour)
- Attempts to load and obtain output parameters or output from inappropriate screen drivers cause havoc — don't try and load Reidel's PCFX.GSX to get a screen dump from a Portable or an F1 driver on a PC/Xi. Rebooting seems to be the way out of this one!
- A bug is present in the GW-BASIC LOCATE statement if you are writing generic Apricot programs. F series machines will return an error massage if the start or stop values exceed 7; PC/Xis will permit values of up to 15. A statement giving a full block cursor on a F series machine produces a half-size cursor on a screen with a 16-pixel character height.
- It is impossible to LPRINT CHR$(26) from GW-BASIC. Use the OUT statement and the machine-specific port number.
- Calls to the Apricot Control Device (at 60H) return the 'device not available at line *nnn*' message in GW-BASIC.

Frank Sendall
Hale, Cheshire

We have the full text of the GSX booklet as a file on DEALS FOR READERS disk AP020.

TAS plea — and fans

I am thinking of buying TAS for use in producing some simple accountancy records and I shall be most interested to hear of any readers' comments on this software — so far I have been unable to find anyone who has used it.

In 1984 I purchased an Xi and the only criticism I have of it has been the noise from the fan. At the beginning of this year I bought another Apricot and hoped that Apricot might have reduced the noise in the interim; but the new Xi is even more noisy than the old one, added to which it has a high-pitched whine which drives everybody in the office to despair after about two hours. Does anyone have a solution to this problem? Perhaps the answer is to buy one of the quiet fans being advertised in the last DEALS FOR READERS mailer; but does £25 really buy peace and tranquility?

J E Bullin FCA
Ford, Du Cane & Co, Suffolk

The Papst fans we sell for £25 as DEALS FOR READERS number AH015 are *low-noise* fans, but they aren't totally *silent*. Certainly they are quieter than the standard issue and should provide some improvement; but the Apricot casing wasn't designed for acoustic damping, so you can't expect *too* much ... ◆

The Apricot 'Control Device'

Complete control — or out of control?

By Nick Wilson

Way back in January 1983, when the first Apricot BIOS was being written, somebody had the idea that it would be useful to be able to change the baud and data information on the serial port by making a call to the BIOS.

Interrupt FC hex was designated for the purpose. The number was used simply because it was the first non-reserved interrupt number in the memory map as defined by Intel and Microsoft.

The first version just handled the serial port data information (baud rates, parity, bits per byte, stop bits). It was then quickly expanded at the request of the author of ASYNC to include character send and receive functions: and at this time it acquired the cognomen 'Serial Control Device'.

Adding more ...

Somebody else had the idea of including the keyboard auto-repeat rate function in the Serial Control Device. This was implemented: and it then became known as the 'Serial and Keyboard Control Device'.

This addition marked the start of a hectic period in which literally every configuration variable in the BIOS could be altered using interrupt FC ... The device was renamed as the 'Control Device'. It controlled a total of ten different devices and a whopping 86 function calls; and the vector at 0060:0000 hex was incorporated so that Basic could access the BIOS.

The Control Device was shipped in this way for the first two years. The Apricot BIOS had two entry points at this stage; the MS-DOS call vector (which is the entry point into the BIOS for MS-DOS), and the Control Device itself. About 20 per cent of the code and data size of the BIOS was dedicated to the Control Device, another 10 per cent to the MS-DOS interface.

... And more still

Given that history, it was inevitable that the decision would be taken to 'close the circle' of the BIOS — to make it an entirely self-contained unit that could be called from any operating system.

The main reason for this was the arrival of the F series computers; Apricot wanted desperately to have common boot disks — and of course, that wouldn't be possible unless the BIOS contained on the disks was machine-independent.

The independence was obtained by writing a small RAM BIOS which contained the code from the original BIOS for the MS-DOS entry point and a new 'back end' of nothing but calls to the Control Device to do all the I/O which MS-DOS required.

So now, MS-DOS calls the RAM BIOS (which is found on disk): and the RAM BIOS either calls the BIOS (which is held in ROM on the F series) or loads it from disk (on the PC/Xi). This reduced the size of the memory-resident BIOS dramatically, made the BIOS independent of the operating system (it meant that only a new RAM BIOS had to be written for each operating system), and achieved the goal of machine-independence — the RAM BIOS does not need to know what machine it is running on.

So today we have a good system of separate BIOS and operating system software units which combine to make a much better and more flexible software environment. The programmer can only gain, because anything that MS-DOS wants to do — and much, much more — can be done through a simple interface which is standard across all the Apricot machines.

Controlling the Control Device

There are two ways in which the Control Device can be called. The first is to use interrupt FC hex, which only Assembler programmers can do.

The second and easier option is to use a special 'vector' (a vector is an address of a routine somewhere else in memory) located in the Apricot's memory at 0060:0000 hex. That way, it's simple enough to call the Control Device from high-level languages such as Basic.

Of course, data has to be sent to the Control Device and received from it. There are three variables passed and one returned; for simplicity we'll call them DEV, COM, DAT and RET.

DEV is the device number, which must be set to one of the values in the table. COM is the command, and each device has different commands; all the possible commands are listed in the panel at the end. DAT is the data which a specific command will require, of which more shortly. And RET is the value that the Control Device returns.

Device numbers

```
31 hex  ...  Screen Driver
32 hex  ...  Keyboard Driver
33 hex  ...  MicroScreen Driver
34 hex  ...  Serial Driver
35 hex  ...  Parallel Driver
37 hex  ...  Clock Driver
38 hex  ...  Sound Generator Driver
39 hex  ...  Floppy Disk Driver
40 hex  ...  Hard Disk Driver
41 hex  ...  Modem [not supported]
42 hex  ...  Cache/Graphics/IBM mode Driver
```

Before we delve in deeper, let's use a practical example: setting the baud rate of the serial port to 1200.

We need first to indicate which device we're talking to by using the DEV parameter. As the table shows, the Serial Driver is device number 34 hex. The command list following indicates that the command to set baud rate is 4. The only other data required by the Control Device is the actual baud rate, and that is put into the DAT parameter as 8 (which corresponds to 1200 baud — see the command table).

After we've done this, the Control Device has to be called with the information; then it must be called again with a different command actually to perform the function.

The Basic code to do this is at the foot of the page. That program will change the transmit rate on the serial port to 1200 baud. If you're not using Basic, it can be done in exactly the same way in Assembler. Rather than using variables, the data is put into the 8086 registers like so:

```
BX  =  DEV      Device Number
CX  =  COM      Command Number
DX  =  DAT      Data information
SI  =  RET      Return data
```

And here's an Assembler program to do the same thing as the Basic:

```
kern -1.5
MOV  BX, 34H   ;  set BX to device number
MOV  CX, 4H    ;  set CX to the command
                  number
MOV  DX, 8H    ;  set DX to the actual baud
                  rate
INT  0FCH      ;  call the Control Device
MOV  CX, 3H    ;  this is the 'Update
                  Hardware' command
INT  0FCH      ;  call the Control Device
```

Control yourself

That's all you need to know, really. The meat of this Workshop is of course in the list we've compiled on the following pages of all the Control Device calls in the latest version of the BIOS; it's arranged in device order, then command order. Any programmer should be able to find a few calls which could be used to make a program better.

But experience has suggested a few pointers in using the Control Device.

- Although Apricot says that the Control Device will save all your registers, we've discovered that this isn't always the case — especially on the F series. So save your own registers before you call the Control Device, and restore them again when you exit.
- Where the Control Device wants an address, the DAT value must contain the segment address in paragraphs (this will normally be the value of either the DS or CS registers in your program); and the RET value must be the offset, relative to the start of the segment in DAT. You'll have to write some Assembler 'hooks' if you want to use these calls from Basic. The best way to do this is to define a string, and put into it the result of the VARPTR function for each value.
- If you write your software purely for the Apricot, portability to other machines like the IBM PC isn't important. So full use of the Control Device can be implemented. It's often quicker to call the Control Device directly than to use MS-DOS. As the serial handling is so good, it's easy to write comms applications using nothing but Control Device calls for all I/O.

Happy controlling! ◆

Basic example: setting baud rate to 1200

```
10 DEF SEG=&H60;                    REM this is where the Control Device vector is
20 DEFINT A-Z;                      REM All variables have to be INTegers
30 CNT=0;                           REM CNT is the CONTROL DEVICE
40 DEV=&H34;                        REM DEVice is Serial Driver
50 COM=&H4;                         REM COMmand is Set Baud Rate
60 DAT=&H8;                         REM ... and set it to 1200 baud
70 RET=0;                           REM There is no return data from this call
80 CALL CNT(DEV, COM, DAT, RET);    REM Call the BIOS
90 COM=3;                           REM Command to tell the hardware about new data
100 CALL CNT(DEV, COM, DAT, RET);   REM Call the BIOS again
110 END
```

The Control Device commands

Screen Device: device number 31 hex

machines	command	description	data
all	00H	initialise Driver	
PC/Xi	01H	set Text mode	000H
		set Graphics mode	001H
PC/Xi	02H	turn display on	000H
		turn display off	001H
F series	04H	print character	ASCII value of character
F series	0EH	output character on default screen	ASCII value of character
F series	0FH	update screen from image	

Keyboard Device: device number 32 hex

machines	command	description	data
all	00H	initialise Driver	
all	01H	disable Auto-Repeat	000H
		enable Auto-Repeat	001H
PC/Xi	02H	set Auto Repeat Lead-in delay	value in milliseconds
PC/Xi	03H	set Auto Repeat rate	value in milliseconds
all	04H	set Fall-Through mode OFF	000H
		set Fall-Through mode ON	001H
all	05H	flush keyboard queue	
all	09H	sound bell	
all	0AH	return keyboard buffer byte count	
F series	0BH	get byte from keyboard queue — wait if none	
F series	0CH	look ahead on queue, get next byte	

Serial Device: device number 34 hex

machines	command	description	data
all	00H	initialise Driver	
all	01H	transmit character	ASCII value of character
all	02H	receive character	
all	03H	update SIO hardware settings	
all	04H	set transmit baud rate [see note 2]	rate
all	05H	set receive baud rate [see note 2]	rate
all	06H	set transmit bits per character [5 to 8]	number of bits
all	07H	set receive bits per character [5 to 8]	number of bits
all	08H	set stop bits per character [1, 2 or 3]	number of bits
all	09H	set parity type [0 = none, 1 = odd, 2 = even]	number of bits
all	0AH	set transmit XON/XOFF	000H
		clear transmit XON/XOFF	001H
all	0BH	set receive XON/XOFF	000H
		clear receive XON/XOFF	001H
all	0CH	set RTS line [Request To Send]	000H
		clear RTS line	001H
all	0DH	set DTR line [Data Terminal Ready]	000H
		clear DTR line	001H
all	0EH	set transmit enable	001H
		disable transmit	000H
all	0FH	set receive enable	001H
		disable receive	000H
all	10H	get CTS [Clear To Send] status	
all	11H	get DCD [Data Carrier Detect] status	
all	12H	get DSR [Data Set Ready] status	
all	13H	set nulls after CR	number
all	14H	set nulls after FF	number
all	15H	set Auto LF after CR	001H
		clear Auto LF after CR	000H
all	16H	enable Serial Mouse	
all	17H	disable Serial Mouse	
all	18H	disable RTS/CTS protocol	000H
		enable RTS/CTS protocol	001H
		auto-enable RTS/CTS protocol	002H
all	19H	set DTR/DSR Protocol	000H

Note 1: There are more calls for the Serial Device than those described here. But in our experience, they are very machine-dependant — even BIOS-version-dependant — and we wouldn't recommend their use.

Note 2: For commands 4 to 7, the baud-rate parameters are:

rate:	50	75	110	134.5	150	300	600	1200	1800	2400	3600	4800	7200	9600	19200
value:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Parallel Device: device number 35 hex

machines	command	description	data
all	00H	initialise Driver	
all	01H	return available buffer space	
all	02H	set fault line status	001H
		clear fault line status	000H
all	03H	set Select line status	001H
		clear Select line status	000H
all	04H	set Paper Out status detect	001H
		disable paper out status detect	000H
all	05H	set auto LF after CR	001H
		clear auto LF after CR	000H
all	06H	set Serial Device	001H
		set Parallel Device	000H
all	07H	print chracter on output device	ASCII value of character
all	08H	flush printer buffer	
all	09H	transmit string	address
all	0AH	return output status	

Clock Device: device number 37 hex

machines	command	description	data
F series	00H	reset time to 00:00:00, date to 1-01-1980	
F series	01H	set date/time [see note 3]	address
F series	02h	get date/time [see note 3]	address

Note 3: The address here points to a parameter block set up like so:

```
WORD    days since 1-1-80
BYTE    minutes
BYTE    hours
BYTE    hundredths of seconds
BYTE    seconds
```

Sound Driver: device number 38 hex

machines	command	description	data
all	00H	initialise Driver	
all	01H	set key click volume [see note 4]	volume
all	02H	set bell volume [see note 4]	volume
all	07H	set bell frequency [see note 5]	frequency
all	08H	set bell duration [see note 6]	duration
all	09H	sound bell	

Note 4: Volume parameters are from 0 to 15, with 15 being off and 0 being maximum volume; values inbetween vary the volume.

Note 5: Frequency parameters are from 0 to 1023, from low to high. Values inbetween vary the frequency.

Note 6: Duration values are from 0 to 155. The given value makes a multiple of 20 milliseconds.

Floppy Disk: device number 39 hex

machines	command	description	data
all	00H	initialise Driver	
all	01H	set drive number [see note 7]	drive number
all	02H	set segment address of format track image	address

all	02H	set segment address of format track image	address
all	03H	set offset address of format track image	address
all	04H	format floppy disk	drive number
all	05H	return status of disk	drive number
all	06H	set drive status to swapped	drive number
all	07H	return disk type in drive [see note 8]	drive number
all	08H	return drive type	drive number
all	09H	check if disk swapped in drive	drive number
all	0AH	get BPB [MS-DOS only]	drive number
all	0BH	read/write/verify/write-verify [see note 9]	address
all	0CH	return status of drive	drive number

Note 7: Drive number is 0 for drive A, 1 for drive B, etc.

Note 8: The RET value for command 7 will be:

0 = 70-track single-sided
1 = 80-track single-sided
2 = 80-track double-sided

Note 9: To perform Read/Write, the address points to a block of data that looks like this:

WORD drive number [0 or 1]
WORD command: 0 = Read
1 = Write
2 = Verify
3 = Write with verify
WORD address of first logical sector
DWORD buffer address [not needed when you're verifying]

Hard Disk: device number 40 hex

machines	command	description	data
F series	00H	initialise Driver	
F series	01H	set disk status to swapped	drive number
F series	02H	return drive type	drive number
F series	03H	check if disk swapped	drive number
F series	04H	get BPB [MS-DOS only]	drive number
F series	05H	read/write/verify [see note 9 above]	address

About the newsletter

Aims

Our aim is to provide a complete editorial and information service to users of Apricot computers.

Except in the normal course of day-to-day contact, sub sales and the like, we are not affiliated with or supported by Apricot plc or any of its dealers. Nor for that matter are we sponsored by or directly associated with any user group or any other such organization, though in principle our goals and guiding intentions are very similar to the declared aims of many such bodies.

The Publisher does not necessarily agree with the views expressed by contributors, and assumes no responsibility for errors of reproduction or interpretation in the subject matter of this newsletter or from any results arising therefrom.

APRICOT FILE is a publication of the TP Group — which until we get the name changed is the trading style of Permapoint Limited, registered in England no 1929195.

The People

Editor	**Dennis Jarrett**
Production	**Fiona Carey**
Writer	**Garet Keogh**
Subscriptions	**Terrence Duggan**
DFR/Technical	**Roger Gann**

Postal address

TP GROUP
PO Box 509
London N1 1YL, England

Personal calls and deliveries

TP GROUP
Unit 4, 47-51 Wharfdale Road
London N1 9SE, England
PLEASE TELEPHONE FIRST

Phone, E-mail, Telex

Telephone **01-833 3501**
Telecom Gold **81:JET767**
[from overseas **10081:JET767**]
Easylink **19011170**
The Source **BDE566**

Telex

265871 MONREF G quoting reference 81:JET767

■■■ ISSN 0268-6066 ■■■

Classified

Free Small Ads

The full list of Classified Ads is featured in the DEALS FOR READERS catalogue published every other month (ads remain in the full list for around three months, or until they start yellowing at the edges, or until you ask us to delete them). This page summarises ads new this month, plus useful-sounding services (including our own!) and most non-sale entries.

TP Group Services

■■■ DISK RECOVERY: problems with deleted files, scrambled hard disks, and the like? We can (probably) help — no fix, no fee! Call Roger Gann at TP Group on 01-833 3501.

■■■ FILE TRANSFER: we can convert files between Apricot single- and double-sided, IBM PC format, PC/AT format, and Sirius — cheap and very quick! Call Roger Gann at TP Group on 01-833 3501.

■■■ TURNKEY CONSULTANCY: we can recommend and/or implement systems using popular packages or custom-written software. We can also convert you from Sirius or Apricot to another species. Sensible and knowledgeable advice, no particular dealership ties or other axes to grind, modest prices, guaranteed work. Call Roger Gann at TP Group on 01-833 3501.

■■■ DESKTOP PUBLISHING: we can produce your newsletters, price lists, press releases etc in near-typeset-quality on our laser printer using our DTP software NEWSWRITER from your disks in virtually any format produced by word processor and/or computer. Fast turnround, guaranteed quality, low prices. You're reading an example of our work right now ... Call Roger Gann at TP Group on 01-833 3501.

Apricot Sales

■■■ APRICOT PC with twin double-density drives, 12in monitor, 768KB memory, Intel 8087 maths co-processor, Microsoft Mouse; with Mannesmann Tally MT180 printer (three spare ribbons) and sheet feeder, voltage regulator, head cleaner kit, dustcovers — plus all the standard software (SuperCalc, SuperWriter, SuperPlanner, CP/M-86, Concurrent CP/M) and DR-Draw, GSX, WordStar 3.3, SWAP (Mercator), dBASE II, PolyPrint, Open Access, various APRICOT FILE disks, a host of other useful utilities; plus manuals for several languages. All with original documentation and packaging, bought from UK in June 1985 and used sparingly. Offers to G S Jayaram, PO Box 2642, Doha, Qatar (phone 417282 6-9pm UK time) — free air-freight despatch to any UK destination.

■■■ APRICOT PC with twin double-density drives, 9in or 12in monitor, and 512KB memory, the usual 'Super' software, COMM, CARDBOX, and utilities. Call Peter any evening on 0483 277169.

Apricot Add-on Sales

■■■ APRICOT 9in MONITOR for PC or Xi. Cream. Little used, £40 plus VAT. Telephone 0705 730566.

■■■ SUPERWRITER BUGS CURED — twelve files identifying and curing errors in SuperWriter version 1.03 program and manual, all for a copying fee of only £5.75 (includes VAT). Call Tudor Wynn-Williams on 097 081 375.

■■■ MICRO FOCUS LEVEL II COBOL (high-performance version) and FORMS-2 utility for sale. Develop applications using the world's most tried and tested language! Call Steve Budd, 0903 65287 (work) or 0903 68400 (home).

■■■ APRICOT PC 9in MONITOR. Brand new and unused (I bought a 12in screen instead): £100. Also APRICOT PC MOUSE, virtually unused: £75. Phone Robin Leaver on 09662 4104.

■■■ APRICOT PC SOFTWARE — OPEN ACCESS integrated system (database, spreadsheet, 3-D graphics, word processing, comms, appointments diary); virtually unused, a real bargain at £250. Sapphire's REPORT MANAGER (3-D spreadsheet, over 60 functions); £100. THE ULTIMATE DIARY (up to 5,000 entries): £15. MICROFILE (superbly simple electronic card index for up to 32,000 records): £20. NUTCRACKER SUITE (simplified text editing with typewriter-like functions): £15. Contact A Leigh on 01-542 4872 evenings.

Other Sales

■■■ DATA GENERAL 16-bit minicomputer with removable hard disks and multiple comms port. Original cost £25,000-plus; project failure now means all offers considered (or exchange for Apricot). All documentation, plus Assembler, Basic, Fortran. Phone 01-948 1603 any time.

■■■ COMMODORE MPP1361 printer; 136 columns, 150cps. As new (hardly used). Offers around £350. Phone Robin Leaver on 09662 4104.

Wants

■■■ DELTA 4 PROFESSIONAL wanted for Apricot Xi with 512KB memory and BIOS 3.1. Contact 037 971 246.

■■■ MMS INSTRUCTION MANUALS AND INTERFACE BOARD required for MMS hard disk unit and tape streamer. Willing to pay for help (even telephone assistance) in setting this up. Call Roy Roberts, 0923 775381 (office) or 0494 775088 (evenings).

■■■ GOOD DEBUGGER wanted for PC/Xi. I hate using Apricot's; can anyone help before I'm forced to write my own? (Preferably with symbols and/or C source display). Phone 01-948 1603 any time.

■■■ APRICOT PC with twin double-sided drives, manuals and monitor (would consider an Xi). Prompt payment. Phone 0525 376339 any time.

■■■ 10/20MB EXTERNAL HARD DISK required for Apricot PC with BIOS 2.4. G S Jayaram, PO Box 2642, Doha, Qatar (phone 417282 6-9pm UK time).

■■■ APRICOT 9in MONITOR (cream) wanted for PC. CARRYING CASE also needed. Phone 01-908 4631.

Help

■■■ PORTABLE user would like to contact another in South London to exchange information. Contact Geoff Reade, 9 Taymount Grange, London SE23 3UH.

Misc

■■■ KITCHENS: I work for a kitchen manufacturer and am able to buy kitchens heavily discounted because the packaging has been damaged — contents are perfect. Call 0484 545807 after 6pm.

Holidays

■■■ HOUSE TO LET in the South of France at Agde, on the Mediterranean coast. Attractively converted 16th Century house with roof terrace: sleeps 4-5. Local British management. Phone 01-352 5925 evenings. ◆

Horror stories wanted

If your software supplier goes bust or withdraws support of a package in favour of a new or updated product, you're generally up the creek — a situation with which Apricot users will be all too familiar. Well, the NCC-run **National Computer Users Forum** is looking into the problem — especially in terms of 'escrow agreements', where an independent third party holds a copy of the source code (which can then be released to the user should support be terminated, thus leaving the punter free to seek alternative support). The NCUF is looking for any horror stories; contact secretary-general **Patrick Raymont** at **NCUF, c/o NCC, Oxford Road, Manchester, M1 7ED.**

Money from DTP

PageMaker purveyor **Aldus** (as patronised by Apricot plc) went public in the States in mid June with stunning success — shares opened at $20 and finished the day trading for $37. The share offer brought in an estimated **$74 million** for Aldus, which plans to use the money for working capital.

CSD in receivership

Creative Sparks Distribution, the company set up by **Henry Kitchen** when his ex-employers at **Thorn-EMI** decided they wanted out from the software distribution business last year, has gone into receivership. Debts are reckoned to top **£1.5 million**. CSD owned some games developers, was one of the bigger distributors of low-cost software (to Boots, Lasky and Tandy, among others), dealt in hardware (Atari, Tandy, Miracle), and had UK rights to a number of business-level packages — notably the Samna line. Likely suitors include distributor **Software Limited.**

Japanese printers here

Oki and **Citizen** are the latest Japanese printer manufacturers to anticipate EEC trade illiberalisation moves by setting up European plants. Oki has taken over a former Burroughs factory in **Cumbernauld**, plans to start production in December, and will eventually employ 340 people there. Citizen is building a new £6 million facility in **Scunthorpe**, will employ 300, and should making 30,000 printers a month by Autumn 1988.

Printers rated

The last couple of issues of the generally excellent *What to Buy for Business* have focussed on DTP and printers. The **desktop publishing** survey was highly commendable, given the complexity of the subject. 'Best Buys' were **Ventura Publisher** for the IBM, **Ready Set Go** for the Mac. Packages rated 'Good Value' were **PageMaker** (on the IBM, not the Mac version — but then it's not fully available in IBM-format yet); Scientex's **Protex** for the PC (from the author of VuWriter, though not from Vuman); and **Quark Express**, an American Mac package. The one we use, **NewsWriter**, wasn't much liked — "not the best package of its type and could really do with a price cut".

The **laser printer** reviews were a bit iffy, giving too little prominence to software-compatibility (like the need for a PostScript-based machine for many DTP packages). Best Buys under £2,000 were the new **Epson** GQ-3500 and **Oki** LaserLine 6 (and the Oki scores well in our view because it uses dead cheap toner cartridges: *What to Buy* didn't mention the exorbitant cost of some manufacturers' consumables).

The following issue looked at **matrix printers under £1,000. Epson, Mannesmann Tally** and **Citizen** got no Best Buy labels, though they were all represented well in 'worth a look'. Instead the laurels went largely to **Micro Peripherals** — four models were Best Buys, the fifth was 'worth watching' (that's the very fast **MP-480**, rated at 480cps in draft mode and 74cps in NLQ, for only **£369**).

Also rated Best Buy were the **Seikosha MP-1300** and **MP-5300** (fast 80-and 132-column printers respectively); the **Canon A-55** (a solid wide-carriage printer — 200cps for £499, and it was the cheapest 18-pin printer reviewed); and in the 18/24-pin category for decent NLQ, it was **Canon** again with the A-60 and A-65 plus the **Star** ND24-10.

PCs in Europe

According to **Intelligent Electronics**, the European PC market grew 42 per cent in unit sales during 1986. The dominance of **IBM** slipped slightly, from 33 per cent the year before to just under 27 per cent in 1986; of the chasing bunch, it was a name from the past that stood out — **Victor Technologies**, described as "the only manufacturer which appears to have a dedicated strategy for the small business markets".

WordPerfect again

Another boost for the best word processor you can get for the Apricot — the latest quarterly market research from **Context** places **WordPerfect** second only to WordStar 1512 in UK sales volume and makes it the market leader by value. For the first quarter of the year, WordPerfect had 15 per cent of WP sales here by value; MultiMate was second at just under 14 per cent, Word and DisplayWrite were above nine per cent, WordStar 2000 had 7.5 per cent, and WordStar Professional was still plugging on grimly with 5.6 per cent.

Apricot vs DDL

A mini-fuss has blown up between **Apricot** and **DDL**, one of its 'master distributors'. DDL has been discounting Apricot products, but the company says that it was just passing on Apricot's own price reductions — though one dealer reckoned that "DDL must be dumping stock because it was certainly selling kit for less than it was bought" (if you can follow that). And DDL has been talking to other manufacturers; but Apricot *won't* be dropped, according to DDL director **Stephen Sowerby.** ◆

Modem opinions?

We want to include a **modem** in our DEALS FOR READERS lists, and we can get a discount on most of those available; so we'd like some feedback about which modem(s) to offer. Our current inclination is to go for the **Miracle Technology WS4000**, largely because it has a good spec and a fair price; but for instance the recently-announced **Linnet** by **Pace Technology** also sounds an attractive contender — V21/V23, Hayes-compatible, RRP about £140, and Dennis once had a girlfriend named Linnet. Which modem(s) can you recommend (or otherwise)? Any and all input is welcome ...