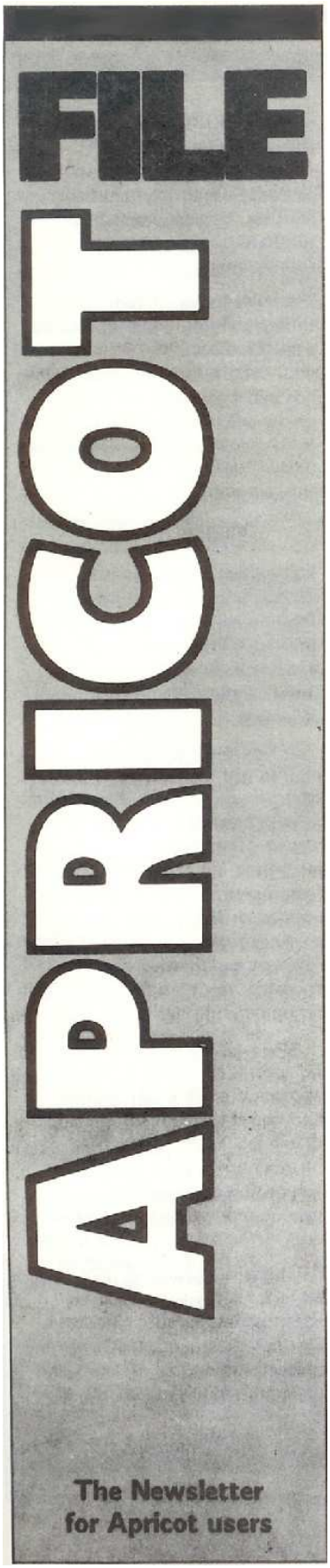
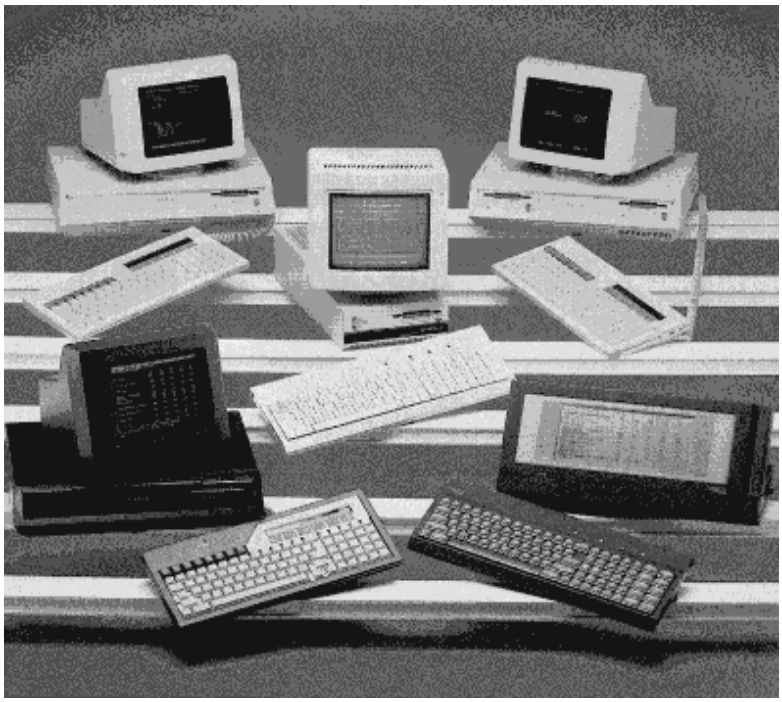


Apricot File 3.7



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 3.7

FILE APRICOT

The Newsletter for Apricot users

April 1988

Market moves

Who needs PS/2?

by Simon Potter

The Spring always sees a clutch of market analyses for the previous year. The current crop makes for some interesting reading, particularly in view of IBM's attempt to swing at least the upper-echelon buyers into the new standards imposed by PS/2, OS/2, VGA graphics — and especially the MCA bus.

So who needs PS/2? And in particular, who needs the much-vaunted Micro Channel Architecture with which IBM obsoleted all those third-party add-ins? Not the punters, if you believe US buyers: a US survey from **Comtec** estimates that fully 63 per cent of 1988 microcomputer purchases will be for systems valued at less than $3,000 — which means non-MCA machines. And the on-line industry newsletter *Newsbytes* reports widespread discounting on the PS/2 (many retailers will knock at least 40 per cent off the RRP, some will go as high as 48 per cent) ... while the best deals on old-style ATs, which are no longer being made by IBM, only approach a 28 per cent discount.

What's more, IBM itself says that Model 30 — which doesn't have MCA — account for half of PS/2 sales.

The recently-published *Annual Market Review* from **Romtec** (price **£695**) gives UK sales figures for last year that tend to support the view that PC/AT-type technology continues to hold sway over MCA. Romtec gives IBM 24.7 per cent of the market, but within that figure IBM's MCA-based machines represent less than 5 per cent.

True, 1987 was early days for PS/2; and IBM has recently been talking eagerly about licensing PS/2 technology to clone-makers. Indeed, a rash of PS/2 clones is just about to hit the streets. But MCA is the most significant feature of PS/2 — the range's only serious USP apart from the IBM name, in fact — and all the indications are that the compatible-makers *won't* be espousing MCA exclusively, even in their so-called PS/2 'clones'.

The buying trends effectively posit two kinds of market: corporate purchasers (who want to run networks and multi-user systems) and the rest (a sub-£500 8088-based PC-type micro for the home user is basically a downgraded version of the same kind of system for which a small business user pays up to £1,500 or so). The corporates want big memories, large disks, lots of comms ports, and fast processors to handle lots of users and lots of usage: so they'll be interested in the PS/2-type 80386 systems, particularly because they have the beef to handle the OS/2 and Xenix operating systems. The rest of us will find the technology of 8086 or 80286s adequate, MS-DOS and the myriad of DOS-compatible packages more than satisfactory, and the pricing of such systems increasingly attractive.

But neither group actually *needs* MCA, and it seems that neither is actually *buying* MCA either. ◆

In this issue

Review: Ansible's APE Apricot emulator 2
Ansible's Update: Langford responds 5
dBASE Primer — Part Two: Selecting data 7
The Q&A Clinic 10
Workshop: A spreadsheet accumulator 12
Readers' Writes: Letters to the Editor 16
The Turbo Pascal Primer — Part Five 17
Notebook: Maintenance for Apricot 19
The Workshop: Customise your keyboard — Part Three . . . 21
The Workshop: GSX revisited 22
Free Small Ads 26

Review

Monkeying around with an IBM

Ansible Information's APE: the Apricot emulator reviewed

By Eddie Bromhead

When I was asked if I would like to review Ansible Information's APE Apricot emulator, I jumped at the chance. In my Department at Kingston Poly we have a number of Apricot-type Apricots along with a medley of IBM clones, including one real IBM PC and quite a few of the latest Apricot Xen-i PCS/286 machines (which are in the process of being linked into a VX network — hopefully completed by the time you read this). Personally I always use WordStar, since I paid out good money for it when the Apricot's free software included only SuperCalc; but many of my colleagues are quite attached to SuperWriter, and APE seems to be a good way of transferring the software and the expertise in using those programs on to the new machines.

The other reason to anticipate the review with relish is my admiration for the dogged way Messers Langford and Priest (*aka* Ansible Information) have stuck with the Apricot, and the way their software provides useful facilities for those who are Super-afflicted. Not only that, but I like the occasional column in APRICOT FILE ... (get well soon, David!).

What and why

One of the laughable claims for the original Apricot was that it was compatible with the IBM PC, the (shaky) grounds for this claim being the presence of a so-called 'IBM emulator' and the support for IBM-style ANSI escape codes that was built into the Apricot BIOS.

I had already started on the review when Release 3.6 arrived with Terence Green's piece dealing with the issues of moving from Apricot to IBM architectures, and so I won't repeat that detail. Instead, readers are referred to a number of previous APRICOT FILE articles which list the specific differences between the two machines.

In particular, Release 2.9 had Nick Wilson reviewing some of the various editions of the IBM emulator. Until Ansible came up with APE, however, nobody had thought of the reverse process — emulating the Apricot on an IBM. It's hardly a logical step for IBM. But it must be worth considering for the poor old Apricot user, seeing that the Apricot architecture is on the way out — except for Xens, which are still available.

The idea is that you can run Apricot software (especially the free Super series) on an IBM-compatible; and therefore you won't need to invest heavily in IBM-format software when you buy your new computer. Without something like APE, you'll find the same sort of annoying incompatibilities that we've grown used to — the Apricot software you know and love will simply crash an IBM.

(By the by, the other silly claim in the original promotion — for which Apricot's knuckles were rapped by the Advertising Standards Authority — was that the cost was under £1,500 ... which it was, if you didn't buy a monitor. Know anybody who uses an Apricot on the MicroScreen alone?)

What APE does

When I was supplied the review copy on a 3½in disk, I chortled with glee. Curses, the disk wasn't in Apricot format but in IBM/Microsoft format, so I couldn't have any fun at Ansible's expense. On the other hand, most machines which handle that format aren't cheap clones — well, not yet. I tried APE on an Apricot PCS/286-30, an Amstrad PC-1512, an IBM PC, and an Olivetti M24. *(Ansible can supply 3½ or 5¼in IBM disk formats, in fact — Ed.)*

APE is a terminate-and-stay-resident program for IBM compatible computers which intercepts calls to the Apricot's Control Device (Release 2.10) and Escape Codes (see Langford's Complete Reference in Releases 2.9 to 2.11). It then replaces the call or escape sequence with the appropriate IBM BIOS call (which it generates in the time-honoured and Peter-Norton-approved fashion of neatly side-stepping PC-DOS and going straight to the guts of the beast with a BIOS interrupt. Doing this makes it appear that an IBM machine is faster than it really is, incidentally.)

The result is that programs which use the Apricot operations can now run on an IBM clone. APE also allows itself to be switched on and off and to be toggled into a mode in which it reports Control Device calls and Escape sequences which it doesn't recognise. More on that in a minute.

Also on the disk are programs to replace KEYS and KEYEDIT — in other words, utilities that allow an IBM user to configure a keyboard just as we Apricot types have always been able to do. Different fonts are beyond the IBM, unless you want to go into graphics mode — and APE doesn't.

Because APE is in a period of rapid development, the penultimate version is also supplied on the disk. This makes for a little uneasiness at first, but on reflection it is a very honest admission on the part of Ansible that software development is usually done at the user's cost. I don't mind this, although previous reviewers have been scathing about software which is this honest (like the review of SIGNWRITER, Release 2.8).

APE run-time options

[no command line]	load APE in the standard way
L KEYBOARDNAME	load a named keyboard
S	save the current keyboard
X	exit APE
D	set the debug switch to report funnies

The big test

Realising that Ansible had aimed this product fairly and squarely at the Sorcim/Computer Associates Super software that came free with the Apricot — SuperCalc, SuperWriter, SuperPlanner — I tried SuperWriter versions 1.03 and D1.09. They worked fine, and SuperWriter is even quite good with Ansible's special keyboard.

SuperCalc 1 worked perfectly as well, and SuperPlanner probably does too.

On the other hand, do you want to run the Apricot Super series? The IBM version of SuperWriter (1.02) is miles better than either of the Apricot versions; and SuperCalc is now up to version 4, which is generally a much improved product relative to its predecessors and can use a co-processor for extra recalculation speed. Paying for brand-new IBM-format copies of either of these packages would give you readable documentation too.

Two of my other favourite workaday programs, the SDIR and FDIR sorted directory public-domain utilities from the DEALS FOR READERS service, also worked fine. SDIR even managed colour when run on an EGA display — Lord knows how, or why!

So, no problems about straight transfer for the software that most Apricot users will be running. And if that's as far as your needs go, then APE is the package for you.

What APE didn't handle

After that, I decided to try a number of other programs, including some I had written myself. Sadly, failure was pretty much the rule of the day. Lest anyone (particularly those chaps at Ansible) think that I'm knocking APE, let me point out that at least Apricot SuperWriter runs under APE, whereas IBM SuperWriter won't run under *any* version of Apricot's IBM emulator ...

First of all, trusty WordStar 3.30. This loads up on my Apricot via a little program named RWS.COM. RWS is one of a family of programs generated by a 'Black Hole' utility called RINSTALL; this labels the MicroScreen, installs a keyboard, then loads the program in question — which in this case is WordStar. Now, RWS.COM works faultlessly with all BIOS variants from 1.4 upwards: and RINSTALL has been widely used, by the Ansible team among others. Nevertheless, RWS failed to run under APE. Setting the debugging toggle in APE revealed that RWS used a Control Device call (032H code 06) not listed in the APRICOT FILE article and hence of unknown purpose; not only that, it also employed the escape sequence ESC { — again, allegedly a 'do nothing' code.

In the BIOS 3.1 manual the meaning of the Control Device function is made pretty clear: various keyboard statuses are cleared down (like CAPS LOCK). This is the rational call to follow function 05 and empty the keyboard buffer. The process is a sensible precaution to take when a program has file deletion as an operation, but it can be a little frustrating for the would-be type-ahead typist. My guess is that many of the BIOS 3 Control Device functions were implemented in earlier BIOSes too.

In the early days of BIOS 1.x, I used to regard RWS as a saviour. For instance, running a program from within WordStar would usually make it possible to recover from a messed-up screen or keyboard — you simply exited from RWS, and all was sweetness and light.

Well, I wasn't really expecting WordStar to work in its RWS guise. And you can configure even the Apricot version of WordStar to accept the IBM's terminal characteristics, and hence to run fine on that machine. The Ansible KEYEDIT and KEYS programs are useful, however, in setting up the keyboard to match the useful keyboard Apricot thoughtfully provided with RWS — for instance, SHIFT plus the cursor arrows moves by a word or page in comparison to the letter or line moved by the unshifted cursor keypresses. A batch file is *de rigeur* if you're to perform the individual functions of RWS one at a time (as indeed Sirius users used to ...).

So Lesson One appears to be that APE works ... mostly. Lesson Two is that APE doesn't like you to do *real* Apricot-type things like switching or loading fonts or keyboards, especially from within a program.

The next thing I tried was a program of mine which causes numeric overflow and underflow on a numeric co-processor. The IBM handles this differently to the Apricot, and it's only relatively recently that Microsoft have got to the bottom of it all. No, APE didn't make any difference: the Apricot version hung up the IBM, just as effectively as the IBM version hung up on an Apricot, even with the IBM emulator installed.

For obvious reasons, Apricot screen dumpers which depend on the precise hardware configuration of the Apricot also don't work — but then, are they needed? IBM machines come with a screen dumper for Epsons built into DOS, and the same sort of thing can be had for other printers too.

So here's Lesson Three: when your program addresses a fundamental hardware difference, you're better off maintaining separate versions. And Lesson Four adds that whatever you thought would be a useful utility on your Apricot almost certainly exists already for the IBM.

Finally, you have to remember that most Apricots run under MS-DOS 2, whereas these days the clone you buy will inevitably have MS-DOS 3. Some programs check for the MS-DOS version and expect to find MS-DOS 2 (rather than MS-DOS 1 — they rather crudely assume that DOS version equals '2' means the same as not equal to '1'). Obviously anything that checks MS-DOS version and expects '2' will fail at the MS-DOS 3 hurdle!

An interesting variant of this was a DEALS FOR READERS program named SYSTAT which reported A and B as existing on the same floppy drive (a well-known feature not implemented on hard-disk

Apricots). It then proceeded to crash when reporting the size of disk C — its author probably never reckoned on 30MB Winchesters ...

Lesson Five tells you that what you fondly imagine is an Apricot utility may well have been developed first on a Sirius under MS-DOS 1.25 and has subsequently been hacked for MS-DOS 2 and/or the Apricot. This applies particularly to short(ish) utilities. They may well fail — but they can, of course, be further hacked using techniques familiar to readers of APRICOT FILE. The point is that APE doesn't do it for you.

(In fairness to Ansible, it's worth pointing out that APE *is in a constant state of flux — as customers request additional facilities, Ansible tries to add them. Some of the deficiencies in Eddie's (early) review copy of the package may well have been rectified in subsequent releases by the time you read this — Ed.)*

When you don't even need APE

Front-ends are ten-a-penny in the IBM world, so I didn't bother with ACTIVITY or MANAGER — anyhow, the GSX-based ACTIVITY uses GSX extensions not in the standard system. Nor was it worth testing GWBASIC (you'll get a Basic with whatever IBM clone you buy) or ASYNC (better comms programs of so little sophistication are widely available in IBM format).

But some Apricot software will run unmodified on the IBM. The list is relatively small; but it does include MSBASIC (also GSXBAS.COM — the later version — and GSX itself, *if* you have an IBM PC screen driver), the BASCOM Basic compiler, and many other language compilers. In short, you'll be able to run 'Apricot' packages that are basically generic MS-DOS applications and/or which were a straight copy from the IBM or Sirius in the first place.

I thought that anything which used GSX might be immediately transferable. But frankly, IBM GSX just isn't good enough — the resolution is appalling, and the primitives too few to make transfer easy. On the other hand, it probably does work. A program of mine which uses VT52 escapes worked fine under APE's emulator and Apricot keyboard; but since IBM GSX lacks the circle primitive, the graphics in it wouldn't work.

Last words

The documentation that comes with APE is typical Ansible stuff — concise, clear and informative. What it lacks is a crib sheet listing what works, and what doesn't. That could make all the difference to a prospective purchaser. (After writing this, I deleted my own list — just suppose I had got it wrong.)

In summary — a nice try, Ansible. APE meets all its design objectives as far as I can see. It's perfect for those who have to pay for all their own software (like me) and who are happy with the Super series; but it probably won't be the choice of the corporate user, who will buy all new software anyway.

But just as a PC-286 upgrade isn't clear commercial sense, neither is APE. They both just let you feel cumfy and familiar when embarking on a complete change of direction. If I had to move to an IBM clone permanently, APE might ease the pain — but not the sense of loss.

The strongest argument for buying an IBM clone is that it opens up a whole new range of software that just isn't available for use on an Apricot. That software is often incredibly cheap, or extremely powerful — and it runs. ◆

What's on the disk

File	Size	Purpose
AP.BAT	246	Batch file for installing APE
APE.COM	5632	Latest version
APE0.COM	5120	Previous version
ESC.COM	512	Utility for sending escape sequences to the screen
KEYEDIT.COM	20110	Replacement for the Apricot utility
MK.COM	2560	MiniKey (equivalent to Apricot KEYS facility)
APRICOT.KB	1024	Sample keyboard
PLAIN.KB	1024	Sample keyboard
SW.KB	1024	Sample keyboard
TEST.KB	1024	Sample keyboard
READ.ME	7238	Release notes

APE Apricot Emulator

Description:	Apricot emulator for IBM-compatible computers. Applies particularly to running Apricot-format SuperCalc and SuperWriter on IBM-type machines.
Supplier:	Ansible Information
Address:	94 London Road Reading Berks RG1 5AU
Telephone:	0672 62576
Price:	£57.50, including VAT and despatch. (Available at this price from DEALS FOR READERS)
Configuration:	Any IBM-compatible machine — supplied on 3½in or 5¼in floppy disk
Conclusion:	In some respects, APE is a super little program. In others, it doesn't quite live up to its promise. But for somebody whose trusty Apricot has to be replaced by an IBM clone, and whose needs were entirely served by the free software, this package really is a good buy.

Ansible Responds

On blowing one's own trumpet

By David Langford

Many thanks for the advance peep at Eddie Bromhead's review of APE. It's a fair and pretty encouraging report; and I thought that rather than making the traditional outraged, whingeing response, I'd send a batch of addenda from the viewpoint of a poor bloody programmer.

Updates

Chris Priest keeps berating me for reluctance to send out review copies of software: I find that the very instant we despatch one, an idea for a major improvement occurs to me! So sending a review copy always means more work, a terrible prospect. Thus the disk sent to APRICOT FILE for review is already a trifle out of date, as the mighty editorial staff deduced. These are the main new features:

- Better keyboard provision for extra keys on AT-type keyboards.
- Multiple command line options allowed (this was long overdue).
- New command line option B (for Bold) to toggle the 'bright' and 'dim' screen appearances as set by Apricot escape sequences. This echo of our old ANSIBLEPATCH fix was inspired by the Amstrad PPC, where 'bright' means 'sort of thin and spidery' while 'dim' is quite a bit more legible.
- Two new keyboard files are now supplied. WP.KB is for WordPerfect — yes, despite the glum predictions of our early manuals Apricot WordPerfect *can* be run on an IBM (and although the complex keyboard table was a real pain to set up, we were sustained by the happy thought of annoying Sentinel Software).
- And SPLAN.KB, believe it or not, configures SuperPlanner: for a long time we didn't bother with this dreadful program, but one day we found a customer who not only used it but — incredibly — loved it so much he wanted to take it to the IBM.
- Better error checking on keyboard files — APE and our IBM KEYEDIT use a different format from Apricot .KB files, which should be discarded. Ok, the awesome Ansible technical staff (D.Langford) ought to write a straight conversion utility, but keeps not doing so for reasons to be explained below.
- A much expanded and corrected manual, just off the presses.
- APEDISK.COM, a little utility of which we're immoderately proud: users of 3½in IBM drives can run APEDISK and stick in their standardformat Apricot disks to be read and even written to by the IBM — without impairment of the selected drive's ability to read and write IBM disks. Either single- or double-sided Apricot disks can be tackled.

This is most useful to Apricot owners who have single-sided drives (which even with BIOS 3.1 and the latest FORMAT version 1.5.3 don't seem able to write enough tracks to cope reliably with IBM's single-sided format) and to users with F series machines without the BIOS 3.1 chips needed to access double-sided IBM disks. Throw away that RS-232 cable and walk again!

A naughty aside: how could TP Group desert the 3½in 'industry standard of the future' by reverting to boring old 5¼in for their advertised OMEGA IBM clone? Think of the money you could have saved by not having to supply two copies of KERMIT and a cable link with each machine ... *(Point taken. In fact we now can supply* OMEGA *with 5¼in or 3½in fitted as standard, no price difference — Ed).*

- SETVER.COM, I must admit, was directly inspired by a passage in Eddie's review. It consists of a means of brainwashing MS-DOS into returning any version number you care to select, from 1.00 to (power users only, please) a macho 9.99. Does the vital Apricot utility GROTTY.COM insist on running under DOS 2.11 and no other, when your IBM has 3.20? Enter 'SETVER 2.11'and pass into a state of unimaginable bliss, until of course you have to turn it off again to run the IBM CHKDSK or DEBUG ... This will also save Roger from sitting all day patching his collection of 5,271,009 versions of DEBUG (SETVER runs on Apricots too).

Minor points from the review

- Yes, Ansible can of course supply in both 5¼in and 3½in disk formats.
- Eddie doesn't mention the (admittedly fairly trivial) command line options which can turn APE's emulation on and off without actually removing it from memory and making the occupied memory available again as does the X option — the options are '+' and '–' respectively.
- Control Device call DEV 32 COM 6 was omitted not because it wasn't in the APRICOT FILE article — our research does go a mite further — but because its main purpose is to reset the Apricot scan code buffer (as distinct from the keyboard queue) and IBMs don't appear to have such a buffer. Since this call doesn't even turn off CAPS LOCK as Eddie implies, there doesn't seem to be anything to emulate!

- Apricot keyboard loader programs like WP.COM or RWS.COM should be dropped when trying to run things on an IBM: users should make up a suitable keyboard with APE's KEYEDIT instead. APE doesn't support alternate fonts, but is perfectly happy with keyboard-swapping ... not from within programs, though.

My major dissatisfaction is the impossibility of mapping the Apricot keyboard perfectly on to the IBM one: every attempt to do a mechanical conversion of Apricot keyboards to APE/IBM format produces annoying inconsistencies. Which key do we call CLEAR? If, as in APE, we map the MicroScreen on to the ALT positions of the IBM number keys, where does that leave the SHIFT and CONTROL positions of the MicroScreen? An orderly mapping of the PC/Xi function keys to IBM ones would include CALC (I admit it, I haven't got around to writing in an entire pop-up calculator routine) and PRINT (whose screen print function is redundant because supplied elsewhere on the IBM keyboard) but would omit the interpolated F5 VOICE key found on the F series. *Et cetera.*

Using the APE KEYEDIT to prepare new keyboards by hand is a bore, but remains the only reliable answer. We've put some more tips about this into the manual, based on the hideous effort of configuring WordPerfect. In essence, when tackling an unknown Apricot program which doesn't respond to the default keyboard, you sit for an hour firing ASCII codes at it (via the IBM ALT-plus-numberpad function) until you've recorded what everything does: then set up the keyboard file from this information.

□—A crib sheet of what works and what doesn't? We're still trying to decide on a satisfactory approach. It's no good saying 'no Control Device calls to device 34H are supported' (which is in fact so) because the average buyer tends not to know what the hell we're on about. More subtly misleading is the translated version, 'BIOS calls to the serial port aren't supported': MS-DOS calls to the COM or AUX devices will obviously work, so now we have to distinguish between these and BIOS calls — and by the time we've finished the info sheet looks terrifying, and people ring Chris Priest to wail 'it's all too complicated ...'.

Why, you ask, has lazy Langford so far failed to provide serial facilities? Partly for the excellent reason given by Eddie: comms utilities better than the average Apricot offering are freely available in the IBM public domain, so the transfer is fairly pointless. Partly, too, because the Apricot serial port controls don't map neatly on to the IBM ones: the joker in the pack is that Apricots support different baud rates for 'send' and 'receive', and IBMs (nominally) don't. Malcolm Messiter tells me, encouragingly, that it took him a mere 1,000 hours of programming and testing to provide such facilities in the IBM version of TRAN-SEND. While not averse to reinventing the occasional small wheel, I draw the line at Malcolm's 56-ton articulated lorry.

Meanwhile, for the record: what won't work and almost certainly never will work on the IBM is any program which bungs characters straight into the Apricot PC/Xi/Xen display memory at segment F000 hex. This memory area doesn't exist on any standard IBM, and output vanishes into a logical black hole. (Yes, it's sort of remotely theoretically possible to take a 'softclone' approach which rewrites your program to change the assumed screen segments and attributes in the actual code as it runs — a bit like rebuilding a car engine while it's doing 80mph down the M4 — but so far experiments in this area have been curtailed by the realisation that Life Is Too Short.)

Apricot software using this display method is relatively rare, but includes WordStar 2000, CardBox Plus and Perfect Filer.

Apricot itself always discouraged the technique: and sure enough, such programs usually hang up without output on the entire F series (different screen address; sound of hysterical giggles and "I told you so" from entire Apricot technical support department). I am not without sin: our ACOLYTE is another offender, though we did include a 'machine type' check and write a separate F series version.

What doesn't *yet* work is a more complicated and embarrassing matter, heavily conditioned by what BIOS calls are actually used by programs we've actually met. APE is rather short on the little-used final-release BIOS calls to interrupt EC (so Psion XCHANGE is still not supported) and on fine control of colour (none of our Apricots have a colour monitor!); and it omits one ESC sequence of some importance"the ESC 4 'configure key' command (see my paragraph on keyboard mapping for why).

Other things too are deliberately ignored: all MicroScreen output, whether commanded by ESC sequences or Control Device calls to device 33h, is discarded. Again, this is a mapping problem: one could theoretically put it on the status line, but the main program often wants to use the status line too, and MicroScreen key labels often rely on there being two lines one above the other, and so on ... APE supports all output through the display interrupts 29 and F1 hex, by the way, as well as normal MS-DOS screen handling.

Endpiece

I wouldn't dispute the review's conclusions. Swapping software is the sensible thing to do when you change to an IBM. But so many people are filled with instinctive horror at the thought, or have 50 disks of SuperWriter files, or want to maintain compatibility with the old Xi or a colleague's machine, or blench at the 'nominal' upgrade-to-IBM-version cost, that APE seemed worth writing. ◆

A dBASE Primer

Part Two: Selecting data

By Terence Green

Once there are a few records in a database, we can begin to move amongst them. One method is to wade around the whole database using the *record pointer*.

Whenever a database is open in dBASE, a record pointer holds the number of the current record. The pointer can be moved directly to a record by number or it can be repositioned with commands like GO and SKIP.

GO is useful for moving around the records in a database. It moves the pointer to the TOP or BOTTOM or to a specific record number:

```
. GO TOP
. GO BOTTOM
. GO 4
```

The effect is to move the record pointer to the relevant record. After a GO, check where the pointer is with a DISPLAY command; that will give the record number and print the contents. (Note that record numbers are *not* unique — some commands, like PACK and SORT, change the record number.)

SKIP also moves the record pointer, either by a single record (when used with no parameter) or by a specified number of records. So SKIP on its own moves the pointer to the next record; SKIP 5 moves it five records forward; and SKIP -8 moves eight records back.

Precision: operators and expressions

A more precise method of moving through the database looks at selected records only by limiting the 'scope' of the command — in practice, by adding a range parameter to the command format.

The scope may be ALL, NEXT *n*, RECORD *n*, FOR *expression*, or WHILE *expression*. An 'expression' in this case is almost anything which dBASE can evaluate — a brief definition which skirts a lot of pages in the dBASE *Reference Guide* — and in the context of this article, an expression defines the scope of a command.

The default 'scope' is ALL, meaning that commands such as LIST on its own (equivalent to LIST ALL) will operate on all records unless further limited in scope.

```
. LIST FOR surname = 'Smith'
```

The equals sign is one of a number of comparison and logical operators; in this example, the expression *surname = 'Smith'* selects any record whose surname field contains 'Smith' as the first five characters. The command LIST operates accordingly, and records that don't match are not listed.

Longer surnames starting with 'Smith' such as 'Smithson' also match the expression *surname = 'Smith'*, of course. Similarly, we could cast the net wider by matching all records where the surname begins with 'S' in our database:

```
. LIST FOR surname = 'S'
```

All surnames beginning with 'S' would be *true* for this expression. The match is case sensitive:

```
. LIST FOR surname = 'Smith'
```

That would find 'Smith', but not upper/lowercase variations like 'SMITH' or 'smith'. To pick up all of these, and avoid the hassle of case sensitivity, try this:

```
. LIST FOR !(surname) = 'SMITH'
```

This one displays the use of a dBASE *function*. Functions are special operations that add to dBASE data-handling; the exclamation mark is the uppercase function for dBASE II, which effectively converts the contents of field 'surname' to capitals before comparing it with the target text. dBASE III has the more explicit UPPER() function instead:

```
. LIST FOR UPPER(surname) = 'SMITH'
```

Multiple expressions can be used with a command, and as a result the selection can be very complex — within the limit on length of a command line at 254 characters. Comparison and logical operators and functions can be mixed to produce expressions:

```
. LIST FOR !(surname) = 'SMITH' .AND. TELEPHONE > '6'
```

Here the tests select records where the surname at least starts as 'Smith' or 'SMITH' and where the first character of the phone number field is greater than '6'.

Note that we're testing for characters rather than digits in the phone number check — character strings for matching are enclosed in quotes when used in expressions. If the telephone field were numeric (which would be pointless because you'd never use a telephone number for calculations) we would not need the quotes. But when numerals are used as text, the same rules of precedence apply — just as number 1 is not greater than the number 2, so the *character* '1' is not greater than '2'. As a result, the phone number '580 5689' would not be selected here because its first character is not greater than '6'.

Comparison operators

=	equals
<>	not equal to
>	greater than
<	less than
<=	less than or equal to
>=	greater than or equal to
$	equal to or contained in

Logical operators

.AND.	True if both conditions satisfied
.NOT.	True if condition not satisfied
.OR.	True if either condition satisfied

Incidentally, when quote marks themselves form part of the contents of a field, surround the character string with double quotes (as in "O'Halloran") or square brackets [O'Halloran].

LIST and DISPLAY have slight differences in operation. DISPLAY used alone shows the current record; LIST always starts at the beginning of a file and lists all records. Both DISPLAY and LIST can be qualified by NEXT *n*:

```
. LIST NEXT 45
```

This will put on to the screen the next fortyfive records starting with the current one. The display will continue unless it is interrupted with the STOP key; by contrast, DISPLAY puts up a screenful at a time and waits for a keypress before continuing.

The LOCATE command

There are many other commands with which expressions can be used. LOCATE, for one, can be used with any expression that works with LIST or DISPLAY; in this case it searches sequentially through the records for the field 'surname' which contains 'Brown' ...

```
. LOCATE FOR surname = 'Brown'
```

If there are a number of records with that name, LOCATE will stop at the first. So you might add a DISPLAY command to display the record and CONTINUE to find the next one:

```
. LOCATE FOR surname = 'Brown'
. DISPLAY
. CONTINUE
```

Simple output

Anything that appears on screen can be sent to printer or disk as well as the screen, and LIST and DISPLAY are both useful for that kind of simple dBASE output.

The output is unformatted because in most cases LIST and DISPLAY by themselves will start each record on a new line and wrap round after 80 characters regardless of content. Generally the output has to be shortened to less than 80 or 130 characters per line or massaged manually with a word processor; the exceptions are where a printer supports condensed print, or has a wide carriage, and is correctly set up to print the fullest line width.

One of the easiest ways to print is to SET PRINT ON (equivalent to MS-DOS's CONTROL-P) followed by a LIST command such as ...

```
. LIST surname,address1,address2,address3
```

Each record starts printing on a new line and wraps if longer than your printer's page width. Don't forget to SET PRINT OFF when finished.

Another method copies all program output to a named disk file. There are two parts to the process. First open the file by giving it a filename, then direct output to it.

```
. SET ALTERNATE TO filename
```

If you don't specify a filename extension, dBASE uses .TXT. At this point the file is opened (and any file on the default drive with the same name and extension will be been erased without warning!); but data will not be sent until it is specifically enabled ...

```
. SET ALTERNATE ON
```

Once again, don't forget to SET ALTERNATE OFF when finished.

As a rule, *formatted* output comes out more or less where you tell it to and *unformatted* doesn't. For straightforward formatted output, use the REPORT command. A REPORT can be duplicated to another file, and modified. REPORT really comes into its own for numeric output with subtotals and calculations. Fancier formatting will require a bit of programming, of which more later.

Applying order

At times it's handy to have the records in some order. SORT and INDEX achieve this — INDEX normally creates an ascending order, SORT gives descending too.

SORT creates an entirely new file, the same size as the source file, which is indexed on one field in dBASE II (more in dBASE III). It needs at least as much disk space again as the source file, and takes a long time on a large database. To sort on more than one field the entire sort process has to be repeated for each field in dBASE II. Once a database is SORTed, new records can be inserted manually in the correct sequence: but APPEND adds records at the end of the file, and will not preserve the sequence.

INDEX creates an index file with the extension .NDX, which contains only the index data and record number. For example, indexing on a six-character name field in a database of 300 characters per record means that the .NDX file will have to store only the six-character field with the

record number. So the index file is created faster and takes up less space than SORT.

Using INDEX is much quicker. But when records are added to an indexed file, each record is indexed *as it is appended* — and that can make screen response very slow when a complex index is being used.

Any dBASE expression is valid in an INDEX command. To organise a bibliography on author sequence within publisher's name, say, the commands would be ...

```
. USE bibliography
. INDEX ON publisher + author TO pubauth
```

dBASE appends the .NDX suffix to the filename, for instance producing an index file called PUBAUTH.NDX from this example. Once created, indexes are automatically updated if loaded with the database or invoked thereafter:

```
. USE bibliography INDEX pubauth
```

Any records appended while the index file is not active will obviously not be indexed; if you later load the index, it will not 'know' about the un-indexed records. This may cause to problems; if you use indexes and have had problems with lost records, re-indexing might help.

With indexed records, the command FIND can be used to quickly move to the required data as only the abbreviated index file is searched for a match. FIND does not require character strings to be surrounded by quotes. To move to the first surname beginning with 'G' the command is ...

```
. FIND G
```

That's it for now. Next time we look back for problems, quirks, and solutions: and forwards to more advanced printed output. ◆

Commands covered here

GO	SKIP
LIST	DISPLAY
CONTINUE	LOCATE
SORT	INDEX
REPORT	FIND

SET PRINT ON/OFF
SET ALTERNATE TO
SET ALTERNATE ON/OFF

About the newsletter

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 organisation, though our goals are 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.

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

Phone, E-mail, Telex

Telephone	**01-833 3501**
Easylink	**19011170**
The Source	**BDE566**

The People

Editor	**Dennis Jarrett**
Subscriptions	**Terrance Duggan**
DFR/Technical	**Roger Gann**

The Guarantee

The annual subscription to APRICOT FILE buys 12 monthly issues as follows:

Britain	£50.00
Rest of Europe	£60.00
Elsewhere (air mail)	£65.00

If having subscribed you come to feel that APRICOT FILE does not meet your requirements or that it does not match our claims, we will gladly cancel your commitment to the magazine and refund the unused portion of your subscription.

■■■ ISSN 0268-6066 ■■■

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. Remote file transfer

Can you help? I wish to connect another computer to my Apricot via their serial ports, and to be able *from the other computer* to command the Apricot to send and receive files. Is this possible? I have failed in my attempts to do it with the MS-DOS command CTTY — I seem to remember reading in APRICOT FILE that CTTY doesn't work.

A. Don't bother with CTTY — use KERMIT at both ends. It includes a command SERVER which enables you to operate the remote KERMIT with commands from the local KERMIT.

To operate the remote machine in server mode, load and run KERMIT at both ends: type 'SERVER' at the local machine and KERMIT should respond with something like 'NOW IN SERVER MODE, TYPE YOUR ESCAPE CODE TO RETURN CONTROL TO YOUR MICRO'. When you enter this code (which is usually CONTROL-C, though it can be customised) the prompt for your local version of KERMIT should reappear. (This is a very important clue as to where you are — on your own computer, in the remote KERMIT, or in your operating system.) Both copies of KERMIT are now up and running; the prompt shows you that you are back in control of your own micro and communicating with *your* version of KERMIT.

Q. V30 and FORMAT

I am using the NEC V30 processor instead of the regular 8086. Apart from the disadvantages of the power-up noise and not being able to play music, I have found that my copy of FORMAT won't allow IBM-format formatting. Any suggestions?

A. The most widely-used version of FORMAT is probably 1.3.2, which doesn't support IBM formatting. Versions from 1.5.0 onwards *do*, and version 1.5.3 is probably the best to use with the V30 — there have been problems reported with the later version 1.5.5.

Q. GWBASIC on Xen

Any idea why I get 'illegal function call' when I try to use any of the GWBASIC graphic functions (CIRCLE, LINE etc) on my Xen?

A. We're grasping at straws here, but this is Eddie Bromhead's offering. It's based on the fact that graphics with Basic basically (yuk) came in three flavours:

- early PC/Xi: MSBASIC and GSX
- later PC/Xi: GWBASIC and GSX
- BIOS 3.1 systems: GWBASIC and 800400EG

Note that those are two different versions of GWBASIC, however. The problem could simply be that 800400EG is not being loaded first. Or the wrong combination of GWBASIC interpreter and graphics extension is being used. It is possible to run GSX and then GWBASIC — but *only* with the GWBASIC version from systems with BIOS versions before 3.1. And things will also fail if that early GWBASIC is being used with the EG screen driver — 800400EG for monochrome, 640200EG or 640350EG for low- or high-res colour respectively.

Q. Xen printers

I have a curious situation *vis a vis* printers on a Xen HD. Using the parallel cables from our original Apricot PCs and Xis, the Xen successfully drives the Brother M1109 and Olympia ESW103 (converted daisywheel typewriter); but our Epson FX-80 refuses to budge.

Further, *none* of the printers will work from WINDOWS. You can toggle the printer on with CONTROL-P at system level; if WINDOWS is subsequently loaded, the printer is disengaged. And when you exit from WINDOWS, the printer is logged on again. My dealers are mystified ...

A. Again courtesy of Eddie Bromhead, here are some thoughts about how to proceed for non-functioning Xen HD printers. For a start, it's worth checking that the hardware is ok:

1. Does the *printer* work on another make and model of computer?
2. Does the *cable* work on another model of Apricot? (You can't test the cable on a different computer, as Apricot parallel-interface cables are unusual in that not all pins are connected.)

If there's nothing work with the printer or the cable, move on to testing the Xen and MS-DOS:

3. COPY a text file to CON: (try \WINDOWS\WIN.INI, you'll probably need to look at that anyway sooner or later). If this fails, the Xen is duff.
4. Otherwise, set the CONTROL-P print toggle and repeat the COPY CON. If you get a printout, there's nothing wrong

with MS-DOS or the port; the problem lies in WINDOWS. Go to step 9.

5. If you don't get any printing, try COPYing the file to PRN — for instance, COPY \WINDOWS\WIN.INI PRN:. If this still doesn't work, the fault is with the printer port.

6. If COPY PRN worked, the problem might lie in a faulty BIOS or COMMAND.COM. Reboot using a simon-pure backup copy of your original system disk (you *do* have one, don't you?).

7. Still no printing? The printer might be set up as serial rather than parallel. Check by using WINDOWS' WCONFIG.

8. If there is still no printout, you have a duffness somewhere between the system software and the parallel port.

9. If any of these did work, load an application that drives the printer — like SuperWriter. Configure it for an Epson and try one of the facilities that are driven by Escape codes — use sub- or superscripts or italics, for instance, or change the print pitch (number of characters per inch). If that works, you can assume that not much is wrong with the hardware.

10. Now try to drive the Epson in bit-image mode; dump a screen, try printing a SuperCalc 3 graph, or output from GEM or GSX.

11. And if *that* doesn't work, check that the DIP switch settings on the printer permit eight-bit data transmissions.

If all this works, you can assume that the fault lies in WINDOWS. It's worth pointing out that WINDOWS sends all of its output in bit-image form — all shapes, including text characters, are created bit by bit. That means WINDOWS can't drive a daisywheel printer; and it must be configured properly for the target printer.

The first port of call should be the WIN.INI configuration file somewhere in the WINDOWS subdirectory. You can print or display this to check its contents.

If WIN.INI isn't set up correctly, selecting PRINT from the File Menu in PAINT produces an unhelpful 'Can't print' message in the dialogue box. Depending on the problem, WRITE might give the same message or make the print options unselectable. When WIN.INI is right, PAINT gives a straight printout; WRITE lets you specify print formats and a change of printer.

WINDOWS on a Xen HD comes set up for the C Itoh 8510A. Somewhere near the start of WIN.INI you'll see a section of text headed [WINDOWS] that includes the line

```
device=C-Itoh
8510/1550,CITOH,LPT1:
```

And later there's a section headed [DEVICES] listing the printers that WINDOWS can recognise — among which you should find a similar line, this time reading

```
C-Itoh 8510/1550=CITOH,LPT1:
```

Now, there are two configuration programs in the subdirectory \WINDOWS\UTILITY. WCONFIG is the overall system configurator, and it sets similar parameters to those offered under ACTIVITY or MANAGER — like bell and keyclick volume, serial port parameters, whether printer output should go to the serial or parallel port.

Assuming that WCONFIG is set correctly for a parallel printer, the other configurator — CONTROL — is the one we want. It sets up WINDOWS for a specific printer, incorporating some of the information in WIN.INI and including some of its own data. The latter needs to be reset each time WINDOWS is called; for example, you can't configure WINDOWS to assume landscape printing *automatically* — that has to be done each time you enter WINDOWS.

CONTROL seems to be good at *adding* to WIN.INI and even *modifying* it ... but it never seems to *remove* anything from WIN.INI. The only foolproof method of installing a printer into WINDOWS involves checking WIN.INI 'by hand' — so here goes.

12. Load CONTROL, select INSTALLATION from the menu bar, then delete the selected printer. Confirm the deletion (but don't delete the printer driver file from \WINDOWS — you might need it later). Continue doing this until *all* printers have been deleted.

13. Select INSTALLATION again and add a printer. Choose EPSON — don't ask to copy EPSON.DRV or you'll get a 'file can't be copied onto itself' message.

14. Stay in CONTROL. Select SETUP from the menu bar. Connect EPSON to LPT1:.

15. Use SETUP again with the PRINTER option. Make EPSON your default printer.

16. Leave CONTROL. The hard disk should work as WIN.INI is rewritten.

The problem with all the foregoing is that you must do it all in the correct order. Otherwise you get left with a [DEVICES] section in WIN.INI that has two entries for Epsons — something like this:

```
Epson FX/MX-80=EPSON,None
...
Epson FX/MX-80=EPSON,LPT1:
```

The first is the original, the second is the one you've just added. But WINDOWS just takes the first entry it finds and decides that no Epson is connected. We suggest loading WIN.INI with the NOTEPAD editor; then you can kill the unwanted line and save the amended file. Just to be on the safe side, leave WINDOWS and re-enter it (just in case there's something vital that's read only on start-up).

17. And finally: once you have a working WIN.INI, save it (under another name, for security purposes) — otherwise you might find that WINDOWS will cock it up for you. ◆

One flower per pot

TIMSHEET.WKS

A spreadsheet accumulator

By Ted Webber

People who like to collect sporting trophies are sometimes known as "pot hunters". Strangely, this association came to mind when trying to design a 'mug's accumulator' for timesheet analysis using Lotus 1-2-3: the crux of the matter is to put one, and only one, lotus lily (the cumulative total) in each pot.

What nonsense is this? Well, suppose column Z of a spreadsheet holds a year-to-date cumulative total, and columns B to Y hold current-period itemised costs; the simplest 'accumulator' first sets recalculation to manual, puts a formula of the type @SUM(Bnn..Znn) into column Z, then puts data into columns B to Y, and calculates. For the next period, all data in columns B to Y is first cleared — then off you go again.

If no mistakes are made, the results will be correct. But most of us do occasionally press the wrong key — and the worst key to press is 'calc'! Have you calculated yet? Just to be sure, press the key now ... Whoops! That's added the current period in twice.

In that case it's now very awkward to recover the correct cumulative total. Worse, the error may easily be overlooked. Calling myself a mug won't help at all.

Accumulating using edit

My Lotus manual is silent on this problem. But *Using 1-2-3* by LeBlond & Cobb *(now available from* DEALS FOR READERS *— Ed)* has some useful macros for the job, and there's one on DEALS FOR READERS disk AP014 in MACROLIB.TXT.

All these depend on doing the calculation in edit mode; the cursor loops down the column like a hypocycloid, testing for blank cells and so on. There are no formulae in the accumulator cells, so the calc key cannot affect them. But those cells cannot be 'protected' either, so an erroneous entry could corrupt the worksheet; unprotection and re-protection could of course be incorporated in the macro.

The best of these macros use features that are found only in 1-2-3 Release 2; I have Release 1A, and I believe Release 2 is not available for old-style 'real' Apricots *(that's right, officially it isn't: we do have a copy, though — Ed).*

Accumulating using @IF

I wanted a method that does the calculation with the calc key, partly because that is how I had trained my staff to use 1-2-3. I also wanted to be able to 'protect' the accumulator column, and to be able to correct data entry errors after calculation.

The solution uses a flag that is incremented as the last item in the calculation, and is cleared by a macro which clears the data from the body of the worksheet at the start of a new billing cycle. Each and every accumulator formula in column Z is modified to test if the flag is set to 1 (here, FLAG is a single-cell range-name):

```
@IF(FLAG=1,@SUM(Bnn..Znn),+Znn)
```

That is the (very elementary) principle. Notice that if the test yields 'false', column Z just copies itself. After long and deep cogitation to come up with it and make it work, I found the very same idea on page 3-106 of the SuperCalc 3 manual! Even so was the wheel re-invented so many times. Anyway, the point is that it works equally well in any spreadsheet program which has an @IF function, not just with Lotus 1-2-3.

Resetting

But that is not the end of the story. How can such an accumulator be reset for the beginning of a period? Replicating a column of zeros won't do — that would overwrite those lovingly-prepared and carefully-checked @IF functions, which must then be re-inserted. A Lotus macro or SuperCalc XQT could do this fairly safely, but I prefer to leave formula cells protected.

Assuming an otherwise cleared worksheet, my solution is to copy column Z to column W with a macro (or XQT), at the same time changing the sign. Add CALC and an erase-column-W to the macro: and Bob's your uncle — the formula resets its own cell value to zero.

I added two more columns for completeness of presentation: 'period total(X)' and 'previous total(Y)'. Now Xnn (with no accumulator) is

```
@SUM(Bnn..Wnn)
```

Ynn is

```
@IF(FLAG=1,+Znn,+Ynn)
```

And Znn — the accumulator — is

```
@IF(FLAG=1,+Xnn+Ynn,+Znn)
```

Setting calculation row-wise takes care of the 'circular' calculations which will otherwise arise.

A timesheet application

If you are lost, hopefully all will become clear from an example application.

A design office charges for its services on the basis of a person's cost per hour multiplied by the number of hours worked on each project. More than one person works on each project, and projects take several months to complete. Timesheets are submitted by staff at the end of each fortnightly pay period.

Clients are billed quarterly; there are three major clients who each commission many projects, starting and finishing at different times, plus many smaller clients with one project each. There are also the other functions of the design office (like filling in timesheets!) which are non-recoverable.

The design office does not prepare the invoices, but the accounts department wants to know how much to bill for what project to each client. Ideally, the spreadsheet would produce the invoices: this is partly catered for by allocating projects in groups, one client per page.

When faced with this problem, my options were ...

- write my own program
- find a commercial package designed for the job
- use a database
- use a spreadsheet.

I could certainly write a program in Basic or Fortran. This would ask for the person, the hours, and the job, and accumulate the hours in a person/job matrix. At the end of a billing cycle, it would print the invoices. But there were complaints that I was spending too much time on the computer, so I thought I should try to use commercial software.

I found two commercial packages listed for calculation of professional fees: TIMEMANAGER from Joint Computer Marketing, and TIMECOST by Interactive Applications. But our office is located thousands of miles from any supplier *(Ted Webber is located in Papua New Guinea — Ed)* and I decided it would be impossible to find out if either did the required job.

We had the DATAFLEX file manager, which looked as though it could do the job but was dauntingly complex. Learning DATAFLEX would have taken longer than writing a Basic program.

The concept of a spreadsheet was easy to understand and we had 1-2-3 on our Xis. So Lotus was used.

In the event, I now believe that it took longer to develop and refine the model than it would have taken to write a special program. And I have not found a way of entering data (hours) in the right place by nominating person and job: it is a case of hunting around for the right intersection on several screens. One could of course put up a wall map which identifies 'J.Bloggs' with column P and the 'Tolpuddle Sewage Treatment Works' with row 123, then GOTO P123. But such a system seems dreadfully crude. Also, the practical limit for number of staff is only 19 for printout neatness.

So although this example may contain some useful ideas, I do not recommend it universally for its designed purpose.

In use

The whole spreadsheet is 'protected' for use by clerical staff. Hours are accumulated in their cells, using the edit accumulator procedure. This is accomplished by the \I input macro (see below); it unprotects the particular cell, adds the entered value to the 'formula' in the cell, and protects it again. Other examples of edit accumulation (see above) add a CALC in edit mode to reduce the formula to a value; but the formula gives a history of entries, and enables easy correction. In any case, the value which is the result of the formula appears on the screen and the printout.

The general layout

ROWS

1:	Titles (cell X1 has the file name)
2:	Column letters, for identification on the printout
3:	Staff names or abbreviations
4:	The staff grades
5:	Hourly rates corresponding to the grades in row 4. This row is used to calculate *rates* * *hours* in column X
251:	The calculation flag is set in C251
252:	Total All Hours — used to check that entries are complete

Last Row: the row at the end of each page (or client) is used for page (or client) totals

COLUMNS

A:	[width 15] Project names or abbreviations (mnemonics only)
B:	[width 10] Project number — these are unique to each project, and used by the \S sort macro to sequence the projects
C to U:	[width 8] Data entry area for staff hours
V to W:	[width 8] Two types of disbursements, purely additive
X:	[width 10] Period Total — `$C$5*Cnn+$D$5*Dnn+ ... +Vnn+Wnn` (note that @SUM cannot be used here)
Y:	[width 10] Previous Total — `@IF(FLAG=1,+Znn,+Ynn)`
Z:	[width 10] Current Total — `@IF(FLAG=1,+Xnn+Ynn,+Znn)`
AA:	[width 4] Accounts Dept account sub-code
AB:	[width 4] Row number, for identification on the printout

With these widths and using a compressed 17cpi font, the complete spreadsheet can be printed on paper 15in wide.

The worksheet is too large for all the detail to be printed here, but an application to APRICOT FILE with SAE and a formatted disk will produce a copy. It will also appear on a forthcoming DEALS FOR READERS disk.

The macros at AC1 onwards

column	AC	AD	AE
row 1:			
2:		*** MACROS ***	
3:	\P	/PPCAA~	PRINT: Clear and Align
4:		OMR240~	Right Margin, 15in wide
5:		ML0~H@~S\015~	Left Margin, Date, comp
6:		QRA1..AB55~	First page
7:		GPRA56..AB98~	Print WB Cap Wks
8:		OBRA1..A7~	Repeat headings
9:		QGPRA101..AB198~	Print architectural
10:		GRA200..AB252~	Print other
11:		GPQ~	Go; Quit
12:			
13:	\R	{GOTO}C8~/FS~R~	File REPLACE
14:			
15:	\S	/DS	SORT by Project Numbers
16:		DA21..AA97~	WB Cap Wks
17:		PB21~	Sort on Column B
18:		A~G/DSDA101..AA147~	... etc.
19:		PB101~	
20:		A~G/DSDA201..AA249~	
21:		PB201~	
22:		A~G/DSDA151..AA197~	
23:		PB151~	
24:		A~G	
25:			
26:	\A	{goto}C101~	Beginning of each page
27:			
28:	\O	{goto}C201~	
29:			
30:	\W	{goto}C21~	
31:			
32:	\C	{goto}C151~	
33:			
34:	\E	/REC8..W250~	ERASE Data Fields
35:		{GOTO}C251~0~{GOTO}X1~	Flag to prevent calculation
36:			
37:	\X	/REC8..W250~{GOTO}C251~1~	RESET all
38:		{GOTO}W8~-Z8~/C~W9..W17~	Reset "Total"
39:		/C~W21..W97~/C~W101..W147~	Replicate column Z in W
40:		/C~W151..W197~/C~W201..W247~{CALC}	
41:		/REC8..W250~	Reset "Previous"
42:		{GOTO}C251~{CALC}	
43:		0~{GOTO}X1~	Flag "off"
44:			
45:	\M	/PPRAB1..AE60~GPQ	Print MACROS
46:			
47:	\F	/PPOOCQRX1..Z252~GRC5..W5~G	Print FORMULAE
48:		RC252..W252~GPOOAQQ	
49:			
50:	\U	/WGPD	UNPROTECT (Global)
51:			
52:	\Z	/WGPE/RPC8..W20~/RPA21..W251~	PROTECT all inc. data
53:			
54:	\I	/RU~{EDIT}+{?}~/RP~	INPUT cumulative hours

In operation, the cursor is moved to the appropriate cell and CONTROL-I is pressed. In the top-left corner appears the current 'formula', consisting of the previous entries since the last billing (eg 2.35+6+.12+); the hours are entered (say 1.5, so now the 'formula' now reads 2.35+6+.12+1.5); and when Return is pressed, the calculated value (9.97) appears in the cell.

Changing staff data can only be done at the beginning of a billing cycle, although if a staff column is vacant it can be used at any time. Spreadsheet columns should not be inserted or deleted within the main body.

To keep the printout tidy, a pageful of projects is allocated to each client and the total length is kept constant. The project number column (B) is filled initially with dummy Zs, so that sorting will always push used rows to the top of the page. Adding a project is straightforward, but deleting it on completion takes a little thought — the row must not be deleted, and range erasure is not a good idea either. I blank the item in column A, put 'Z' in column B, and sort: this dumps the unwanted row at the bottom of the page. This should be done only at the beginning of a billing cycle, when the hours data area has been erased and protection is off.

At the start of a year, one may decide to start with a completely blank worksheet. To do this, macro \U unprotects the sheet: \E erases the hours data area, \X resets all totals, and \Z re-protects.

At the end of a pay period data entry (not the end of a billing cycle), you can optionally CALC and print. The totals to date will be shown in column X, but Y and Z will be unchanged.

At the end of a billing cycle, unprotect with the \U macro, set C251 to the formula +C251+1, then CALC, and print. C251 now contains '2', so another CALC (in error) will not upset the accumulator columns. To be on the safe side, save a backup file under another name at this stage, then use \E to erase the hours data and \Z to protect again. (There's an opportunity for another macro here!)

In conclusion

Spreadsheet programs are sold for their amazing size, speed, and modelling capability. But this exercise has highlighted for me their limitations. In particular, 'large' equates with 'unmanageable'. Or do I need to be a bit more clever, with lots of help screens and even more sophisticated macros? Let's get back to Basic or Turbo Pascal! ◆

Background

Printers in Europe:

Anti-Japanese update

Epson reportedly paid something over £150,000 (some say £250,000) for adverts in the first six lefthand pages of the 12 April *Financial Times*.

And was Epson selling its printers there? Well, yes and no. The ads attacked the EEC anti-dumping levies that come into force next month for imported Japanese printers. The levy is an attempt to redress the imbalance in trade; Epson's argument is that initially it means higher prices for the end user (because imported printers are cheaper than European production) — and in the long run could lead to the withdrawal of Japanese products altogether.

That might be a good case to make. After all, there is an immediate precedent — the American restrictions on imported Japanese DRAMs has lead to a severe shortage of memory chips and consequent price rises. It has even affected the smallest scale of supply: our DEALS FOR READERS service is having to scratch around even now to find Apricot expansion boards at a reasonable price.

On the other hand, most of the big names have been setting up European plants to evade the levies — Citizen is building in Scarborough, Fujitsu is in Malaga, NEC and Epson itself are in Telford. Epson has 55 acres there, the largest factory in the area; in May that gets the official ribbon-cutting from Environment Secretary Nicholas Ridley, but it is already producing 30,000 printers a month.

The levy might lead to the disappearance of some of the bargain-price units like Seikoshas, many of which are rebadged by European suppliers; and some of the better deals in 24-pin NLQ printers are also rebranded imports, like Shinwas. But European manufacturing (or assembly) should mean that prices for the best-known matrix printers shouldn't change much for UK users as a direct consequence of the EEC action. Indeed, there's a view that the EEC's aims are based more on creating employment here by encouraging Japanese firms to go into local manufacturing — rather than supporting the local printer makers (Olivetti, Newbury Data, Mannesmann Tally, Honeywell, Nixdorf and Philips did the European lobbying).

So why Epson's investment in advertising? Because the current levy doesn't apply to *components*. Printers are so relatively cheap primarily because of the volume production of components in the Far East for a worldwide market. But if the lobbying gets more intense, the EEC could well be persuaded to penalise imports of parts as well. ◆

Letters to the Editor

Readers' Writes

Upgrading to MS-DOS 3.2

In one of my large batch files, I found that switching from MS-DOS 2.11 to 3.2 reduced execution time from 20.5 seconds to 1.9! That alone seems like a good reason for changing ... These notes may help others making the switch.

- The patch given in APRICOT FILE Release 2.9 to increase environment space worked fine for 2.11. When moving to 3.2 I wished to use the /E switch with SHELL that assigns a larger environment area; but when this was added, my AUTOEXEC.BAT was not executed and no date/time was requested. Using the /P switch as well cures this, such that the relevant line in my CONFIG.SYS now reads ...

  ```
  SHELL = COMMAND.COM /P /E:1024
  ```

  (Here '1024' is the size of the environment in bytes.)

- With MS-DOS 2.11, you can use this to remove ECHO OFF from the screen when running batch files that need the command:

  ```
  ECHO OFF^[2K^[2A
  ```

  It helps tidy the screen a little, and incidentally works on an IBM-compatible. It doesn't however seem to work with 3.2, as the 'OFF' is not recognised if it is followed by something else. The fix is ...

  ```
  ECHO OFF
  ECHO ^[2K^[1A^[2K^[1A
  ```

- Similarly, ECHO with blanks no longer echoes a blank line. To do that there are four options (obviously you don't type the underlined bits, just press the relevant key):

  ```
  ECHO space x'00'
  ECHO space x'FF'
  ECHO space backspace
  ECHO.
  ```

  The last one — ECHO followed by a dot and no spaces — doesn't work for MS-DOS 2.11

- With MS-DOS 3.2, the public-domain sorted-directory utility SDIR (version 2.5) doesn't work if you try to list a directory other than the one you're currently in. This does work with 2.11.

- I found that MS-DOS wouldn't boot when a null modem cable was connected — this was due to have the DTR/DSR and RTS/CTS protocol operational. The problem was solved by changing those to 'off' in the boot sector. To effect the amendments, you can use LABEDIT to display and change the current values. Or use DEBUG:

  ```
  -L 100 0 0 1   loads the boot sector into memory from
                 drive A
  -D 1B0         position 1BD/1BE should be 01 01
  -E 1BD 00 00   set to no protocol rate for RTS and DTR
  -W 100 0 0 1   write sector 0 back to drive A
  ```

- The key repeat rate which I had patched for MS-DOS 2.11 reverted back to 'slow' with 3.2; this can also be corrected in the boot sector. (Funny how ACTIVITY never included anything to set this variable.) Again, use LABEDIT or DEBUG:

  ```
  -L 100 0 0 1   loads the boot sector into memory from
                 drive A
  -D 190         position 193 should be about 05
  -E 193 02      set to fast repeat rate
  -W 100 0 0 1   write sector 0 back to drive A
  ```

Guy Sturges
Trélex, Switzerland

Faster-branching batch files

Mike Florey's F.BAT (APRICOT FILE Release 3.6) is safe and sound. It tests for no parameter or an invalid one, and it checks for upper and lower case parameters on the command line. Unfortunately, this can be a slow process — especially if there are many possible branches as a result of the tests. This illustrates a neater and faster implementation, providing you accept that an invalid parameter will produce only an error message.

```
ECHO OFF
ECHO ^E Demo — case-independent branching
REM  Test for no parameter:
IF NOT %X==X GOTO %1
GOTO EXPLAIN
:DATA
FORMAT B:VOLABL /I
DISKCHK B:
REM Instant exit (see below)
XIT
:SYSTEM
FORMAT B:VOLABL /SA
DISKCHK B:
XIT
:APE
FORMAT B:VOLABL
DISKCHK B:
XIT
:EXPLAIN
REM Use ECHO for instructions etc here
XIT
:%1
GOTO EXPLAIN
```

In line 2, ECHO ^E is preferable to CLS because it homes the cursor as well as clearing the screen. Note that it can also be followed by a line of text, which is displayed on the top line of the screen.

The important line is the fourth one with that GOTO %1. Labels are case-independent, so the branch will be equally effective for a parameter in upper or lower case — for F DATA or F data on the command line, for instance.

XIT.BAT is a dummy line containing CONTROL-Z to force a quick exit — it replaces the slower search for the label END after checking for bad blocks. You can't name it EXIT.BAT, because EXIT is a valid MS-DOS command that does something else. You can create XIT.BAT simply as follows:

```
COPY CON XIT.BAT [return]
^Z [return]
```

Ted Webber
Papua New Guinea

A Turbo Pascal Primer

Part Five: Control Structures

By Mark Whitehorn

Simple computer programs consist of a series of instructions which are executed sequentially. This is all very well, but sequential operation like this is very limited; more complex programs make use of those commands which allow the programmer to alter the order in which the instructions are executed.

There are two general reasons why it is usually necessary to direct the flow of the program in this way. First, the program design may specify that certain bits of the program are repeated a number of times. Clearly this could be achieved by writing the necessary code over and over again; but this is tedious, boring, inelegant and wastes space.

Second, the programmer often doesn't know which path will be followed each time the program is run. Most useful programs are adaptable and respond to the user's input: so the programmer must include statements in the program which will interpret this input and modify the flow of the program in response to it.

This requirement to control the flow of execution within a program is met by a collection of commands known as 'control structures'.

Early versions of Basic had limited control structures; everything had to controlled with IF, FOR and GOTO. This in turn led programmers to write complex programs which were difficult to debug because of their abstruse control of flow.

It's important to realise that *any* flow structure can be built up with simple IFs, FORs and GOTOs — no extra control structures are essential. Pascal, however, is provided with several different control structures; and the extras in Pascal make life easier and programs more readable.

So control structures are all about controlling when (and indeed, if) certain bits of code are executed. All of the structures described below will allow you to control the flow around bits of code ranging in size from a single command up to many thousands.

IF ... THEN ... ELSE

This one will be familiar to Basic programmers. As the name suggests, it applies when you have a piece of program which should only be used under certain conditions — **IF** (a particular condition is true) **THEN** (do something) **ELSE** (do something different). And it can read very much like the English language: **IF** the Penguins are hungry **THEN** feed them, **ELSE** allow them to play in the bath.

```
If Count = 42 Then
 Writeln('Life, the Universe and Everything.')
Else Writeln('Try again.');
```

Digression on truth

Now that we have one control structure under our belts, it is worth digressing to consider two general points which apply to all five control structures.

The first is the problem of Truth. In essence, the condition which must be true (in the above example, it is the bit just after the IF) may be any condition which Pascal can evaluate as true. The Turbo Pascal manual goes into great detail here, and is worth consulting for exact definitions; but it also tends to be rather turgid, because it is so detailed.

A good rule of thumb is to use your common sense. If something seems to be true to you, Pascal will probably think so as well. For example, suppose that early in the program a variable called *Count* is set to the value of 10. Assuming that *Count* hasn't been altered, would you expect it to be true that *Count* is greater than 9 at a later stage in the program? Of course you would; and so would Pascal. So we could write:

```
If Count > 9 Then
 Writeln('Count is now 10 or greater.');
```

We could put this anywhere in the program to tell us when *Count* reached double figures. (Like most bits of demonstration program this might seem totally valueless, but the general principle is very useful.)

Digression on size

The second important consideration is that the bit of program which is controlled by a control structure can range in size from a single command to many thousands. This has important implications for program-writing:

- If you want to control the flow around a single command (such as a *Writeln*), the syntax will be as shown in the *Count* example above.
- If you want to have two or more statements under control, remember to wrap them up with a *Begin* and *End* like this:

```
If Balance < WithDrawals Then
Begin;
 Writeln('Sorry, you're out of money.');
 Credit := 0;
End;
```

■ If you have more than about ten commands, think about declaring them as a separate procedure and then call that by name. For instance:

```
{A Procedure called DealWithOD has been declared
earlier in the program}
If Balance < Withdrawals Then DealWithOD;
```

Thus endeth the digression on Truth and Size.

FOR ... TO ... DO

Again this should be familiar in concept (if not actual execution) to a Basic programmer. Suppose you have a piece of program which you wish to be executed five times. This can be accomplished like so:

```
For Count := 1 to 5 Do
Writeln('The number I have reached is ',Count);
```

This will repeat the *Writeln* statement five times and produce the following output:

```
The number I have reached is 1
The number I have reached is 2
The number I have reached is 3
The number I have reached is 4
The number I have reached is 5
```

We can often use this in a more flexible way, however. Suppose you are writing a program to keep a check on your cheques (oh look, a pun) which is run at the end of each month. At the start of the program you ask the user for the number of cheques which have been issued during the current month and put the answer into a variable called *NumberOfCheques*; later in the program you want to subtract the amount of each cheque from your balance.

```
For Count := 1 to NumberOfCheques Do
  Begin;
    ClrScr;
    Writeln('Enter amount of next cheque.');
    Readln(Amount);
    Balance := Balance - Amount;
End;
```

Each time the program is run, this section of code will be executed just enough times to deal with all your cheques.

CASE ... OF

The CASE ... OF construction will not be familiar to many Basic programmers. It exists to cope in a tidy manner with all those embarrassing occasions in Basic when you'd be forced to use lots of IF statements — perhaps a section of program which sorted the input from a menu.

In Basic it might look like this:-

```
210 IF ANSWER = "A" THEN GOTO 500
220 IF ANSWER = "a" THEN GOTO 500
230 IF ANSWER = "B" THEN PRINT "Correct"
240 IF ANSWER = "b" THEN PRINT "Correct"
250 IF ANSWER = "C" THEN GOTO 700
260 IF ANSWER = "c" THEN GOTO 700
270 IF ANSWER = "D" THEN GOTO 800
280 IF ANSWER = "d" THEN GOTO 800
```

In Pascal we can use a CASE ... OF to deal with this.

```
Case Answer of
 'A','a' : FirstChoice;
 'B','b' : Writeln('Correct');;
 'C','c' : ThirdChoice;
 'D','d' : FourthChoice;
End;
```

This can be read as follows: "if it is the case that *Answer* is either 'A' or 'a', do the procedure called *FirstChoice*. If *Answer* is 'B' or 'b', write the word 'Correct' to the screen. If *Answer* is 'C' or 'c', do the procedure called *ThirdChoice*; if *Answer* is 'D' or 'd', do *FourthChoice*".

Turbo Pascal actually goes beyond the original specification of Pascal to provide us with an ELSE option that can deal with any selections you haven't foreseen. This can be quite useful; but it should probably be avoided if you are thinking of translating the code into another implementation of Pascal. Yes, I know that you won't find a better version of Pascal on a micro, but some of our students port programs to our mainframes and then run into problems.

Repeat ... Until

Like the others, this one works as a sort of contraction of an English phrase — REPEAT (a particular action) UNTIL (some condition becomes true).

```
Repeat
  ClrScr;
  GotoXY(5,5);
  Writeln('Choose an option between 1 and 10.');
  Readln(AnswerFromMenu);
Until AnswerFromMenu < 11;
```

Notice that the control structure is written with the condition at the end, *after* the code to be executed. This ensures that the bit of code must be done at least once, before Pascal has a chance to decide if the condition is true or not. So use the *Repeat ... Until* construction when it is essential that the bit in the middle is done at least once.

Also, note that in this particular control structure we don't have to use a *Begin* and *End* to wrap up the section of code.

While ... Do

This one is functionally almost exactly the same as the previous control structure — except that in this case the condition is evaluated first, *before* the code is executed. So it is perfectly possible for the code to be missed out entirely.

```
While NumberOfCheques > 0 Do
Begin;
  ClrScr;
  Writeln('Enter amount of next cheque');
  Readln(Amount);
  Balance := Balance - Amount;
End;
```

Last words

This has been a simple, get-you-going introduction to control structures. Note that by using sensible names for variables you can write programs which read like the English language:

```
Repeat
  GetInputFromTheUser
Until AllTheInputIsLegal;
```

All of this is perfectly valid. So, write lots of clear concise code to practice control structures! ◆

If you wish to run these examples, some or all of the following declarations will be necessary. Note that the procedural declarations are 'dummy' ones — the procedures don't actually do anything, but Turbo Pascal needs then to be present in order to compile the program.

```
Var
  Count,
  NumberOfCheques,
  AnswerFromMenu      :Integer;
  Balance,
  Withdrawals,
  Credit,
  Amount              :Real;
  Answer              :Char;
  AllTheInputIsLegal  :Boolean;
{---------------------------------}
Procedure FirstChoice;
Begin;
End;
{---------------------------------}
Procedure ThirdChoice;
Begin;
End;
{---------------------------------}
Procedure FourthChoice;
Begin;
End;
{---------------------------------}
Procedure GetInputFromTheUser;
Begin;
End;
{---------------------------------}
Procedure DealWithOD;
Begin;
End;
{---------------------------------}
```

Notebook

Maintenance wanted

By Graham Hutchinson

Thanks for the excellent Terence Green/Simon Potter *Notebook* article in APRICOT FILE Release 3.6 about the future for Apricots. I agree with much of it, but I take issue with one point — Messrs Green and Potter talk of maintenance for 'original' Apricots being a problem for the future; personally I think that maintenance is a problem that APRICOT FILE should address now.

My own experience of Apricot Computer Maintenance Ltd is hardly favourable. I would dearly like to find a maintainer for my beleaguered F10 who is quick, cheap and competent, and who will take business from private customers.

May I suggest that APRICOT FILE publishes a list of independent maintenance companies for 'original' Apricots? I imagine that many readers would be prepared to contribute suggestions and comments — and apart from solving my immediate problem, the list could help to keep a few maintainers going after Apricot pulls the plug ... How about it, Mr Editor?

Good idea. Any and all contributions are welcome for a Register of Recommended Maintenance Companies prepared to deal with Apricots: we'll run it as a regular list. Just to get it started, here are three offerings.

CSL (Computer Services Ltd)
51 Colney Hatch Lane, London N10 1LJ
tel 01-883 1186

... we've used these people ourselves and have little complaint. As with most such firms, though, there's a minimum call-out charge.

Martin Filmer Computing Services
10 Leafield Close, London SW16 3EY
tel 01-761 0425

... Martin is a knowledgeable one-man-band who occasionally writes for APRICOT FILE, packages our own OMEGA Apricot-IBM upgrade systems, and fixes computers. He's worth trying if you're within 50 miles or so of him.

Matrix Computers Ltd
Hawthorn House, Hawthorn Road, Chippenham, Wilts SN15 1BU
tel 0249 654175

... claims to specialise in Apricots (among other machines). We don't have any reports on this firm, but it does employ some ex-ACT people and seems keen for Apricot business. ◆

Customise your keyboard

Part Three: Adding ALT keys, matching IBM 3270

By Guy Sturges

With BIOS 3.1 on a PC or XI, you can use ALT keys for some useful results — in particular, it's possible to input a three-digit decimal value on the numeric keypad in ALT-shifted mode to produce a single character of the desired ASCII value (à la IBM).

I came across this very enthusiastic fellow who runs a computer shop in the depths of Valais, Switzerland, who gave me more useful information than any dealer in England. (I suppose that's not difficult. I got no information at all from Apricot themselves — they don't talk to plebs — and have found no dealer who was really on top of Apricots. The bulk of my Apricot information was acquired via APRICOT FILE.) Anyway, it was he who put me on to the track of the ALT key possibility.

So how do you add an ALT key layout?

Neither KEYEDIT nor ACTIVITY supports ALT-shifted keys. At the beginning of each *.KB file produced by these programs there is a text/comment block of 128 bytes followed by 1,024 bytes of the key layouts and strings that are assigned to individual keys in normal, shifted and CONTROL-shifted modes (each of the key layouts is 208 bytes long, the remaining space is allowed for strings).

Each of the permissible 104 keys uses two bytes to denote its value. It can either be a two-byte address to the string assigned to the key or a one-byte key attribute plus a one-byte key value.

Now, there is an area of 1,024 bytes in the lower part of memory into which the Apricot KEYS program installs any *.KB table. This limits the size of a keyboard table, and in particular the maximum length of strings that can be assigned to individual keys. It does, however, allow room for different values for normal, shift, CONTROL, ALT, and strings (but not very many).

Producing an ALT-shift is easy. Go into DEBUG or SYMDEB (*aka* MASM) and move everything about — but *every* key that is associated with a string has to be modified. ALTernatively, remove all your strings; fairly simple, but you lose half the advantage of a *.KB file.

Or you can use the GTfixkey from my GuyTec disk. This divides up an area of 2,000 bytes into 128 bytes followed by nine 208-byte sections. The normal, shift and CONTROL areas stay where they are; and an ALT key layout is loaded from another *.KB file into an area after the strings. You prepare this ALT key layout as the 'normal' part of a separate keyboard file, together with any strings if you wish. The program adjusts everything so that the strings are all last and the normal, shift, CONTROL and ALT key layouts are first. Save it, then load it with KEYS — and it works.

The important bit is that the utility changes the value of Shift-Caps Lock to 09 C0, denoting the ALT key, and the Caps Lock key in the ALT key layout is given the same value. So ALT mode is selected when Shift-Caps Lock is held, and the keyboard reverts when they are released.

But how can you type the three-digit decimal value to generate one ASCII character? It's necessary to set special values for the ALT-shifted number-pad keys: F0 C0 for 0, F1 C0 for 1, and so on — C0 is the attribute denoting a special key. (This is done automatically when the GuyTec utility loads in the ALT keyboard file.)

So where are the disadvantages? Well, you get only 192 bytes instead of 400 bytes for strings. And you can't edit your keyboard any more, because the editors get confused with strings in the wrong place — though that's easily resolved with GTfixkey, as the utility can save the *.KB file with the ALT keyboard still loaded. You can then edit it with KEYEDIT or ACTIVITY and then relocate everything after editing. (Incidentally, the relocation process also compresses some Apricot escape sequences that the editors create as strings, so that they do not require space in the string area.) It is always worth keeping your original ALT table around in case you do something wrong or destroy your working keyboard file.

To get around the first disadvantage, the Apricot KEYS program needs to be replaced — no great loss as it doesn't support subdirectories. The GuyTec GTldkeys utility gets around this path problem; and version 2.10 creates a new 2KB area into which the *.KB file is loaded, so there is room for 1,216 string bytes as well.

One point to note if you have a key which points to a string after byte 1,024: when KEYS is used to load the *.KB file, the string is chopped off. The keyboard driver gets muddled and will lock up the machine if that key is pressed.

Incidentally, although the editors restrict the length of the strings that can be assigned to keys to 15 characters, in fact there is a limit of 255 characters per string and a theoretical total *.KB size of 32K.

The 3270 layout

I generally use a 3270 terminal, a type of IBM workstation that connects to a mainframe and which can work in full-screen mode. By comparison with the Apricot, it has a different layout for the cursor keys — among others. It seems that the device I used for the longest consecutive period is the one that becomes the most familiar; and eventually I was making mistakes on both.

You can easily adapt the keyboard to different layouts. For a start, you can move the keytops around by gently levering them off and switching them on to different keys (you can even move those with an LED — the lamps are not fixed to the keytops).

I tried moving the keytops around on the Apricot and setting up the corresponding keyboard file. It worked fine in MS-DOS but not in GWBASIC or Symphony. Lotus was no help; in the end my Swiss enthusiast said there was a keyboard table at the end of GWBASIC — and sure enough, there is.

You can extract this with DEBUG or SYMDEB (MASM), work on it with a keyboard editor, and put back the new version. This was so successful I even added commands to the MicroScreen's keys — like CHR$, GOTO, GOSUB, SYSTEM and so on.

Two down, one to go. I found the device driver in the master library for Symphony, performed the same magic, and regenerated my driver set. Hey presto! I now use the same keyboard layout on the Apricot as on the 3270 terminal — and no more mistakes. (Lotus says modifying its software invalidates the warranty, but as I was unable to get any useful assistance from Lotus I don't suppose it matters. My rule now is: buy the cheapest software possible. If it works, it's good value. If it doesn't, you won't get any less support than if you'd paid a lot for it.)

The GTfixkey utility for making ALT keyboards also has a feature for moving around the keys in the *.KB file to match a 3270 terminal and back again.

Of general note: the existence of *two* Caps Locks may not be well-known. One puts letters into upper case, but CONTROL Caps Lock also puts the numeric keys into upper case.

And you may not be aware that the MicroScreen calculator uses the CLEAR key to set the calculator to zero. After moving the the keys around for 3270 mode, the CLEAR and DELETE keys are exchanged. The calculator now uses the key labelled DELETE to clear the calculator.

Next month: how to match the MicroScreen display with keyboards, and uses for ALT keys and IBM emulators. ◆

Contributors in this issue

Dennis Jarrett is Editor of the newsletter. His magazine credits include being founder-editor of *16 Bit Computing*, *Office Automation Report*, *Practical Computing* and *Which Computer*. Among his books are *The IBM PC Book*, *The Sirius/Victor 9000 Book*, *The Electronic Office*, and the brand-new *Fast Track: Information Technology for the Fast-Track Executive* (£17.95 inc P&P from DEALS FOR READERS).

Edward Bromhead is Reader in Geotechnical Engineering at Kingston Poly, where he uses a couple of Apricots to model the behaviour of soil slopes and groundwater (mainly using his own Fortran software and GSX). He specialises in earthworks (eg for motorways) and landslides (he predicted the collapse of Carsington dam in November 1983: it fell over eight months later).

Roger Gann works full-time running the DEALS FOR READERS section of the newsletter as well as contributing words, ideas, humour, hints, legal advice, spittle for mass-mailings, etc. He was formerly the principal writer on 16 BIT COMPUTING; before that he was a builder (and a pioneer Sirius user), a lawyer, and a kind of grandiose bank clerk.

Terence Green is an unemployed Apricot owner (currently looking for a job). Formerly an unlicensed truck driver in Lesotho, he started on IBM 370s and worked his way down to micros; these days he minds the kids and scrapes a living washing dishes (and writing occasionally for APRICOT FILE).

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.

Guy Sturges worked as a programmer for the largest software house in England before moving to Switzerland. He is now working in telecommunications, systems and programming in Geneva. Guy purchased an Apricot PC in October 1986 ("it was only then that they were a reasonable price") and is putting all the pieces together to make it work as it should.

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).

The Workshop

GSX revisited

By Eddie Bromhead

While browsing through some DEALS FOR READERS disks, I came across a goody — a later version of the GSX screen driver than the one I'd been using for some time. Now it just so happens that I used to collect GSX screen drivers and noted the improvements wrought with every one, so I pounced on this.

It came at a time when I had been sensitised to the defects in the earlier GSX drivers after trying to mount some of my software on an old machine which did not have the latest software. That version of the screen driver didn't draw circles properly, and so the program performed very badly indeed.

If *you* have an old version, I can assure you that it's time to move on up unless one of the very few now discontinued facilities is of vital importance to you. The latest version is just as happy running under any BIOS version and ROM chip set as is the most primitive under MS-DOS 3.20 and BIOS 3.1.2, and can be had from DEALS FOR READERS.

When I started to investigate GSX three or four years ago, what made life particularly difficult (apart from serious mistakes hanging up the machine) were the allied facts that (a) the documentation and the implementation don't match, (b) the documentation is pretty difficult to understand anyway, and (c) hard copy is difficult to obtain. I can't do anything about (a) or (b), but I have put a little distilled experience into a section of this article on hardcopy. Although I have slanted it towards the ACT Writer 10 and 12 printers, the information should be translatable for Epson users.

The loader

We all know how the GSX system works (don't we?). It needs the three files GRAPHICS.EXE, ASSIGN.SYS, and a screen driver. I'll be confining the majority of my remarks to the PC/Xi, although some of it will apply to F series machines too.

(For obvious reasons, CP/M-86 and Concurrent GSX is a bit of a closed book. I have only been able to track down two versions, the one supplied with Concurrent 2.0 and XIOS 1.4 dated 16 Dec 83 and 10K long, and the somewhat later version of 5 July 84 which is 12K long and was supplied with version 3.1 and XIOS 1.6. Neither of these is particularly sophisticated in respect of text or marker scaling, but they do offer a good range of patterns.)

As far as I can tell, there are three versions of the GRAPHICS.EXE loader program. They are functionally identical; the only differences that I can see are that the earliest version was very coy about GSX being for 'MS-DOS and PC-DOS' and the latest doesn't put the Digital Research copyright notice on the screen at all (secretive, isn't it? You can't find this message — well, not easily — by using DEBUG on GRAPHICS.EXE).

If your Apricot came with ACTIVITY, you probably got the latest version, but it really doesn't matter a jot ...

Screen drivers

The big differences are in the screen drivers.

If your machine was supplied with a very early release of the GSX system, like my single-sided PC, it's likely to have come with a screen driver named DDACRT.SYS. You probably also got BIOS 1.x, and snow on the screen when it scrolls, and many other raw deals.

On the other hand, a considerably enhanced GSX driver also named DDACRT.SYS was still being shipped up to about BIOS 2.4. At that point, probably to make it easy to identify GSX drivers across the F1/Portable/PC divide, the file names were changed: and PC owners will have found PC.GSX appearing in their issue disks. (For what it's worth, DDACRT means 'Device Driver Apricot Cathode Ray Tube'; PC.GSX is a trifle more self explanatory!)

Just compare the facilities offered by the BIOS 1.4 DDACRT.SYS and the latest PC.GSX in the accompanying table. DDACRT.SYS drivers have less of almost everything than do the PC.GSX drivers, except in one significant respect: line thicknesses, and large-size bit-image characters. Basically, DDACRT offered a choice of line thicknesses (1, 3, 5, 7 or 9 pixels wide); the later PC.GSX drivers only have one line thickness. Also, DDACRT.SYS had one character font which could appear in four sizes — either the same as the screen font, or enlarged two, three or four times (so that the characters appeared the same as those used in the ladder-based utilities). Early DDACRT fonts could not be rotated either — except in 90° steps.

The very earliest DDACRT drivers didn't recognise a mouse, or they needed MOUSE.COM to be loaded. All the early publicity shots of the Apricot showed it with a Microsoft serial mouse, and not the trackerball keyboard mouse which connects to a socket in the back of the keyboard so thoughtfully provided right from the beginning. The drivers responded erratically (if at all) to the enquiry functions — or, more specifically, they returned duff information to the OPEN WORKSTATION call. The information returned is often unreliable — for example, does the device support colour? Yes. Ok, so how many colours? Two!

Later on ACT coded the device drivers in assembler (I think the earlier ones were done in MS-Pascal) and took the opportunity to introduce the 'stroke font' as the default character set. I know some people don't like it very much, but at least it supports rotation and a huge range of character sizes. The benefit of this is that it is possible to emulate graphics terminals like the Tektronix so very much more easily. By being the first, or default, font, the driver should return correct answers to the questions "is the font rotatable?" and "how many sizes are there?". There are actually 400 different sizes for text, and for markers.

Again for what it's worth, the earliest DDACRT drivers do not work on a Xen, because they cause the screen to flicker. This is correctable (by changing a single byte in the code with DEBUG: see my article on 132COL in APRICOT FILE Release 3.5) but it really isn't worth attempting. Drivers which were coded in assembler are compatible with the Xen.

By the time Apricot had decided on the name change to PC.GSX — sometime between mid 1984 and early 1985 — a lot of improvements had been made to the driver. A number of the bugs were fixed, not least those which appear in the release notes on the issue disks at about this time — things like circles of one pixel radius crashing the system. Also, the release version number is now in plain ASCII text and so can be found in the driver with the aid of DEBUG. It appears that PC.GSX drivers were numbered release '1.6.x' while the DDACRT.SYS numbering was relatively inconsistent (although the release notes do refer to version '1.5.x' after a while).

An important change occurred at about the time of the release of ACTIVITY: the fonts in PC.GSX changed places! The bit image font became number 1, and the stroke font became number 2. The net effect of this was to make any program which used the default font probably work incorrectly — you needed to modify the program to select a font specifically.

Along with the ACTIVITY changes, the later PC.GSX drivers offer some of the functions from GSX 2.0 (so *that's* what the codename for GEM was ...). These are notably in the area of bit block moves and in accessing the mouse, changing the cursor form and so on.

You need these functions when picking up objects and moving them around on the screen if performance is to be reasonable. The alternative is to draw the object in XOR mode: when it is moved, redraw it at the original position (to blank it out, and restore the original): then draw it at the new position. This is still a good strategy if the object to be moved is simple, like a line or a letter or a box; but where it is at all complex you can see the blanking and redrawing going on. With a bit block move, you can do the same XORing with a rectangular area of the display. Function op-codes of 100 and up are used for these.

Anyone writing GSX applications with an eye to transferring them to IBM should choose the earliest driver they can find ... and you'll probably still have better facilities on the Apricot! Don't use the bit block moves and so on, or when you move to the IBM you will have to move that bit further to GEMVDI.

(Drivers for the F series and Portable pretty much duplicate the facilities of the PC.GSX family — every time an enhancement was made to one, the upgrade was applied right round the range. In principle, there are a whole series of these, involving the different screen options on each machine. It would be nice if someone could detail them at some stage.)

Some of the GSX drivers

(This is by no means a complete list ...)

File name		Date	Size	BIOS
Early systems:				
DDACRT.SYS		4-01-00	19968	1.4
		5-01-84	21632	2.3
Substantially revised:				
DDACRT.SYS	rel 1.5.2	11-07-84	16128	2.4
Later versions:				
PC.GSX	rel 1.6.2	07-02-85	19072	2.7
	rel 1.6.3	28-02-85	19968	
	rel 1.6.4	19-04-85	20096	
	[on DEALS FOR READERS disks]			

Hard copy

Hard copy from GSX graphics can be generated in five ways:

- You can produce a screen dump from the bit image stored in memory. There are screendumpers on the market or available through APRICOT FILE DEALS FOR READERS which will do the job
- Use the hard-copy facility which exists within the GSX screen driver
- Load up a GSX printer driver and re-execute the graphics instructions
- You can in principle generate a metafile, and import that into a graphics program for touching up and output from there — functionally equivalent to method 3
- Finally, under BIOS 3.1 you can load up the graphics system *before* GSX and then use the BIOS 3.1 screen copy utility. This is not a straightforward screen dump.

Simple screen dumpers all suffer from the same restriction: the screen is 800 dots across by 400 down, and the printer isn't. So the screen image loses its aspect ratio, or it isn't black enough, or something else intervenes to give you a poor result. Wide-carriage printers are generally necessary for best results.

The hard-copy utility within GSX is invoked by a specific function key (Shift-F6 or Shift-PRINT) and generates a screen dump on an Apricot Writer 10 be generated from within your own program by using the GSX 'generate hard copy' command. Escape sequences must be sent to set the printer to the correct dot spacing across the page, and the right linefeed spacing down the page first, which is a nuisance (escape codes for this are dealt with in one of the inset boxes).

A variation on this theme is the PCFX.GSX driver supplied on DEALS FOR READERS disk AP005, where the C Itoh bit image data is patched to make it compatible with the more common Epson printers.

The third option is generally my favourite. The printer driver can use every dot on the printed page, which may give significantly finer results than 800x400 — and for that matter, produce just as good a result when run on a Portable as on a PC.

The different screen drivers

	DDACRT	PC	Concurrent
Fonts	1	2	1
Character heights	4	400 and 1	1
in steps of	1147	82	
from	1147	82	1311
to	4588	32767	
Character rotations	4	variable and 4	4
Marker types	5	5	8
Marker heights	16	400	1
in steps of	164	82	
from	82	82	1311
to	2540	32767	
Line types	6	7	8
Line widths	5	1	1
Fill halftones	6	10	8
Fill hatchings	6	10	8
Primitives	1	4	1
Line widths	41	41	41

But the drivers are primitive and they support fewer fill/hatch patterns, line types, marker types and sizes and graphics primitives than do the screen drivers. What's worse, the text is pretty poor, it doesn't rotate nicely (only in 90° increments), and it can wrap around the page (*across* the page it wraps — *along* the page it actually prints out in full).

So why it is my favourite? Because you can alter the aspect ratio as you take that second bite at the cherry, with the effect that it is possible to draw extremely accurate diagrams.

To load hardcopy drivers, the later GSXBAS.COM is required (dated 31-12-86, length 4,750 bytes — it's on DEALS FOR READERS disk AP020). And be warned: unless APRICOT FILE has changed it, the early version supplied with Lindsay Doyle's GSX tutorials on DEALS FOR READERS disk AP005 can cause screen corruption after a hard copy driver is loaded. *(Yes, the* AP020 *version is now supplied on* AP005 *— Ed.)*

Similarly, with compiled Basic a revised interface object module is required; this is on a Black Hole disk too. Other compiled languages normally have a big enough stack not to cause problems.

The hardcopy drivers I have are dated 21-12-83 and are timed at just before 11pm. Was this a rush to get it done before the Christmas break? Are there later versions which work on an Apricot?

According to the documentation, the metafile option should be available under GSX just as it is under GEM. Just try and find a GSX metafile driver! No doubt there's something useful in the DR DRAW package — has anyone got a copy for the Apricot? A GSX metafile, just like a GEM metafile, consists of the contents of the five arrays copied into the file for each operation. That's why the length of each array is specified in CONTRL items — no point in using up excess space in the metafile, so only the bits of each array which is used are written.

Finally, using the BIOS 3.1 utilities 800400EG and SCRNCOPY (see David Langford's comments in APRICOT FILE Release 3.6, or go the whole hog and buy DEALS FOR READERS AP028), a superior screen dump can be obtained. I have no idea how it works: but it's significantly better than a straightforward screen dump, since you get a degree of choice in how wide the plot is, and whether to opt for a higher-grade plot, or one which maintains (as near as possible) the aspect ratio.

The C Itoh printer screen dump

ACT or Apricot Writers 10 and 12 are the C Itoh 8510 and 1550 series respectively. They were all supplied with parallel interface only, and without the factory-fitted optional download character RAM. Some of the later model 1550S were sold: these are significantly faster.

The 8510 is an 8in printer, the 1550 has a wider platen. In order to set the printer up for a screen dump, you need to set the inter-line spacing to 16/144th of an inch via the escape sequence ...

ESC T 1 6 [1B 54 31 36 hex]

... and you need to select unidirectional printing to make vertical lines straight:

ESC > [1B 3E hex]

In addition, you have to set the horizontal dot pitch. There are four choices, corresponding broadly to Pica (10 characters per inch or 80 dots per inch), Elite (12 cpi, 96 dpi), Compressed (17 cpi, 136 dpi) or Proportional (160 dpi). Ideally, on the 15in carriage machine you need the Elite spacing, for which the command is ...

ESC E [1B 69 hex]

That gives a 2:3 aspect ratio, the closest you can get to the Apricot's 3:4 without printing sideways. To centralise the picture on the page, issue a 'set left margin' command *after* the pitch setting (because it is specified in columns, and these are affected by pitch) to somewhere about column 36:

ESC L 0 3 6 [1B 4C 30 33 36 hex]

Personally, I'd set up the whole lot by using the public domain utility EPSON.COM from DEALS FOR READERS:

EPSON /1B5431361B3E1B691B4C303336/

... and keep it in a batch file!

If you have the narrower carriage printer, like me, you have two choices. You can use the next narrower character print (17 cpi) and centralise the picture which has an aspect ratio of about a little wider than 1:1. Or you can set 12 cpi pitch and force the last 32 dot positions to crumple up at the end of the line (you have to command this, otherwise the last 32 dot positions are printed out at the *beginning* of the line and therefore do not all run in together but appear at the right spacing so that the start of the picture is overprinted).

The narrower option requires the two escape codes ...

ESC Q [1B 51 hex]
ESC L 0 2 0 [1B 4C 30 32 30 hex]

This moves the start of the line to column 20 at 17 columns per inch. So the EPSON command would be ...

EPSON /1B5431361B3E1B511B4C303230/

The 12 cpi option requires the additional escape ...

ESC [[1B 5B hex]

... to yield

EPSON /1B5431361B3E1B691B5B/

Beware: the last few bit image bytes on each line are printed in a manner suggestive of the printer going beserk, and at a significantly slowed down rate!

(If you use the reverse backslash \ instead of the ordinary slash / in the command tail of EPSON, nothing is set. Pressing \ is easy enough to do if you are used to changing subdirectories, but EPSON does not give an error message).

When using SCRNCOPY, you can give a printer width. SCRNCOPY knows about the different print pitches, and will give you the next narrower pitch to whatever is implied by your requested screen dump width.

Be warned, though: the INT EC graphics extensions 800400EG, 640200EG, 640350EG and their ilk also come in a whole series of versions. The earliest do not support SCRNCOPY well, if at all ...

C Itoh and Epson printer drivers

	C Itoh	Epson Hi-Res	Epson Lo-Res
Dots across page	1088	960	480
Page width (ins)	8.00		
Dots down page	680	1368	672
Page length (ins)	9.44		
Fonts	1	1	1
Character heights	12	12	12
in steps of	386	192	391 NDC
from	386	192	391 NDC
to	4627	2300	4682 NDC
Character rotations	4	4	4
Marker heights	12	12	12
in steps of	386	192	391 NDC
from	386	192	391 NDC
to	4627	2300	4682 NDC
Marker types	5	5	5
Line types	6	6	6
Fill types	6	6	6
Primitives	1	1	1
Line width	31	35	69 NDC

The C Itoh and Epson printer drivers

The important factors to bear in mind here are those relating to character size. Fonts are scaled up by making one dot two by two, three by three, or whatever. As a result, large letters get very jagged.

Second, the physical size of the letters is governed not only by the aspect ratio of the page (ie one unit in NDC is different in x and y) but also the pixel to unit NDC ratio (because the font is generated with a pixel pattern).

Across-the-page text is the neater of the two. Neither matches what you see on the screen, and the fewer drawing primitives (no circles for instance), allied with the lesser number of halftones and hatching patterns, may make a screen dump preferable in some circumstances.

Further, text strings which overlap the page edge are not 'cropped' at the page margin. Across-the-page text wraps round and appears, overlapping, at the other side of the page. To avoid this, compute the length of the string in NDC units before calling GSX at the chosen character size, and crop it yourself. (The same effect is met with the stroke font in the screen driver.) The effect is serious with large characters, and easily met. Up-and-down-the-page text is drawn outside the basic plot area. This is inconvenient too, because it causes the important plot material to be out of alignment with the basic paper perforation pattern.

Plotting is quite slow. This is partly because of the driver's internal calculations (and so it's faster on a Xen), but mainly it's due to the data transmission problems — over 1,000 bytes per line at about 120 bytes per second for some 80 or more lines.

There is no advantage in having the wide-carriage 1550 when using GSX; the plot area is not enlarged. It is, however, advisable to offset the left margin as outlined above.

A major problem with the GSX driver is that it fails to set unidirectional print: this makes vertical rules come out ragged. To correct this, the Escape sequence ESC > has to be sent before the driver is invoked. ESC < restores bidirectional print. Alternatively, set DIP switch SW 2-8 to establish unidirectional print as the default.

Pretty much all of this applies to the Epson printer drivers, except that the escape sequences are different. The plural 'drivers' is used advisedly, because there are three! One gives double-pass, high-resolution copies; the others low resolution, but higher speed. One of the drivers operates with seven-bit data transmission, so you can leave your printer's DIP switches set to strip the high bit (hence 'unWordStaring' data) and still get a hard copy. The hi-res driver has a bigger choice of usable character and marker sizes — the total number is the same though.

Last words

Unless you *must* have lines of variable thickness, the best driver to go for is the latest PC.GSX. Hard copy via a screen dump gives you exactly what appeared on the screen — but at a distorted aspect ratio, so circles no longer appear round for instance. You can get round this by redrawing intentionally distorted before dumping. Hard copy with the GSX driver gives you just about the best your printer can achieve, but requires the re-draw anyway — at this stage, you need to be aware that the printer driver is more primitive.

Many users will have obsolete software which doesn't function properly, including GSXBAS.COM. So make sure that you have the latest of everything! ◆

Free Small Ads

Apricot Sales

■■■ APRICOT Xi — 768K RAM, 10MB hard disk, 720KB floppy, mono monitor. Plus manuals and lots of software. Like new — used very little. £700 ono. Call 0753 73232 ext 493 [office] or 0628 70504 [evenings].

■■■ APRICOT F2 — 512K RAM, twin 720K floppies, monitor, mouse; hardly used. GEM, word processor, spreadsheet, other software. Plus Apricot printer, all leads, and Apricot carry cases. Ideal for a small business or home use. £545 complete. Phone 0734 732512.

■■■ APRICOT PC 768K RAM, twin 720K floppies; regularly maintained. With MS-DOS 2.11, internal modem and Communiqué, WordPerfect, WordStar, SuperCalc 3, CardBox Plus and other software — and carry case, 37 disks, manuals, magazines, UserPrompt Guides. Price £750. Second machine also available. Call Richard Gaunt, 01-370 5946.

Other sales

■■■ APRICOT 9in MONITOR — new and unused. £35. Phone 0734 732512.

■■■ EPSON CUT SHEED FEEDER for LX80. Boxed, never used: £30 ono. Contact Terry TP Group on 01-833 3501.

■■■ ANSIBLE ACOLYTE (ver 1.4) disk and booklet containing 31 useful SuperWriter extras (I am now using WordPerfect). Price £30. Phone D Loebl, 01-642 5380.

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! Call Roger Gann at TP Group on 01-833 3501.

■■■ HELP for PC/Xi/Xen users — setting up systems, MS-DOS, directory organisation, comms, modems. ALSO utility and business software written (in C, assembler or Basic). Call Tony Bridges, 01-948 1603 any time.

■■■ REPAIRS guaranteed. Martin Filmer Computing Services. 01-761 0425.

■■■ A0 PLOTTING — time on a Calcomp A0 plotter available for rent: Sirius or Apricot disks plotted on your paper or ours. Terms negotiable, but cheap. Phone James McEvoy on 01-859 4014 or 01-852 7981; or write to 16 Dunvegan Road, London SE19 1SA.

■■■ 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 turnaround, guaranteed quality, low prices. You're reading an example of our work right now ... Call Roger Gann at TP Group on 01-833 3501.

■■■ HARDWARE AND SOFTWARE at bargain prices. Will beat any written quotation. Try us before you buy. 0382 66312 (day), 021-454 0898 (evening).

Wants

■■■ URGENTLY REQUIRED — F10 to join two others. Good price paid for a unit in good condition. Phone David Sparks on 0734 62787 or Telecom Gold 81:TWH167.

■■■ APRICOT PC with monitor and twin 720K drives wanted — would consider an Xi. Call James Gray, 01-839 6651 (day).

■■■ APRICOT PC TECHNICAL REFERENCE MANUAL wanted. Phone Simon Protheroe, 01-836 5454 ext 2161 (office hours) or 01-263 4489 (after 6pm).

■■■ APRICOT PORTABLE wanted, 512KB. Also SOFTWARE for FP and PC. Phone 0423 521439 or write E Holgate, 18 Montpellier Parade, Harrogate, HG1 2TG.

■■■ WANTED — Keypad for F2 (even a dead one will do). Robert Clews, Flat 6 Lindsworth School, Monyhull Hall Road, Kings Norton, Birmingham B30 3QA.

■■■ WANTED — dead/damaged/cheap Xis or PCs. Phone Derek on 01-650 4900.

■■■ APRICOT XEN (non-IBM-compatible) hard disk system. Also wanted — a good C INTERPRETER (to avoid having to link after every re-edit) or Apricot CODEVIEW debugger; and a TELETEXT TUNER/READER that can be connected to Apricots (is there such a thing?). Call Tony Bridges, 01-948 1603 any time.

■■■ F10 10MB hard disk system (with manuals if possible). Contact R Noakes, 051-336 6898.

■■■ IBM-COMPATIBILITY FOR F SERIES — either a fully-effective IBM emulator or a patch permitting the IBM version of Microsoft Word 3.0 to run. Also wanted — a WORLD MAP or GLOBAL MAPPING PROGRAM. I have scores of home-grown GW-BASIC programs to sell, exchange or share — including full documentation and sample programs for the infra-red Mouse. Contact Bob Mandel, Lewis & Clark College, Box 96, Portland, OR 97219, USA.

■■■ SECONDHAND OR EX-DEMO HARD DISK — internal or external, 10 or 20MB, for PC. Cannot afford £800+ for a new one. Write to T Orekoya, FAO (FIRI), Via della Terme di Caracalla, 00100 Rome, Italy.

■■■ 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.

■■■ 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 PC or 10MB XI wanted; also PRINTER if available. Fair price paid, immediate cash. Phone Mrs Morrison, 0636 704208.

Help

■■■ F SERIES user wants to get in touch with others in the States. Contact Bob Mandel, Lewis & Clark College, Box 96, Portland, OR 97219, USA.

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

Use the service!

The Classified Ads service is free to all readers of APRICOT FILE. Just let us have a sheet of paper with the words (not too many); your ad will appear at least once in the newsletter and for around three months in the DEALS FOR READERS broadsheet.

News for the Apricot World

Disk deal

New levels of reliability and durability are claimed by top diskette manufacturer **Maxell** for its new RD floppies. They aren't cheap, but they are good — and if you buy ten boxes from Action Computer Supplies you'll get an eleventh free. We've always found Action to be quick and competitive as a mail-order consumables supplier; call Freephone **0800 333 333** for its catalogue.

Newsbytes via Microlink

The on-line computer newsletter *Newsbytes* is now available in Britain via the **Microlink** on-line service that runs as a closed user group on Telecom Gold's System 72. *Newsbytes* is good: it carries weekly reports from seven correspondents (in the UK, Japan, Brussels, Canada and three US centres) that are short, sharp, readable and generally provide the best international coverage of computing we've come across from a single source. Formerly *Newsbytes* was most accessible to the likes of APRICOT FILE on The Source (via PSS to the States).

Microlink has just completed a major system upgrade — moving on to £250,000 worth of new Prime minicomputer, run by Telecom Gold but dedicated to Microlink's services and subscribers. And in March, within 18 months of start-up, Microlink's bulletin board — the UK's biggest and busiest — carried its 20,000th message.

We think Microlink is A Good Thing; it has all Telecom Gold's services and then some, and in use it seems to work out a bit cheaper than the straight Telecom Gold service. So we're switching our own mailbox on to Microlink — we don't yet have the number, but as from the end of this month our present 82:JET767 mailbox will no longer be operative. Anyone else interested in Microlink can call **0625 878888.**

... Gold grows

Meanwhile **Telecom Gold** itself celebrated its sixth birthday on 1 April (no sniggering at the back, there) by announcing that it now has **105,000** active users.

And there are four new databases — the full text of *Marketing Week*; the *Wall Street Journal* (updated daily, with articles appearing 72 hours after publication); **BIS Informat** (English-language summaries of marketing info from UK and overseas publications); and the Peat Marwick McLintock Grants database (guide to public and private-sector support in the UK — intended for development agencies, libraries, banks, local authorities, chambers of commerce, business and legal advisors). None of these require an additional registration, but the usual pricey connect-time surcharges (between £1 and £2 per minute) will apply.

Gold says several other databases ("selected from the best sources of information covering the main categories of news, companies and markets") will be added to its portfolio in the next few months. And users are also promised "significant developments" in terms of menu presentation.

Instant accounts

"Teach yourself financial skills in only a few hours" promises the blurb for Ivy Software's *Financial Accounting for Non-Accountants*. It's a computer-based tutorial (four disks plus booklet) intended to give an overview of business accounting leading on to some more detailed appreciation of P&L and balance sheets, and it has received some rave reviews from users. Amazingly, it's available in Apricot format too! The price is £50 including VAT; the company is on Freephone **0800 282 765.**

Share tips

The list of Britain's 200 fastest-growing companies published recently by *Business* magazine includes surprisingly few computer-related entries.

The mag looked at growth over the last three years among established firms with a turnover between £10 and £80 million; semiconductor distributor **Electron House** did best of the computer-related firms, followed by keyboard maker **Alphameric** (riding on the back of all those Big Bang terminal installations) and the mainframe maintenance outfit **DPCE.**

Then there are three of our own stock-market favourites in the distribution game — peripherals by **Northamber**, components and printers by **Norbain**, systems by **MBS.** The only other computer-industry entrants were consultants **Systems Designers** (lots of defence work) and terminal maker **Pericom** (another Big Bang success).

Of that lot, Northamber's stock exchange shares are getting near their highest 1987 price. But USM-quoted Norbain has a share price about as low as it's been in the last two years; worth a flutter? And we think MBS in particular has a lot of mileage left in it. In 1985 the group's headlong expansion turned a decent profit into a thumping loss, but heavy reorganisation during 1986 and 1987 turned the business round while revenues continued to grow at a regular and most impressive rate. Its valuation was clobbered in the October crash, but the balance sheet for 1987 looks very prudent — write-offs and provisions of over £2 million, operating profits for the year of £6.5 million on sales just over £29 million.

MBS is just about to embark on a £21 million rights issue, too. The shares are around 75p (and rising) as we go to press, having hit 175p last year.

It's also worth keeping an eye on **P&P Micro**, another of the big IBM-oriented distributors that is going public at the end of this month. P&P hopes to raise around **£12 million**; reportedly it expects to put at least some of the cash into acquisitions, not hitherto the P&P style. MBS is valued at around eight times earnings; P&P is looking for a P/E of over 12, with profits last year of £4.6 million on a turnover that almost doubled to £73 million. A far cry from the days when the original P&P, Pete and Pam, jumped on to the personal computer bandwagon in the late 1970s by flying to the States and filling their suitcases with anything they could buy ...

Bargain Miracle modems

Sydney Harrod of Apri-Soft-Exchange has come across a source of bargain-priced Miracle Technology modems. We called **Larry Fulcher** to discover that he used to work for Miracle, retains contacts within the firm, and has bought up all Miracle's stocks of out-of-production WS3000 and WS4000 modems.

These have the same performance spec as current products but come in now-redesigned packaging and use last year's casing and some components. But they are brand new or fully refurbished, upgraded to current specification where necessary, and are boxed with standard documentation — *and* you get a full guarantee from Larry Fulcher (part of Miracle's terms for dealing with him).

The prices are great: the V21/V23 **WS4000** for **£88**, **WS3000** from **£98** (that's a V21/V23 unit too, but Larry can do all versions up to the V22 *bis*). The WS4000 is half the price of Miracle's current version, the WS3000s are even better value — especially as there's no VAT on them, and those prices *include* P&P. A great deal. The only problem might be Apricot-specific RS-232 cables, but our DEALS FOR READERS service should be able to help out there if necessary. Larry Fulcher is on **0473 87539**.

Comms coming

The European market for **data comms** services will treble in the next five years to hit a value of **$55.5 billion** by 1992, according to a massive £7,500 report from consultants **Butler Cox** (no, we didn't buy it — they sent us a press release). Over the same period the market for *products* will only double, to hit $12.8 billion. The disparity in growth is attributed to a worsening skills shortage that will continue to constrain in-house development of networks, combined with the users' need to invest in network management rather than hardware to maintain competitiveness.

Micro sales

Back to the statistics for the UK microcomputer market we mention on page 1. Overall, unit sales rose by 32 per cent in the year; but because prices are falling, the UK business micro market grew by only 4 per cent in value. In terms of unit sales, Romtec says **Amstrad** was predictably enough the clear leader with over **100,000** shipments during the year — **26** per cent of the total. Because Amstrads are cheap, the company claimed only **7.5** per cent of the total value. IBM's **24.7** per cent carried its higher prices that gave it top place in terms of sales value — **31.8** per cent of 1987's total revenue. **Compaq**'s 386-based machines gave it second place, albeit a long way behind at **8.4** per cent (4.3 per cent of shipments); **Olivetti** was third with **7.9** per cent by value, about the same proportion for units.

Apricot had 4.9 per cent by value, just ahead of Tandon and Apple; both were ahead of Apricot for units sold, with the British company clocking 4.1 per cent of the total.

An **IDC** report covering the same ground gives both Amstrad and IBM a few more unit sales (**160,000** and **110,000** respectively), both well ahead of **Olivetti** with **37,000**.

Best Buys in computers

What to Buy for Business magazine has just published another of its mammoth round-up reviews of **IBM-compatible desktop computers**. Not surprisingly it doesn't include the new Dixons/Olivetti low-end Amstrad competitor (which looks good value to us) or our own OMEGA 286-based system (well, *WTBFB* has probably never heard of us). Best buy among PC/XT-style systems is the **Multitech Acer 500+** at **£889** — V20 processor, 65ms 32MB hard disk; everything else the magazine rated had an 80286 or 386 processor, with **Dell** scoring Best Buys in both categories. **Apricot** also got among the seven top-rated systems, with its big **Xen-i 386/50** tower (£4,298); and the rather more limited desktop **Xen-i 286** (£1,998) and **386** (£2,997) were both scored as 'Good Value'.

Incidentally, on *WTBFB*'s criteria it would appear that our OMEGA deal — 80286 processor, 20MB disk, price £1,475: lots of Apricot-upgrading extras — does compare well with the Dell System 200 with a basically similar spec and price.

UK subscriptions to *WTBFB* are £59.50 per year (ten issues). Details from **01-730 0403**.

Apricot's aims

Apricot is pushing hard for the 386 market now: MD **Ed Sherman** is quoted as saying the plan is to grab **35 per cent** of that business. To that end, Apricot and **Tetra Business Systems** have signed up as distributors of each other's products — Tetra gets the **VX9000**, Apricot will supply Tetra's well-regarded accounts packages.

Following the recent restructuring of its distribution, Apricot has also named three more elite-grade systems houses that will be given the VX9000 parallel-processing systems — **Genesis, Kestrel** and **McGuffie Brunton**. Up to 30 such specialists will be appointed in the next 12 months.

And another top rank has been created for Apricot's **desktop publishing** under the name 'Apricot DTP Centres of Excellence'; some 22 DTP dealers who demonstrate "above-average skills" in PAGEMAKER and Ventura Publisher will join a club that meets regularly to discuss problems and applications. The aim here is obviously to pool expertise and experience in selling to big public-sector clients. ◆

Bargain bits

Apricot fans looking for **bargain-price** used and liquidated hardware and software in London could do worse than try **Morgan Computer Co** in the Tottenham Court Road (**01-636 1138**). The last time we dropped in, Morgan had **128K cards** for **£25**, 9in mono **monitors** at **£35**, and a bunch of packages including **QED+** for **£25** and brand-new **Perfect** series software at **£20**.