UCSD Pascal on the Macintosh

A D N Edwards

&

M Woodman

Computing Discipline
Mathematics Faculty
Open University
Walton Hall
Milton Keynes
England
MK7 6AA

1.0 INTRODUCTION

For the Pascal programmer there are three implementations for the Apple Macintosh currently available in Europe. Two of the implementations are derived from UCSD Pascal. Apple's own implementation, MacPascal, is not a UCSD derivative, but is close to the ANSI standard. It is unusual in that it is **interpretive**; programs are not compiled, but translated directly on execution. As such, it represents an interesting tool for teaching programming, but is not a suitable vehicle for developing real Macintosh applications.

The major difference between the two UCSD versions is the degree of access they provide to the Macintosh's special features. The Macintosh is a remarkable microcomputer in that it is the first one cheaply available with a user interface which includes windows, icons and pull-down menus (sometimes known as a WIMP interface). The UCSD implementation known as MacAdvantage provides fairly complete access to these facilities, whereas the other version - which we will refer to as the Mac p-System - has only a very limited access. (As does, incidently, MacPascal).

Both UCSD systems were implemented by SofTech Microsystems and are available in the UK from TDI of Bristol.

The basic Macintosh machine has 128K of user RAM and a built-in 3 1/4 inch floppy disc drive. However, to run either UCSD implementation you will need a machine upgraded to 512K RAM (in order to get reasonable execution speed) and an extra floppy disc drive.

2.0 MACADVANTAGE

If you have a Macintosh you will undoubtedly want to make the most of its special features. MacAdvantage is the only Macintosh Pascal implementation currently available outside the USA which allows you to do this. Be warned, however, that before you will be able to use the facilities you will need further documentation on them, specifically the manual 'Inside Macintosh'. This is available from Apple - but it will cost you around one hundred pounds. It comes in two large volumes - and is almost less portable than the computer itself! (There may shortly be a more portable, book version of this available, price as yet unknown). You will also really want 512K of memory, not only for the performance reasons mentioned above, but also because system overheads can mean that with 128K you may actually get less than 64K data space for your application. (See also section 2.9 below on memory restrictions). As stated above, to run MacAdvantage an extra, external disc drive is required. In fact, when developing a large program two discs barely provide enough space and files must be archived off to other discs.

The Pascal compiler generates UCSD p-code which is interpreted at runtime by a p-Machine emulator which is an executable Macintosh application. In addition, a file known as the Pascal Runtime is required for program execution. This file contains assembly language routines which support UCSD Pascal programs on the Macintosh.

The Pascal implemented is fairly standard UCSD except for a few omissions and the extensions mentioned in section 2.9. The major omission is that the unit I/O intrinsics (unitread, unitwrite, unitstatus etc.) are not supported. MacAdvantage also supports some other features of programs which are specific to the Macintosh: objects such as menus, strings, windows and control icons can be defined as "resources" which are handled in a file separate from the program code. Whether this is as desirable as Apple seem to think is debatable; it might be said that the definition of all data should not be separated from the program code. The principle behind separating the resources is that their design can be altered without recompiling the program, using a resource editor program - which is not yet generally

available. The motivation for this appears to be that it is thus easier to produce different versions of a program oriented to different natural languages.

A useful feature of MacAdvantage is the ability to open a special debug file. This enables the program to write to the bottom few lines of the screen, regardless of what else is happening on it. This is handy because it would be difficult to get a faulty program which manipulates windows to correctly write debugging information into a window.

MacAdvantage provides a set of tools which run under the Macintosh's Finder operating system. Each of these tools is described below.

2.1 Editor

The editor is not the UCSD screen editor but a fully mouse-oriented one which works much like most Macintosh applications. It allows the user to open more than one file at a time which is useful for cutting and pasting between programs. It has quite powerful commands for finding and replacing strings, although the commands have the annoying feature that if they find no further occurrence of the target string they reset the cursor back to the beginning of the file. This is frustrating, for example, when globally replacing text since you often do not know whether you have changed the last occurrence of a string. Also the find and replace commands will only work forwards through the file.

The released version of the editor does contain some intermittent bugs. Cutting and pasting large amounts of text do not always work properly and sometimes the screen does not accurately reflect the contents of the file.

2.2 Resource maker

There is a language for defining resources. To incorporate resources into a program a text file must be created in the resource maker language which is then compiled into a resource file using the resource maker. For instance, a window might be defined in the resource file as below. It will be identified by the number 100, have the title 'A window', its initial corner coordinates will be (50, 40) and (300, 450), it will be visible on the screen and it will have a 'go-away' control box.

```
TYPE WIND
  , 1000
A window
50 40 300 450
Visible GoAway
0
0
```

This will not be necessary in future when a resource editor program will be available.

2.3 Compiler

The compiler user interface appears quite conventional. To some extent that is a shame, since it relies on the usual interaction of typing in file names - instead of selecting from menus as is normal on the Macintosh. One extra file name which must be entered is that of the resource file for your program. The compiler is not particularly fast: a program which consists of around 2500 typed lines, plus 1500 lines of imported units, takes around 5 minutes to compile. This seems to be largely due to the slowness of disc access on the system, the discs being connected by fast serial lines. If an error occurs during compilation you only have the options of continuing or quitting; there is no interface to the editor which will allow the user to go into the editor at the point of the error, as in the p-System.

The version currently being distributed has an unfortunate bug which means that a compiler listing is generated even if the programmer did not request one (by hitting carriage return in response to the "File for listing" prompt). This clogs up the discs - which have little enough space for developing large programs as it is. What is more, it can be very annoying indeed when the compilation of a syntactically correct program fails during its last phase for lack of disc space for a listing - an unwanted one at that. The preparation of a compiler listing also slows down the compilation.

2.4 Librarian

This appears to be quite standard.

2.5 Set options

There are a number of options which must be selected for a code file. These are such things as what libraries are used, the name of the file containing the p-Machine Emulator and Pascal Runtime and whether the program will use a default window or create its own. The set options tool allows you to do this quite easily.

2.6 Debugger

Sadly when most language implementors write debugging tools they still expect programmers, who write in high-level languages, to think in machine language - or p-code. The same it true with this system; there is a debugger, but it is strictly for p-code freaks.

2.7 Executive

This is an operating system interface which is provided which is a cross between the Macintosh Finder (the familiar Macintosh 'desk top') and a UCSD-style interface; it has a set of pull-down menus, but does not provide all the file manipulation facilities of the Finder. It is supposed to be faster than the Finder, but the actual improvement in speed is negligible.

2.8 Documentation

The system comes with two manuals - the standard Clark and Koehler UCSD Pascal Handbook and one on MacAdvantage itself. The latter is one of those manuals which has most of the MacAdvantage-specific information you want - somewhere - but it is not always easy to ascertain it. It contains language and internal architecture details as well as a description of the unit interfaces to the Macintosh Toolbox. The most useful sections are the description of the extended addressing operations, a sample program and the listing of the Toolbox unit interfaces, although there are some errors in that interface listing. Furthermore, the unit interfaces to the Toolbox only define the syntax for using the facilities. The reader must refer to the 'Inside Macintosh' manual for details of the semantics. Indeed, frequent references are made to 'Inside Macintosh', which demonstrates that the user really needs to be quite conversant with the Macintosh system.

A problem with the manual, as supplied by TDI, is that its binding is too flimsy and will not stand up to the frequent use it inevitably receives.

2.9 Accessing Macintosh facilities - address restrictions

The Macintosh's system facilities are accessed by calling procedures and functions - known as the Toolbox - which are stored in ROM. These can be included in MacAdvantage programs through the usual UCSD Pascal Unit mechanism.

Unfortunately using system routines is not as straightforward as it might seem. There is a fundamental restriction of the p-machine in that it is strictly confined to sixteen-bit addresses, and thus to a 64K data space. The Macintosh has a much larger address space - at least 128K plus all the Toolbox ROM. In order to access the system data in the wider address space, MacAdvantage Pascal has been extended to allow restricted use of 32-bit addresses. The situation is further complicated by the Macintosh's memory management. Memory allocation is highly dynamic. This means that doubly indirect pointers (known as "handles") are used to address movable memory blocks.

Although the extended addressing facilities provided are sufficient for most applications, they are rather clumsy. To inspect a system variable - in a location outside the p-System address space - it is necessary to execute a call which will copy the data into a variable within the p-System address space and then look at it. Similarly to change a system variable the value must be set up locally and then copied into the appropriate system location. This copying is obviously rather slower than direct addressing, especially since it is usually necessary to copy more data than you actually need (when using one field within a record, offset from the record's base).

The implementation of the Toolbox interface relies on relaxed type checking across the interface and in fact it becomes necessary quite often to resort to such a 'tricky' style of programming if one wants to do more advanced operations.

3.0 THE MAC P-SYSTEM

This implementation is really a straight adaptation of the p-System. It includes a limited ability to call Macintosh drawing ('Quickdraw') routines. This gives facilities to draw items such as lines, rectangles and circles. The system comes as three Macintosh files, one of which is executed under the Finder to boot the p-System. Thereafter everything runs as a normal p-System.

Since it has no special merits over other p-Systems and cannot use most of the Macintosh's facilities, its likely use would seem to be for people who have existing UCSD programs they wish to run on a Macintosh. It will do that successfully, importing either source or compiled p-code, but there is a problem in getting the imported code onto the system. The p-System file system is not compatible with the Macintosh Finder's. As far as the Finder is concerned a p-System disc volume is just a single file. It is not possible to implant a Macintosh file into a p-System volume. Thus, the only route for importing code is to load it via the serial port. (Remtalk is available if you can persuade it to talk to the Remtalk on the machine from which you are importing).

The lack of any file interface to the Finder operating system is a very serious omission and significantly reduces the usefulness of the Mac p-System. Units for manipulating Macintosh files from UCSD Pascal should have been provided as well as a Macintosh Filer. Both these features are to be found in the MS-DOS-hosted p-System.

We also had no success in configuring the system to run on a Macintosh RAM disk. The RAM disk was too small to hold the p-System boot volume file and we were unable to create a smaller boot file. It was not clear whether our problems lay with the Macintosh RAM disk or the Mac p-System - but the experience did nothing to enhance our view of the system.

The Fortran-77 compiler is available on the Mac p-System as well as UCSD Pascal.

3.1 Documentation

A set of p-System manuals is supplied with an additional small document on using the p-System on the Macintosh: how to boot it, how to make backups, non-standard features etc.

4. [illegible]

The Macintosh is perhaps the first available cheap member of the new generation of computers, based on a window, icon, mouse and pull-down menu user interface. There is a valid argument that the last generation of programming languages are not suited to handling this kind of program. Implementors of these new systems ought to look at object-oriented languages provided by systems such as Smalltalk, not just the attractive graphics they provide. However, given that programmers will want - and may be forced - to use the old languages, how useful are these particular implementations?

The Mac p-System is the best vehicle for moving p-System software developed on another machine onto a Macintosh. However, the restricted access to the Macintosh Toolbox means that there is probably little reason to want to transfer such software - why not continue to run it on whatever machine on which it was developed? Moreover, the lack of an interface to the Macintosh file system not only makes file transfer to the Mac p-System difficult but also removes the option of progressing to Macadvantage once the original UCSD Pascal code is up and running. If this were possible it would be a useful route through which existing UCSD Pascal software could be ported onto the Macintosh and then enhanced to make use of Macintosh facilities.

Thus, MacAdvantage is the best chance of getting into the Macintosh and writing programs which really use its capabilities. However, it too is limited. Building applications of any real level of ambition foces the programmer into using unsound programming techniques. Thus it is far from being an ideal language for creating realistic Macintosh applications. Nevertheless, it is the only Pascal implementation currently available outside the USA which allows you to do this at all.