TECHNICAL QUERIES

Alan Wrigley answers queries on processor flags, deleting icons, the font cache, and OPI and OLE

Q Dear Sir
I have noticed some SYS calls that have the form:

```
SYS "TypicalCall",r0,r1 TO r0;flags
```

Could you please explain what the flags represent?

G H Keathley

A The ARM processor has a number of flags built into it which indicate certain conditions. For example, the zero flag (Z) indicates that the last operation resulted in a value of zero, the negative flag (N) indicates that the result was negative (i.e. the top bit was set), and the carry flag (C) indicates that a carry occurred. ARM code programs often need to know the status of these flags, since all conditional instructions depend on them. For example, the suffix NE when added to an ARM instruction means 'perform this operation only if not equal to zero', i.e. if the Z flag is not set.

When making SYS calls from Basic, you can find out the status of the flags on exit from the call by adding a semi-colon after the list of register values to be returned, followed by a variable into which the status register value will be placed. If you only want the flags and no registers, the semi-colon comes immediately after the keyword TO. Most of these flags are irrelevant to Basic programmers since Basic has its own conditional logic. However, there is one flag that may be quite important and that is the overflow flag. This is set on return from a SWI if an error has occurred. The overflow flag is conveniently bit 0 of the status register.

To force a SWI to return an error to the calling program, you must use the X form of the SWI. Having done so, you can then read the overflow flag, and if it is set, generate your own error. A good example is illustrated by the following:

```
  SYS "XOS_SWINumber From String",swina
 me$ TO swinum%;flags%
  IF flags % AND 1 THEN ERROR 1,"Never
 heard of that SWI"
```

where swiname$ should hold a SWI name on entry. If the SWI is not known then the overflow flag will be set. This is picked up in the second line and an error generated.

Q Dear Sir
I wish to alter the shape of an icon while my program is running, but there appears to be no way apart from using 'Wimp_DeleteIcon', and then 'Wimp_CreateIcon' to create it again. Is this correct or have I missed something?

Brian Durne

A If you are lucky enough to have a Risc PC, there is a new SWI call in RISC OS 3.5 which will do the job for you. SWI 'Wimp_ResizeIcon' takes the following parameters:

R0	window handle
R1	icon handle
R2-R5	new bounding box

This can also be used to move an icon to another part of the window.

For lesser mortals, you do indeed have to delete an icon if you want to alter its size, although there is an alternative solution to the one you mention which is neater and rather safer to use (provided you don't use it on writables). If you create two icons instead of one, you can swap between the two by setting and resetting the deleted bit of the icon flags (bit 23) using 'Wimp_SetIconState'. Setting this bit removes the icon from the screen but does not remove its handle and parameters from the list. This avoids the possibility that recreating the icon might result in a different handle from before. It also gets around the problem that deleting and recreating an icon doesn't in itself redraw the screen, since you can at the same time set the 'needs help to redraw' bit (bit 7). This ensures that the Wimp redraws the area where the icon was after removing it.

So to remove an icon you would use:

```
bitset%=(1<<23)+(1<<7)
!block%=whandle:block%!4=ihandle%
block%!8=bitset%:block%!12=bitset%
SYS "Wimp_SetIconState",,block%
```

and to reinstate it:

```
!block%=whandle:block%!4=ihandle%
block%!8=0:block%!12=bitset%
SYS "Wimp_SetIconState",,block%
```

By performing the opposite action on the second icon, you can make it appear as though a single icon is changing shape.

Q Dear Sir
I bought my Acorn mainly because I was impressed with the speed at which information was displayed on the screen. I now find, particularly with DTP work, that it seems to take much longer to write text to the screen than it did when the computer was demonstrated to me, and each time I move down a page the whole laborious process starts again. I feel somewhat cheated as it appears the dealer was using a souped-up machine in order to make a sale.

Emma Copeland

A You can rest assured that you haven t been cheated. One of the strengths of an Acorn computer is that the machine can be optimised for a particular working situation, by reshuffling the memory which is allocated to various functions. What happens when text is displayed in an outline font is that the font manager has to look up the relevant data for each character as it draws it on the screen. This data is contained in a file on disc. As you can imagine, if the disc has to be accessed for each character it would take forever and a day to display a page of text, and so a section of memory is set aside as a font cache. When the font manager encounters a new font or point size, it attempts to retrieve the data for every character in that font and size and place it in the font cache. From then on, it only needs to access memory to find any of those characters, making the process virtually instantaneous.

If your document then uses a different font or a different weight or size, the new data will be added to the old in the font cache. Clearly this can t go on for ever as eventually you would have no memory left for anything else, so the font cache is normally set to a sensible size, and the font manager discards old data if there is no more room for the new.

The size of the font cache can be configured by the user, so that you can use a setting which suits your own needs. If you rarely use outline fonts, and then only in one or two variants, you can get away with a smaller font cache than if you do a lot of DTP work and use many different fonts. It is impossible to be specific about how much memory to allow for the cache, as it depends on the amount in your machine in the first place, but if you can spare 256K this should cope with many situations.

An individual font may require from 10K per weight and size to over 40K. You can get an idea of how your font cache is being used by typing:

```
FontList
```

at the command line.

The default size for the font cache can be set using the Configure command, as follows:

```
Configure FontSize nK
```

where n is the required size in kilobytes. For RISC OS 3 you can include the following line in your desktop boot file:

```
ChangeDynamicArea -FontSize nK
```

while on a Risc PC, you can double-click on !Boot, click on the memory icon, then set the font cache size in the dedicated window.

The cache size can also be set dynamically as you work on all machines by dragging the slider bar in the Task Manager window.

Q Dear Sir
I have seen references to software which supports OPI , and other software supporting OLE . Are these variants of the same thing, or what? I see no evidence at all that computing is getting less jargon-ridden!

Geoffrey O Connor

A The two things are not the same at all (and I agree wholeheartedly about jargon). OPI stands for Open Pre-press Interface, and is a system which enables you to manipulate large images in DTP work without having to handle huge files. If you are preparing material to be printed, and you need images to be scanned to professional quality standards, you would normally take these to an imagesetting bureau to be scanned rather than do the job yourself on a cheap scanner. If the software you are using supports OPI, the bureau can then give you low-resolution copies of the images which you can manipulate in your document, while keeping the high-resolution versions at the bureau. When you then take your document files to the bureau to be processed, the hi-res copies are automatically substituted before the plates are made. CC s Publisher Plus supports OPI.

OLE is something completely different, as they say, and stands for Object Linking and Embedding. It enables you to edit an element of a document (a picture, say) using the relevant editor, and have the copy of the picture in the document updated automatically. For example, suppose you are working on a DTP document containing a sprite. With OLE, you can edit the sprite in