C Notebook

David Spencer rounds up some more C hints

Icon Bar Sprite Pools

Andrea Gallo

Although it is not documented in the User Guide, it is possible to place sprites from the Wimp's sprite pool onto the icon bar using calls such as baricon(). All that is needed is to give a sprite area pointer as 1, rather than the actual address of a sprite area.

File Handling

David Spencer

Whilst using the file manipulation operations provided by stdio.h ensures that your code will be as portable as possible, these functions are relatively slow, and inflexible. For serious desktop applications (which are very unlikely to be ported anyway) you are better off using the calls provided by kernel.h to interface with RISC OS directly.

kernel swis and their Errors

Lee Calcraft

The approved way of performing swi calls is to make use of the kernel library (in Clib) rather than the Risc_OSLib os library. There are no handy packaged functions for making swi calls in kernel, but it is easy to use something like the following, which takes just four parameters - the swi number and a pointer to the values to be placed in r0, r1 and r2. The return values are then written directly into the three variables used by the calling function.

```
_kernel_oserror *swi_3(int swi_no,int *r0,int *r1,int *r2)
{
    _kernel_swi_regs r;
    _kernel_oserror *e;

    r.r[0]=*r0;
    r.r[1]=*r1;
    r.r[2]=*r2;
    if ((e=_kernel_swi(swi_no,&r,&r))==0)
    {
     *r0=r.r[0];
     *r1=r.r[1];
     *r2=r.r[2];
    }
    return e;
}
```

This is all pretty straighforward, but there is a problem when you come to interface it to the Wimp's error handling functions. For example, you cannot use:

```
wimpt_complain(swi_3(swi_no,&r0,&r1,&r2));
```

because wimpt_complain() is expecting a pointer to an os_error, while our function returns a _kernel_oserror pointer. In fact, these two structures are identical, and all you need to do is to cast one pointer to the other:

```
wimpt_complain((os_error*)\
  swi_3(swi_no,&r0,&r1,&r2));
```

Handling Shutdowns

Andrea Gallo

A quirk of RISC OS is that if an application objects to the computer being shutdown by claiming the PreQuit message when it is received, then it must later restart the shutdown procedure by simulating Shift-Control-F12 being pressed. The PRM suggests doing this by sending a Wimp message, but by far the easiest way to do it in C is to issue the call:

```
wimpt_complain(wimp_processkey(0x1fc));
```

Array Sizing

Lee Calcraft

The sizeof operator can be very handy for determining the size of an array that you have filled at the time of declaration. For example, suppose you write the following:

```
int my_array[]={199,2,44,992,790...etc};
```

If you are using a #defined value MAX_SIZE to hold the maximum size of the array, it is only too easy to increase the number of entries at a later date without adjusting MAX_SIZE.

However, you can automate the process by defining MAX_SIZE as follows (on any line after you have declared the array):

```
#define MAX_SIZE sizeof(my_array)/sizeof(int)
```

Using Assembler in C

David Spencer

If you are using the ObjAsm assembler to code functions in assembler, then it is absolutely vital that they obey the ARM Procedure Call Standard (APCS) which is documented in the Assembler User Guide. If you don't then weird and wonderful bugs caused by unexpected register corruption can result. Similarly, the stack limit checking mentioned in the APCS documentation should also be included to ensure robust code.

Please send us your C hints, and technical