

# PocketStation Application Shell

The purpose of the application shell is to allow rapid development of PocketStation applications without the programmer having to worry too much about the boot process or commonly required features. It is intended to be used with the SCEE sample programs, the routines from which should all be compatible with the shell and easy to slot in.

## **Boot routines**

Relevant files: BOOT.S (ARM), MAIN.C (Thumb)

The boot routines are kept in the BOOT.S ARM assembler module. The startup code sets the system clock to 4MHz, and installs the interrupt service routines *IRQ\_Callback* and *FIQ\_Callback*. The *IRQ\_Callback* routine services the button interrupts directly by updating the *keybuff* variable from the keyscan module (see later). Other interrupts are handled by calling the service routines specified in the jump table beginning at *IRQ\_Branch*. These default to the *IRQ\_Callback\_\** and *FIQ\_Callback\_\** routines provided in the callback module

Once all the interrupts are initialised, user SRAM is cleared to 0x00, the stack is set to the top of SRAM and the function *mymain()* is called. By default this is a Thumb C function in MAIN.C. Take care not to name the function *main()* because the linker gets upset when there is a function called *main()* which is not at the ENTRY point.

When *mymain()* exits, control passes back to BOOT.S which returns control to the PocketStation OSD at the correct point.

## **Callback routines**

Relevant files: BOOT.S (ARM), CALLBACK.S (ARM)

The application shell divides the interrupts by source and calls the corresponding user callback functions, stubs for which are provided in CALLBACK.S. It is recommended that you write the callback functions in assembler for speed, especially when the processor clock is set to a low frequency. It is possible to write these in ARM C by removing the stub and EXPORT directive from CALLBACK.S. BOOT.S contains the jump table for the callback functions beginning at the label *IRQ\_Branch*.

## **Key scanning routines**

Relevant files: BOOT.S (ARM), CALLBACK.S (ARM), KEYS.C (Thumb)

Because the buttons generate level triggered interrupts, it is important to install a callback function to ensure that no button presses are missed, especially when polling infrequently. In conjunction with BOOT.S, KEYS.C provides a general-purpose button abstraction which allows the main program to poll the buttons at will. Use the *initkeys()* macro to initialise the keys, clearing all button presses which have occurred and starting a timer.

During normal program execution, when the user presses a button, the *IRQ\_Callback* in BOOT.S will be called and the variable *keybuff* will be updated with the buttons which have been pressed. The function *initkeys()* sets up counter 0 to interrupt about every 1/32 second. The callback routine for counter 0 copies the contents of the global keys. So keys always

contains a bitmap of the buttons which have been pressed in the last 1/32 second slot. This latency will be acceptable for a PocketStation program. The global `entertime` contains the time that the enter button has been held down, in units of 1/32 second. This is for the {Continue,Exit} menu.

As a side-effect of using a counter to scan the keys, the programmer must take care when changing the CPU frequency. `stopkeys()` must be called before changing the frequency, then `startkeys()` after to update the counter reload value.

### ***{Continue,Exit} menu***

Relevant files: MENU.C (Thumb), MENUSND.O

Call `exitmenu()` to use a TRC compliant exit menu. The return value is 0 for continue, or 1 for exit. During execution of the menu the CPU is switched to 32KHz and the Display to 32Hz. Sections 1 and 4 of the display are turned off. On exit the sound will be switched off but the display and CPU frequency are fully restored. Be sure to link MENUSND.O which is called to make the beep.