ARM System Calls

`SWI_WriteC (SWI 0)`

Write a byte, passed in register 0, to the debug channel. When executed under the symbolic debugger, the character will appear on the display device connected to the debugger.

`SWI_Write0 (SWI 2)`

Write the null-terminated string, pointed to by register 0, to the debug channel. When executed under the symbolic debugger, the characters will appear on the display device connected to the debugger.

`SWI_ReadC (SWI 4)`

Read a byte from the debug channel, returning it in register 0. The read is notionally from the keyboard attached to the debugger.

`SWI_Exit (SWI 0x11)`

Halt emulation. This is the way a program exits cleanly, returning control to the debugger.

`SWI_Clock (SWI 0x61)`

Return, in r0, the number of centi-seconds since the support code began execution. In general, only the difference between successive calls to `SWI_Clock`, can be meaningful.

`SWI_Open (SWI 0x66)`

r0 addresses a NUL-terminated string containing a file or device name; r1 is a small integer specifying the file-opening mode: `0` - read mode, `4` - write mode, `8` - apend mode. If the open succeeds, a non-zero handle is returned in r0, which can be quoted to `SWI_Close, SWI_Read, SWI_Write, SWI_Seek, SWI_Flen` and `SWI_IsTTY.` Nothing else may be asserted about the value of the handle. If the open fails, the value 0 is returned in r0.

`SWI_Close (SWI 0x68)`

On entry, r0 must be a handle for an open file, previously returned by `SWI_Open`. If the close succeeds, zero is returned in r0; otherwise, a non-zero value is returned.

`SWI_Write (SWI 0x69)`

On entry, r0 must contain a handle for a previously opened file; r1 points to a buffer in the callee; and r2 contains the number of bytes to be written from the buffer to the file. `SWI_Write` returns, in r0, the number of bytes not written (and so indicates success with a zero return value).

`SWI_Read (SWI 0x6a)`

On entry, r0 must contain a handle for a previously opened file or device; r1 points to a buffer in the callee; and r2 contains the number of bytes to be read from the file into the buffer. `SWI_Read` returns, in r0, the number of bytes not read, and so indicates the success of a read from a file with a zero return value. If the handle is for an interactive device (`SWI_IsTTY` returns non-zero for this handle), then a non-zero return from `SWI_Read` indicates that the line read did not fill the buffer.

`SWI_Seek (SWI 0x6b)`

On entry, r0 must contain a handle for a seekable file object, and r1 the absolute byte position to be sought to. If the request can be honoured then `SWI_Seek` returns 0 in 0; otherwise it returns a host-specific non-zero value. Note that the effect of seeking outside of the current extent of the file object is undefined.

`SWI_Flen (SWI 0x6c)`

On entry, r0 contains a handle for a previously opened, seekable file object. `SWI_Flen` returns, in r0, the current length of the file object, otherwise it returns -1. `SWI_IsTTY (SWI 0x6e)` On entry, r0 must contain a handle for a previously opened file or device object. On exit, r0 contains 1 if the handle identifies an interactive device; otherwise r0 contains 0.

```
;
; An example to write a short file on disk
;
        AREA  Example, CODE, READONLY          ; name this block of code
SWI_Exit    EQU   0x11          ; tidy finish
SWI_Clock   EQU   0x61
SWI_Open    EQU   0x66
SWI_Close   EQU   0x68
SWI_Write   EQU   0x69
write_only  EQU   4             ; mode 4 = open to write
        ENTRY                               ; mark first instruction
                                ; to execute
start   ADR   r0, filename      ; r0 points to string
        MOV   r1, #write_only
        SWI   SWI_Open          ; open a file for writing
        MOV   r5, r0            ; save file-handler in r5
        ADR   r1, String        ; point to a string
        MOV   r2, #14           ;   ..... 14 characters long
        SWI   SWI_Write         ; write to file
        MOV   r0, r5
        SWI   SWI_Close         ; close the file
        SWI   SWI_Exit

filename = "test.txt",0
String = "Hello World!",&0a,&0d

        END
```