LibFalcon

A simple C/C++ library for linking into hobby OS kernels.

Syed Nasim, 7 June 2016.

Introduction to LibFalcon

LibFalcon is a simple (and incomplete) C/C++ library to be linked with Hobby Operating System Kernels. Almost all essential functions required by kernels are provided. (memcpy, strlen, atoi, etc.) However, there are No math functions at this moment. The reason is that I am in 9th standard, and I do not have enough knowledge about math functions. LibFalcon is distributed under the BSD license; (which is more permissive than GPL and friends.) see the accompanying LICENSE file in LibFalcon source directory for more information.

- ***Can I use LibFalcon for commercial and Closed-Source projects?***
- *Yes, as long as you mention that your project uses LibFalcon. Use LibFalcon under the terms of the BSD license.*

Features of LibFalcon

Apart from standard library functions, LibFalcon also has functions for low-level port I/O and reading from/writing to control registers. In later versions, more and more functions would make use of inline assembly and optimization measures, to make the code faster. LibFalcon also uses Durand Miller's Liballoc for kernel heap functions.

Installation

For Installation of LibFalcon You should have:

1. a working copy of GCC. (4.8.0+ recommended.)
2. NASM. (tested with NASM version 2.09.10)
3. GNU make.

Go to the LibFalcon Project root directory and run `make clean`, just to be sure there are no stale object files. After that run `make all`. If everything goes well and LibFalcon Compiles correctly, The build system will print "Build Successful." A file named `libfalcon.a` should be located on the Project root directory. The library is now almost ready to be linked with your kernel.

Now you need to provide some functions yourself in your kernel, as LibFalcon relies on them. Without these 'hook functions', you would not be able to link the library with your kernel. (As of now, all of these functions are required by liballoc, which is shipped along with LibFalcon. If you do not wish to use liballoc functions, just implement the hooks and leave them empty.) The functions are:

int liballoc_lock()	This function is supposed to lock the memory data structures. It could be as simple as disabling interrupts or acquiring a spinlock. It's up to you to decide. return 0 if the lock was acquired successfully. Anything else is failure.
int liballoc_unlock()	This function unlocks what was previously locked by the liballoc_lock function. If it disabled interrupts, it enables interrupts. If it had acquired a spinlock, it releases the spinlock. etc. return 0 if the lock was successfully released.
void* liballoc_alloc(int)	This is the hook into the local system which allocates pages. It accepts an integer parameter which is the number of pages required. The page size was set up in the liballoc_init function. return NULL if the pages were not allocated, or return a pointer to the allocated memory.
int liballoc_free(void*, int)	This frees previously allocated memory. The void* parameter passed to the function is the exact same value returned from a previous liballoc_alloc call. The integer value is the number of pages to free. return 0 if the memory was successfully freed.

Using LibFalcon

You can link the library with your kernel by running something like:

gcc -o <your kernel's name> <Compiler Flags> <kernel objects> libfalcon.a

NOTE: In your kernel, you should always use header files found in include/ directory for your kernel code.

Complete list of LibFalcon functions

Here is the complete list of all LibFalcon functions. More coming soon!

isalnum	ispunct	inportw	calloc	strcat	strlen	strspn
isalpha	isspace	outportb	free	strchr	strncat	strstr
iscntrl	isupper	outportw	memchr	strcmp	strncmp	strtok
isdigit	isxdigit	reboot	memcmp	strcoll	strncpy	atol
isgraph	tolower	get_cpuvendor	memcpy	strcpy	strnlen	atoi
islower	toupper	malloc	memmove	strcspn	strpbrk	
isprint32_t	inportb	realloc	memset	strerror	strrchr	

Help Make LibFalcon Better!

LibFalcon is hosted on GitHub. (here's the link) If you think some functions can be optimized, or some essential functions need to be added for the aid of Kernel Developers, mail me with a patch. (Don't make pull requests on GitHub.) The subject of the mail should be “PATCH FOR LIBFALCON”, or it may be go un noticed.