

## A Modified Spinapi USB library for OSX, update March 10, 2011

Included here you will find modified spinapi source code (incorporating changes made up to December 2011; spinapi\_source\_0120310; there are not many substantive changes to the current spinapi code since then, see the changelog for details), a functioning dynamic library, and an Xcode Tools project file. These instructions describe how to take the included, modified spinapi source code, and compile and link it into a dynamic library (dylib) under Apple OSX for use with RadioProcessor and PulseBlaster USB boards. After this is done, you will have the following capabilities:

- You should be able to compile, link, and use any pulse programs that use USB Radioprocessor and PulseBlaster boards. The only functions that I know definitely behave differently are *pb\_init* and *pb\_select\_board* (see below), but this should have no effect in most cases.
- You can write programs to control a maximum number of boards set in spinapi (currently set in spinapi.h to 32); this is different from previous versions where the limit was set to 8.

Things fixed in this version:

- The code will now properly recognize and configure various types of boards supported by SpinAPI. It should work with all USB boards, but has only been tested with Radioprocessor and PulseBlaster boards.
- The method of retrieving data off Radioprocessor boards via usb was changed to be pseudo-asynchronous. This was found to give more reliable data transfer – at least under OSX 10.6.6.

A few things to note:

- This has been tested with os version 10.6.6. It probably works fine with 10.5.x, but I don't know how it will work with os versions older than this (e.g. 10.4.x or 10.3.x).
- There is no support for the fft functions – they are commented out in the source code (*spinapi.c*). If you want to use them, you should be able to un-comment them and recompile the library (see below) after installing the fftw library (see [www.fftw.org](http://www.fftw.org) for details).
- This version mostly preserves the SpinCore source filenames, with a couple of exceptions. The main files that contain functions differing from those SpinCore versions are *driver\_usb\_osx.c* and *extra\_pb\_stuff.c*.
- The primary file that interfaces with USB under OSX here is *driver\_usb\_osx.c*. However, there are differences that show up in other source files; for example, the original spinapi uses the watchdog timer to tell if data transfer has timed out. Under OSX there are “timeout” versions of USB read (and write) statements that can be used instead, so these are sometimes used here rather than watchdog. It is recommended that you check the return value of the function *pb\_get\_data* when retrieving data from the RadioProcessor – if the usb transfer times out this return value will be non-zero.
- Some functions don't work exactly the same as the original spinapi code; for example, *pb\_init* initializes all of the boards it finds on the USB bus at once, you can then use *pb\_select\_board* to select any board at any time; note that initializing the boards involves for the most part initializing USB communication and finding the board capabilities – if a board is initialized it does not lose whatever was previously programmed. Also, the *debug* function prints to the terminal rather than to a file (i.e. if debug is set to 1, debugging info will be displayed). Also, if in *pb\_setup\_filters*, DO\_ZERO is set, then the actual hardware filter setup is skipped, and just the calculated values are returned. This is done because the hardware filter setup requires several seconds in my experience and should only be done when absolutely needed. Note that DO\_ZERO has an actual function only on PCI Radioprocessor boards, so it's use here has no effect.

- I added an extra function to write the data – *pb\_write\_datafile*. This function has the same format in use as *pb\_write\_ascii*, but writes a data file that has the header information as ascii and the data as binary (32 bit integer). Because the data values that come off the RadioProcessor are large in my experience ( $>10^6$ ), this form of storage takes up about ½ the space of a full ascii file. The function *pb\_append\_datafile* is also included which appends data to an existing file for serial experiments. Note that, in these functions, the header has a fixed length so it can be easily skipped for reading files consisting of multiple records.
- When *pb\_init()* is executed, the boards are ordered by their USB location id in the case of multiple boards. This gives them a software-independent ordering that I have found generally follows the port number when the boards are all connected through a USB hub (e.g. if two boards are connected, one to port 0 and one to port 3, port 0 will be board 0 and port 3 will be board 1).
- Regarding the PTS functionality – this is included in this library, but you must have properly installed the ftd2xx library (see <http://www.ftdichip.com/Drivers/D2XX.htm>; the library is best installed in */usr/local/lib*; note that this will need to be renamed or symbolically linked to */usr/local/lib/libftd2xx.dylib* in order for the Xcode project to link it properly). The SpinCore code now allows addressing of multiple PTS units via their order on the USB bus. There are also included functions that are locally used to address PTS synthesizers that have specific differences in their device and product ids.

The installation instructions that follow assume you have Xcode Tools installed on your Mac. This is included in the main installation disk for the operating system in case you don't have it already installed – it can also be downloaded from the Apple web site <http://developer.apple.com>. It is also assumed you are familiar with entering unix commands from the Terminal, and that you are also familiar with unix file permissions.

1. Expand the included tar file. You should then have a directory **spinapi\_osx\_lib0111**, inside of which you will find a subdirectory **Build/Release**. The dynamic library file included there, **libspinapi\_osx\_lib.dylib**, is a version of the modified spinapi library for Intel processors only in osx 10.6. This library file does not need to be re-compiled and re-linked, but this could easily be done in Xcode if needed. The current version should also work building a PPC Mac version as well – you will need to build the library using Xcode on a PPC Mac if you want to do that. The primary change to make this possible is to check for byte ordering effects when reading and writing ram (functions *usb\_read\_reg* and *usb\_write\_reg*) – I thank Dieter Suter for realizing how to fix this. If you have questions about PPC vs Intel use of the driver, please let me know.
2. Before attempting to use the library, put the dylib file into a directory that is easily accessed by the compiler – I recommend */usr/local/lib*; e.g. **cp libspinapi\_osx\_lib.dylib /usr/local/lib**. You can also create a symbolic link.

Now, you just need to compile your programs; note that *spinapi.h* and *caps.h* need to be in the same directory as the program you are compiling (or in a shared directory such as */usr/local/include*) in order build the programs as written in the example code. For example, to build the excitation test program included in the RadioProcessor directory, for example:

```
cc excite_test.c -lspinapi_osx_lib -o excite_test
```

That is all that is required.

*Tom Pratum, Department of Chemistry, Western Washington University,  
September 30, 2011, [pratum@chem.wvu.edu](mailto:pratum@chem.wvu.edu)*