Neuroscope with NWB File Support

Neuroscope now has Neurodata Without Borders (NWB or HDF5) support, and Neuroscope has also been upgraded to nearly the most recent version of Qt.

For Mac OSX users, the following DMG can be used to install Neuroscope.
https://github.com/catalystneuro/Neurosuite5/blob/master/Install_Mac/neuroscope.dmg
For Windows, the following setup can be used to install Neuroscope.
https://github.com/catalystneuro/Neurosuite5/blob/master/Install_Win/Output/neurosetup.exe
We will provide notes to help build the software for other operating systems.

To read NWB format files, we need to tell Neuroscope where the data is found. For now, we use XML location files with nearly the same name as the original file.
We assume the NWB file has an extension NWB.
If your data file is called test.NWB, then the locations will be in a companion file called test**_loc.XML**.
That is, the location file replaces ".NWB" with "_loc.XML" and Neuroscope will find it. Place the location file in the same folder as the NWB file.

An example file is:

```
<parameters creator="neuroscope-1.2.3" version="1.0" >
   <nwb_dataset_name>/processing/ecephys/LFP/lfp/data</nwb_dataset_name>
   <nwb_sampling_name>/processing/ecephys/LFP/lfp/starting_time</nwb_sampling_name>
   <nwb_voltage_electrodes>/processing/ecephys/LFP/lfp/electrodes</nwb_voltage_electrodes>
   <nwb_voltage_electrodes_shanks>general/extracellular_ephys/electrodes/shank_electrode_number</nwb_voltage_electrodes_shanks>
   <nwb_spike_times_values>units/spike_times</nwb_spike_times_values>
   <nwb_spike_times_indices>units/spike_times_index</nwb_spike_times_indices>
   <nwb_spike_times_shanks>units/electrode_group</nwb_spike_times_shanks>
   <nwb_event_times>/stimulus/presentation/PulseStim_0V_10001ms_LD0/timestamps</nwb_event_times>
   <nwb_event_times>/stimulus/presentation/PulseStim_0V_10011ms_LD0/timestamps</nwb_event_times>
</parameters>
```

Location File Format
The NWB files may have voltage traces, spike times, and events.

Voltage:
The voltage trace data location uses the <nwb_dataset_name> tag.
We supply the location of the starting time in <nwb_sampling_name> that also tells us the sampling rate frequency.
We color the data according to shank using integers in <nwb_voltage_electrodes>.
These integers give us the indices from < nwb_voltage_electrodes_shanks>.

Spike Times:
The array of times for all units are found in <nwb_spike_times_values>.
<nwb_spike_times_indices> tells us when a new unit starts in the above array.
The shank of each unit is found in <nwb_spike_times_shanks>.

Events:
Events are found in <nwb_event_times>.
You may have more than one set of events, such as the example above that displays two sets.

The voltage traces may also be represented using time stamps that are found in: <nwb_voltage_timestamps>. For timestamped data, do not include: <nwb_sampling_name>. An example time-stamped file is shown below.

<parameters creator="neuroscope-1.2.3" version="1.0" >
  <nwb_dataset_name>/acquisition/test_ephys_data/data</nwb_dataset_name>
  <nwb_voltage_timestamps>/acquisition/test_ephys_data/timestamps</nwb_voltage_timestamps>
  <nwb_voltage_electrodes>/acquisition/test_ephys_data/electrodes</nwb_voltage_electrodes>
</parameters>

Reading NWB Files
Assuming the software is installed, run Neuroscope.
Using the **file** menu and the **open** command, select your NWB file. You should see the voltage traces.

To read units, use the **file** menu and **Load Cluster File(s)** option. You may need to change the extension to NWB, but read in the same NWB file you used before.

To read events, use the file menu and the **Load Event Files(s)** option. You may need to change the extension to NWB, but read in the same NWB file you used before. If you have 5 events, add the 5 events to your companion _loc.XML file.

From here out, the Neuroscope software should behave the same way as before.

We have tested the software using a few files. HDF5 is such a broad standard that there is a good chance that we missed a format that you use. Please pass on any bugs along with a small file example for the developers to use to fix the bugs. Thank you.

Neuroscope Development Notes

Building a Running Program in General

You will need QT, Cerebus, HDF5, LibNeurosuite, and Neuroscope to build a running program.

C++ Compiler

You need a C++ compiler, and you need to know where it is.
Microsoft Visual Studio, clang, gcc have all worked fine.

CMake

This tool is used to build libraries, if you are building libraries from scratch. If the Windows and Mac OSX libraries work for you, you don't need CMake.
Download Cmake from: https://cmake.org/
I also use the GUI version.

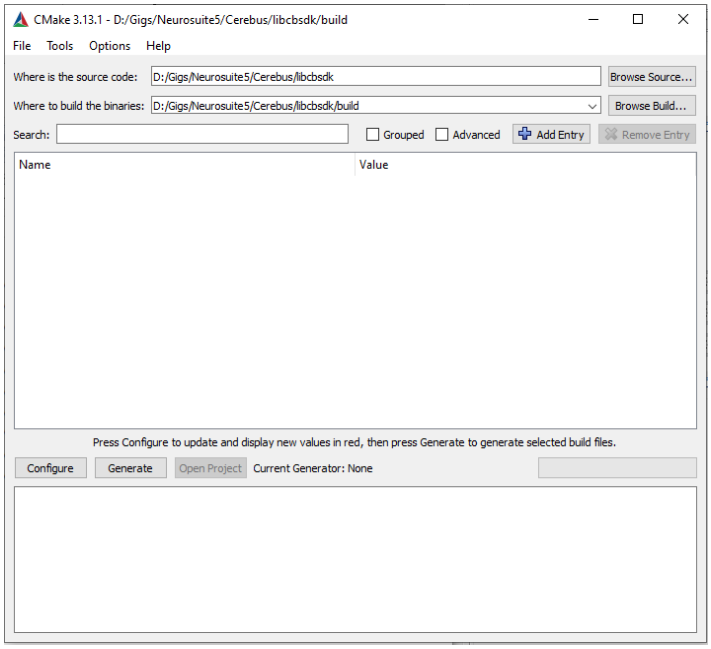
QT

- On the Mac (OSX), attempt to get the latest version of QT, say QT 5.12 or newer. Also install XCODE. Older versions of QT are not compatible with some OSX constraints.
- On Umbuntu or Linux Mint or similar, try to install QT 5.5.1. Older versions likely won't work, and versions 5.6.3 and newer use a Chromium component that Umbuntu may not want.
- I have not tried other versions of Linux, but I'd guess that you may find success using the latest version of QT, say QT5.13 as of May 2020.
- Windows needs 5.5.1 or newer.

For all operating systems, install Qt Creator, since we will use it to build the projects.

Cerebus

If you are not using Mac OSX or Windows, then clone
https://github.com/CompSciEng/libcbsdk.
You may want to get it from Florian Franzen's page, but I copied it in case it gets moved.
Run the Cmake GUI and point to the libcbsdk inside Cerebus. Point also to the build folder.



Click **Configure**, and I hope everything is OK. If not, fill in the missing information and click C**onfigure** again. Then click **Generate**.
On my Mac (OSX), I had the following settings.

- Cerebus CMake
- CMAKE_BUILD_TYPE Release
- CMAKE_EXECUTABLE_FORMAT MACHO
- CMAKE_INSTALL_PREFIX /usr/local
- Qt5Concurrent_DIR /Users/robertmoore/Qt/5.12.2/clang_64/lib/cmake/Qt5Concurrent
- Qt5Core_DIR /Users/robertmoore/Qt/5.12.2/clang_64/lib/cmake/Qt5Core
- QtXml_DIR /Users/robertmoore/Qt/5.12.2/clang_64/lib/cmake/Qt5Xml

My output file was found at:
Cerebus/libcbsdk/build/src/release/libcbsdk.dylib

The main output you need is a library called: **libcbsdk t**hat has an extension appropriate for your operating system. (e.g. Mac = dylib, Windows=dll) Write down where yours is located.

HDF5

From the official site, download HDF5. https://www.hdfgroup.org/solutions/hdf5/
You may want HDF5View to help diagnose issues, so I'd grab that also.
They will give you instructions on how to build the libraries. If you only want release libraries, you may be able to find pre-built libraries, and then you can ignore the rest of this section. I wanted debug libraries for Windows, so I had to build them.

Download, unzip, and look for the folder that holds HDF5config.cmake. For **Windows**, from that folder in a command window, type:
ctest -s HDF5config.cmake,BUILD_GENERATOR=VS201764,--enabledshared-shared –enable-cxx –with-zlib –enable-**debug** -vv -O hdf5.log

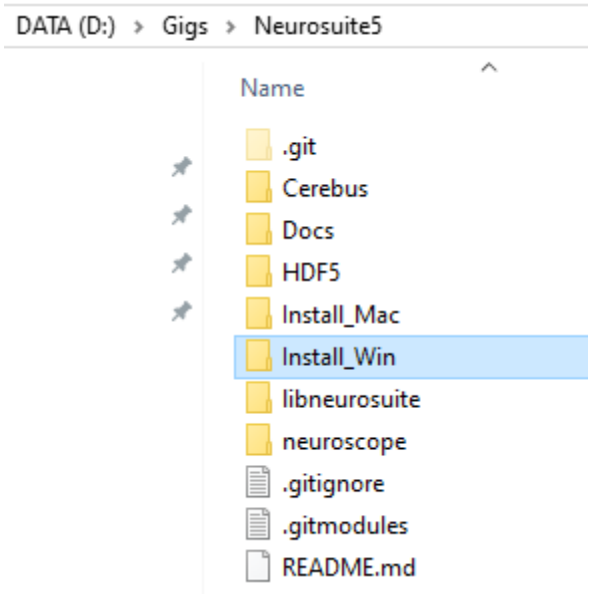
Find the debug libraries that you just built and copy them someplace else. Now rerun this script for the release libraries:
ctest -s HDF5config.cmake,BUILD_GENERATOR=VS201764,--enabledshared-shared –enable-cxx –with-zlib –enable-**production** -vv -O hdf5.log

If you did not move the library files, they get overwritten.
You now have the HDF5 libraries you need for debug and release.

LibNeurosuite
Neuroscope

Clone these to your local location.
You will use QtCreator to build these files.

On my Windows box, I use the following folder structure.



QtCreator
I could not get cmake to build neuroscope, so I used QtCreator, which is a free app from Qt.
You need to install QtCreator, if you have not already done so. Then configure your compiler of choice. I used clang for Mac OSX, and maybe Visual Studio 2017 for Windows. I think I used gcc for Linux and maybe also for one pass on the Mac.

The two **project** files are found on my system at:
Neuroscope5/neuroscope/neuroscope/neuroscope.**pro**
Neuroscope5/libneurosuite/libneurosuite/libneurosuite.**pro**
Load them both into QtCreator, and make neuroscope the master project.

Build libneurosuite and then neuroscope and see if they run.
You likely need to change the folder locations for a few libraries.

Ultimately, we will want to use cmake to build this system.

If neuroscope runs, then you have a working executable.

Bundle Application Files for the Mac OSX
The Mac needs a few changes to get the program and libraries to communicate. Here are the steps to do this the first time.
1. Copy neuroscope.app to ./Install_Mac folder.
   I put it in a folder called DoNotShip, since it may be huge
2. Build a nearly empty structure like:
   ~/Qt/5.12.2/clang_64/bin/**macdeployqt** neuroscope.app
   Your macdeployqt software is likely found somewhere else.
3. Set up your app folder like this:
   neuroscope.app
   Contents
   Frameworks
   folder:Cerebus dylib

folder:libneuroscope dylib
QT files
info.plist
MacOS
neuroscope
PkgInfo
PlugIns
Resources
icon file

4. copy the custom dylibs inside Framework.
/Cerebus/libcbsdk/build/src/release/libcbsdk.dylib
/libneurosuite/build-libne....-Release/*.dylib
5. modify how neuroscope finds the libraries.
A. Go to the executable inside MacOS
B. otool -L neuroscope
This is optional, but shows the libraries and locations.
I found the @rpath directory this way.
C. // install_name_tool -change liblibneurosuite.1.dylib
@executable_path/../Frameworks/libneurosuite/liblibneurosuite.1.dylib neuroscope
install_name_tool -change @rpath/liblibneurosuite.1.dylib
@executable_path/../Frameworks/libneurosuite/liblibneurosuite.1.dylib neuroscope
D. install_name_tool -change @rpath/libcbsdk.dylib
@executable_path/../Frameworks/Cerebus/libcbsdk.dylib neuroscope
6. Double click on neuroscope.app and make sure it runs.
Now the application and libraries communicate and you have a proper bundle.

Note: The author made a script called FixPaths.sh to replace step 5.

Make a DMG file:
1. Run Disk Utility.
2. File -> New Image -> Image from Folder take the DoNotShip folder from inside Install_Mac.
3. SaveAs Neuroscope.dmg likely in folder above Install_Mac.
4. Move the dmg inside Install_Mac.
5. Don't push the neuroscope.app file to github. It has files that are way too large.
6. Github likely stores it at:
https://github.com/CompSciEng/Neurosuite5/blob/AddNWB/Install_Mac/neuroscope.dmg

Handbook Documentation
The author could not find the proper software to modify the handbook information. Thus, it was converted to an html file using pandoc as follows.

$ pandoc --from docbook --to html --output myDoc.html index.docbook.txt

Programming Notes
A few overall comments may help the next developers.

Functions that read HDF5 files were placed in HDF5Utilities and assumed no knowledge of neuroscope in general. Perhaps, other software can reuse this code. One development fear is that we tested a floating point format and an integer format, but there could be other formats that we did not test and have not been implemented. Please pass on bug reports and a small example file for us to fix.

NWBReader.cpp is the glue between Neuroscope and HDF5. It uses NWBLocations to find where to look for data. Filling out the _loc.XML file may be considered tedious. We should consider writing a python script to simplify the process. Alternately, there may be a standard set of locations that avoids the need for a location file.

NWBTracesProviders was patterned a bit after the newer NEV and NSX trace providers, so we did not implement all the extra sorting of shanks. We assumed the shanks are arranged in the NWB file.

NWBclustersProvider reads the units and is relatively straightforward.

NWBEventsProvider reads events, but it assumes that multiple sets could be found in the same NWB file. Neuroscope in general was designed for separate event files, so we did a bit of a hack to permit multiple event “files” in one NWB file.

The software was upgraded to Qt5.12 or newer.