PyPlexStim Readme

PyPlexStim is a Python package for controlling the Plexon PlexStim electrical stimulator. Up to four PlexStim stimulators can be controlled through this package, as long as none of the stimulators are being controlled by the PlexStim GUI. PyPlexStim is only for Python 2 (2.7.3+) in Windows 7/8 32 or 64-bit, and is only compatible with PlexStim stimulators with hardware number 14-20-A-10-F (written on the bottom of the stimulator).

Installation

PyPlexStim's folder and file structure looks like this (folders in bold):

PyPlexStim
- **bin**
  - PlexStim.dll
  - PlexStim64.dll
- **pyplexstim**
  - __init__.py
  - pyplexstimlib.py
- PyPlexStimExample1.py
- PyPlexStimExample2.py
- PyPlexStimExample3.py

The easiest way to use PyPlexStim is to place the **bin** and **pyplexstim** folder into the same location as the Python program you will be writing that will use the modules in the package.

Sample Programs

Three sample programs are included with PyPlexStim. They demonstrate the general workflow of setting up, loading stimulation parameters, and stimulating. When a PlexStim stimulator is attached to the PC, these examples should run without error.

Technical Details

PyPlexStim depends on PlexStim.dll. This .dll contains functions for initializing the stimulator(s), getting and setting configurations, stimulating, etc. Python includes a module with its standard library called ctypes, which provides classes for communicating with .dll files. In this sense, PyPlexStim is a Python wrapper for PlexStim.dll.

Functions

Each function in the PyPlexStim class located in pyplexstimlib.py is documented with a description of its purpose, and an example of how to call it. The functions are listed below.

Initialization Functions

res = ps_init_all_stim()
res = ps_close_stim(stim_n)
res = ps_close_all_stim()

Loading Channel Functions

res = ps_load_channel(stim_n, ch_n)
res = ps_load_all_channels(stim_n)

Stimulation Functions

res = ps_start_stim_all_channels(stim_n)
res = ps_stop_stim_all_channels(stim_n)
res = ps_start_stim_channel(stim_n, ch_n)
res = ps_stop_stim_channel(stim_n, ch_n)
res = ps_abort(stim_n)
res = ps_abort_all()

Pattern Functions

res = ps_set_pattern_type(stim_n, ch_n, pattern_type)
pattern_type, res = ps_get_pattern_type(stim_n, ch_n)
res = ps_set_rect_param(stim_n, ch_n, param)
param, res = ps_get_rect_param(stim_n, ch_n)
res = ps_load_arb_pattern(stim_n, ch_n, pattern_path)
n_points, res = ps_get_n_points_arb_pattern(stim_n, ch_n)
coords, res = ps_get_arb_pattern_points(stim_n, ch_n, n_points)
x_coords, res = ps_get_arb_pattern_points_x(stim_n, ch_n, n_points)
y_coords, res = ps_get_arb_pattern_points_y(stim_n, ch_n, n_points)

Settings Functions

res = ps_set_digital_output_mode(stim_n, mode)
mode, res = ps_get_digital_output_mode(stim_n)
res = ps_set_monitor_channel(stim_n, mon_ch_n)
res = ps_get_monitor_channel(stim_n)
res = ps_set_period(stim_n, ch_n, period)
period, res = ps_get_period(stim_n, ch_n)
res = ps_set_rate(stim_n, ch_n, rate)
rate, res = ps_get_rate(stim_n, ch_n)
res = ps_set_repetitions(stim_n, ch_n, repetitions)
repetitions, res = ps_get_repetitions(stim_n, ch_n)
res = ps_set_trigger_mode(stim_n, mode)

mode, res = ps_get_trigger_mode(stim_n)
res = ps_set_vmon_scaling(stim_n, scaling)
scaling, res = ps_get_vmon_scaling(stim_n)
res = ps_set_auto_discharge(stim_n, enabled)
is_auto_discharge, res = ps_get_auto_discharge(stim_n)

Information Functions

n, res =es = ps_channel_stim_started(stim_n, ch_n)
error_s ps_get_n_stim()
n_ch, res = ps_get_n_channels(stim_n)
is_started, rtring, res = ps_get_extended_error_info(error_code)
description, res = ps_get_description(stim_n)
fw_version, res = ps_get_fw_version(stim_n)
serial, res = ps_get_serial_number(stim_n)
duration, res = ps_get_stim_pattern_duration(stim_n, ch_n)
is_balanced, res = ps_is_waveform_balanced(stim_n, ch_n)

Tips and Tricks

By default, the PyPlexStim class looks for PlexStim.dll or PlexStim64.dll in the .\bin folder, but you can override that folder location when initializing the class. For example:

```
>>>from pyplexstim import PyPlexStim
>>>p = PyPlexStim('c:\\other\\location')
```

There are several variable names in pyplexstimlib.py that have been included for readability convenience (and for consistency with the C/C++ SDK). For example, PS_OK is a return value, equal to 0, that all methods in PyPlexStim will return if the function executes correctly. It's equally valid to check for 0 on return, instead of PS_OK.

The functions ps_get_arb_pattern_points(),ps_get_arb_pattern_points_x(), and ps_get_arb_pattern_points_y(), return coordinates from a loaded arbitrary pattern. These functions are used for the PlexStim GUI's waveform drawing view, and won't be generally useful for anything else. They have been included for consistency with the C/C++ SDK.

If you're sensitive to stimulation timing, don't try to use any of the PlexStim SDKs (C/C++, Matlab, Python) to control stimulation output onset. The PlexStim device has a digital input port that can be set up to control output onset. The PlexStim SDKs can be used to set parameters of the stimulation, and set the stimulation onset to be controlled by the digital input port. This way you can use a more precisely timed digital output device to control stimulation onset.