Pulse-Sequence Instrument Control Tools (PSICT)

About

The Pulse-Sequence Instrument Control Tools (PSICT) package is a Python module designed to facilitate programmatic, script-based control over experiments using Labber. It is specifically designed for use with the Single-Qubit Pulse Generator driver (amongst others), but functions with any instruments that can be controlled via the Labber interface.

The PSICT-UIF (Unified Interface Framework) is the primary tool in this module, directly responsible for providing a convenient interface between user scripts and Labber. Additional tools in this module include a higher-level automation class, virtual instrument drivers for use with Labber, and examples of user scripts to showcase PSICT in action.

The latest version of PSICT can be found on the repository. If you are reading this document in pdf format, it has been generated from the README file in the repository root. If you desire to read this document in a pdf format, it can be found under `docs/README.pdf`.

Requirements

Version numbers indicate latest compatible version. Backwards compatibility is thus not guaranteed. If backwards compatibility is required, please create an issue.

- Python 3.6.8+ (not tested with 3.7)
- Labber, as well as the `Labber` python API (`1.6.3`)
- `numpy` (`1.16.2`)
- `h5py` (`2.9.0`)

Installation

Currently, the only available means is directly from the repository. Clone or download the repository and add to your import path. To test, run the follwing in a Python prompt or script:

```
import PSICT_UIF
PSICT_UIF.__version__
```

We are planning to release this package via a Python distribution platform (`pip` or Anaconda), and that will be done as soon as all the kinks are ironed out. Stay tuned!

Using the PSICT-UIF with Labber

Please note: the full documentation is still very much under construction. Please contact the devs directly if you need assistance with specific functionality!

The PSICT-UIF is designed to be used with Labber through a 'worker script', written and executed by the user. Various PSICT-UIF methods in the worker script will set the values (including iterations and relations) of instrument parameters, invoke Labber to actually carry out the desired experiment, and do administrative tasks such as copying scripts and additional files for record-keeping. The basics of the tasks and relationships are shown in Figure 1. More intricate use cases are described in the documentation.

Unfortunately, even basic usage is a bit too complex to describe in a humble readme file. Thus, the content here is provided merely as an introduction/overview, and a more complete explanation of how to actually go about using PSICT and the PSICT-UIF with Labber must be outsourced to the documentation.

There are several advantages to using a script-based approach to running experiments (as compared to the Labber GUI). However, the biggest advantage (and in fact the reason for PSICT's existence in the first place) is centered around the design of the SQPG driver (or in general, any pulse-sequence generation driver that aims to provide *full control over every pulse parameter canonically within the Labber framework*).

Briefly, the emphasized requirement results in each pulse being implemented 'statically' in the driver, resulting in channel names which are fixed by the position of the pulse within the sequence, and are not related to the function of the pulse itself. For example, the amplitude of the second pulse in the sequence will always be named `Amplitude #2`, regardless of whether pulse 2 is serving the function of a readout, trigger, qubit excitation, etc. The PSICT-UIF was created to remedy this, allowing the user to have more idiomatic control and understanding of pulse parameters. As a happy consequence, this also makes re-arranging existing pulses for new or modified sequences a trivial task (which is not the case when using the Labber GUI), as well as providing motivation for all the other good features that are now present in PSICT.

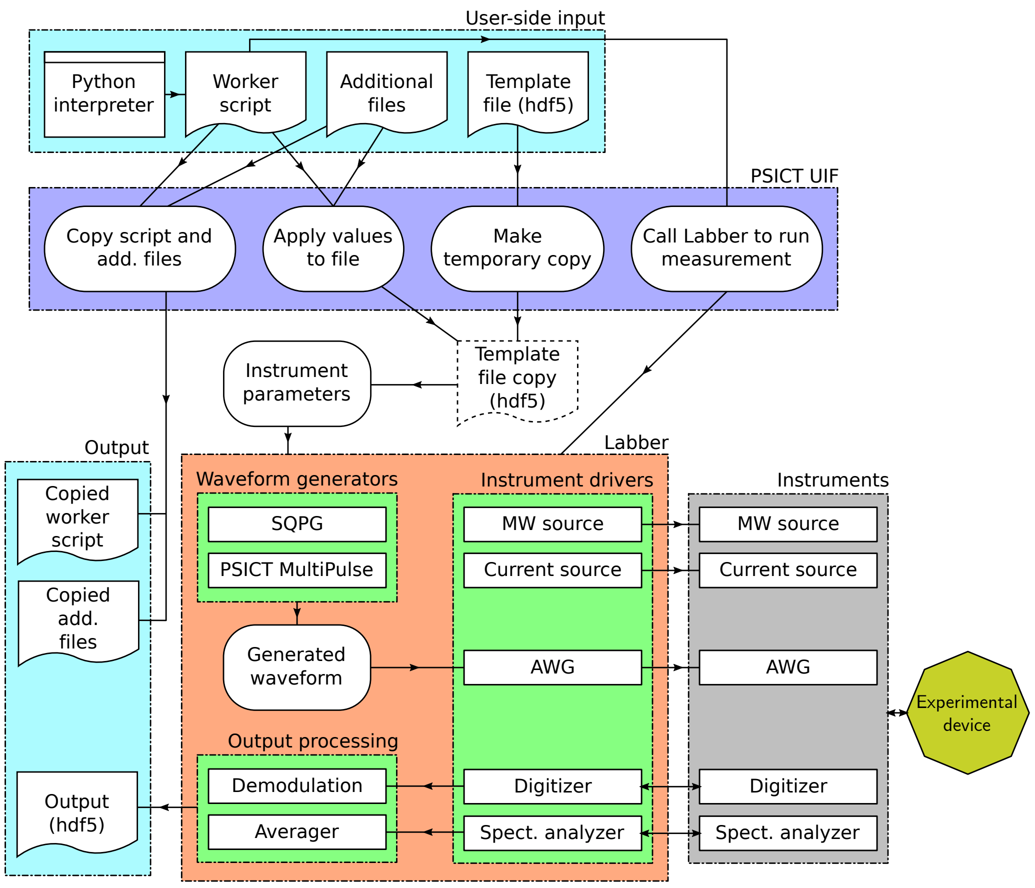

Figure 1: *Organogram of relationships between PSICT-UIF, Labber, physical instruments, and all files and scripts involved in the context of running a single experiment. Cyan groups highlight input and output files. Note that, in principle, any appropriate drivers and instruments can be substituted or added to those shown; the diagram lists the most common examples for the purpose of illustration.*

Additional modules in PSICT

Here, we provide a brief overview of the roles and capabilities of additional modules/classes/etc that are provided as part of the PSICT package, but are not part of the core PSICT-UIF functionality.

For more in-depth guides and descriptions, please see the full documentation.

WorkerScriptManager

The `WorkerScriptManager` class is designed to be used in 'master' scripts, i.e. scripts controlling and running multiple experiments in series. Importantly, analysis of results can be interspersed with experiment execution, allowing automated feedback without the need for user intervention.

The guiding design principle of the `WorkerScriptManager`-enabled 'master' scripts is the ability to seamlessly switch between running multiple experiments with a master script and single experiments with a worker script, with all parameter values updated automatically (if done right). The PSICT/Labber hierarchy (and how the layers interact with each other) is explained in more detail in the full documentation.

PSICT MultiPulse

`MultiPulse` is a virtual instrument driver for Labber, aimed at creating pulse sequences that are longer than those allowed by drivers such as the SQPG. In principle, pulse sequences of arbitrary length (in terms of number of pulses) can be generated; however, the trade-off is the loss of control over individual pulse parameters within the Labber interface. It is extremely well-suited for applications such as randomized benchmarking, where a large number of random permutations of pulses are selected from a small, predefined set.

Development

Releases

Two primary branches are available on the repository:

- `master` is the latest stable release, currently `1.2`.
- `dev` tracks more experimental features which have been tested to some degree, currently `1.3dev0.1`.

For a specific release version, please find the appropriate tag in the repository history.

Features in development

`1.3` *planned features*

- Full documentation!
- Copying additional files on measurement to be included in the UIF [#6].
- Make MultiPulse pulse generation more efficient for very long pulse sequences by pre-calculating envelopes and modulation signals [#7].

`2.0` *planned features*

- Overhaul of the SQPG/pulse-parameter-specification backend to allow more direct interfacing with the SQPG and Labber. Amongst numerous fundamental benefits in terms of simplifying the encoding of convoluted pulse sequences in the worker scripts, doing this will also allow the `Spacing` parameter to be used directly as an iteration variable [#5].

License

This module is licensed under the Expat (MIT) license.

Authors

Code

Sam Wolski

Experiments and testing

Dany Lachance-Quirion

Nakamura–Usami group, Research Center for Advanced Science and Technology, The University of Tokyo