Willow Documentation

Release 2.0

LeafLabs LLC

Jul 20, 2017

CONTENTS

Contents:

CHAPTER

ONE

LEGAL AND OTHER NOTICES

1.1 Work In Progress

This document represents work in progress, and should not be construed as final or binding.

1.2 Copyright Notice

CHAPTER

TWO

ABOUT WILLOW

Willow is an electrophysiology system developed by LeafLabs in collaboration with MIT's Synthetic Neurobiology Group. The system is based on the Intan RHD chips, and utilizes a scalable, direct-to-disk architecture to efficiently record 1024 channels of neural data.

This documentation explains the setup and usage of the Willow system, as of the 2.0 release, Spring 2017. The latest release of Willow is supported by the website:

http://willowephys.com

This documentation is version 2.0 and was last updated on Jul 20, 2017. A PDF copy of this documentation is available here.

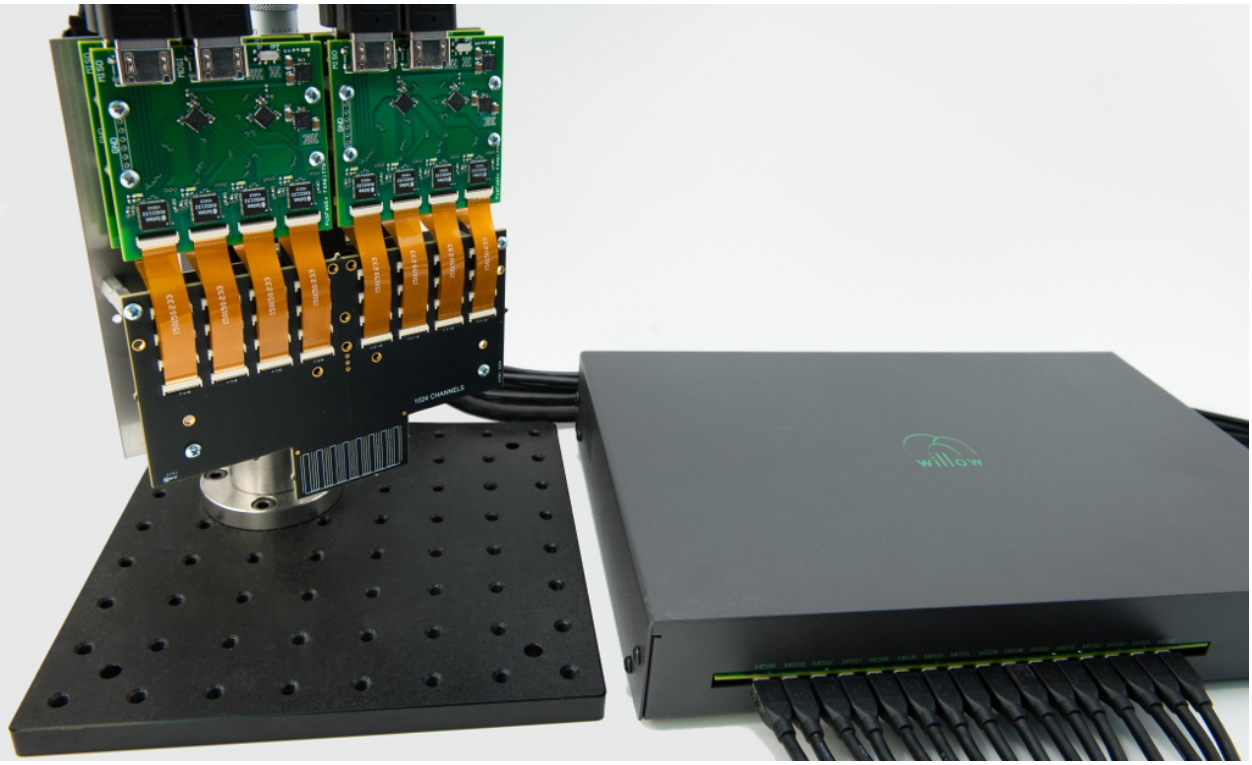


Figure2.1.: Willow Datanode with a headstage assembly.

CHAPTER

THREE

WILLOW QUICK START GUIDE

Contents

3.1 Hardware Connections

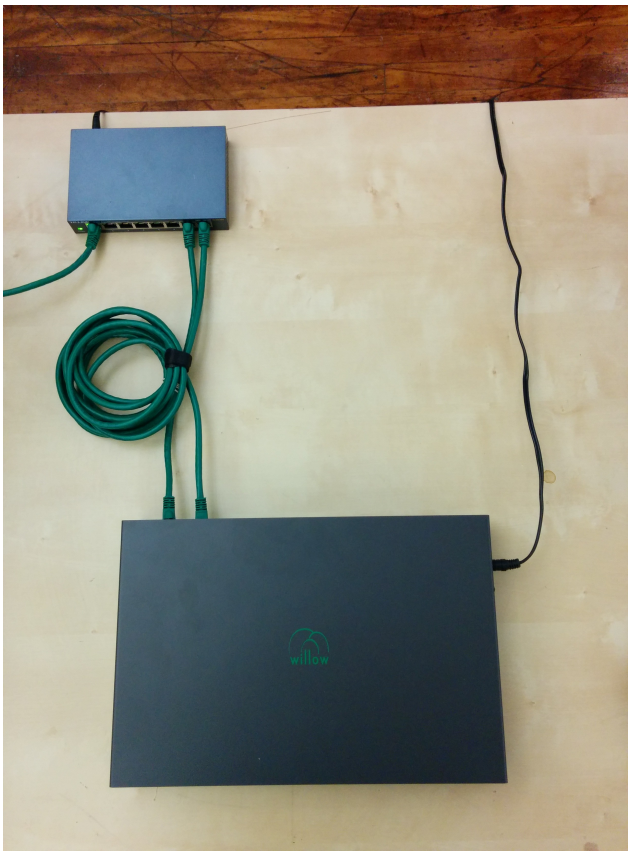
1. Connect Datanode Power Supply to port labeled 12VDC on right side of Datanode.

Figure3.1.: Datanode 12VDC power port and on/off switch.

Warning: All Willow Datanodes come with a floating 12VDC power supply, and are thus *not grounded*. This configuration allows the experimenter to chose how the Willow system is grounded for best signal quality, e.g. close to the animal. However, it also means that the experimenter must take care not to operate the Datanode in an ungrounded state such as with no headstages attached, lest static charge accumulate on the Datanode. Switching

to a grounded power supply resolves any grounding concerns (since the Datanode is always grounded) at the risk of inferior signal quality due to increased noise.

2. Connect TCP port and UDP port on Datanode to network switch (any two ports will do) using two Ethernet cables.



3. Connect workstation to the same network switch via Ethernet cable. For information about networking with pre-configured workstations from LeafLabs, see *Network Configuration*.
4. Connect Datanode to headstages via HDMI cables. Connect any Datanode MISO/MOSI paired-ports to respective MISO/MOSI paired-ports on any headstage. Repeat desired number of paired-port connections between Datanode and headstages, as needed. Arbitrary configurations of headstage connectivity at the MISO/MOSI interface is allowed, provided that at least one headstage is connected to the Datanode. If zero headstages are connected, then a snapshot will result in a timeout error.
5. Ensure that an SSD is in the drive bay.
6. Slide power switch on side of Datanode to ON position to power Datanode. Wait until white LEDs on headstages flash twice (takes about 15 seconds) and then remain dark. At this point, the connection topology should look as shown in *System Overview*.

Note: The status of LEDs (lit vs not) is random before completion of initialization.

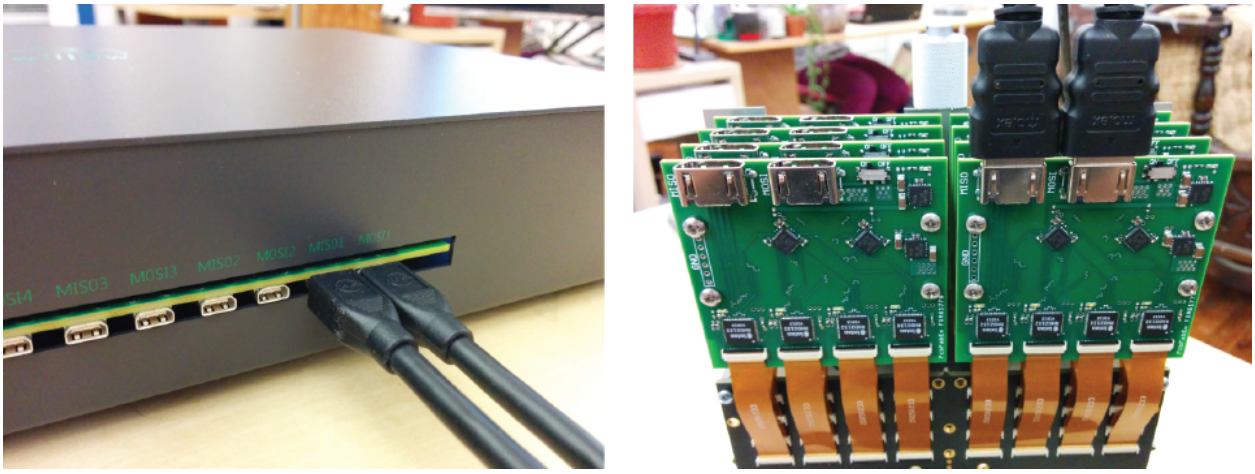


Figure3.2.: Example of Datanode paired-ports MISO1 and MOSI1 (left) connected to a headstage's MISO/MOSI ports (right).

3.2 Login to the Workstation

On a pre-configured Willow Workstation from LeafLabs, the default login account is *leaf* and password is *helloworld*.

Note: The default login account does not have root/administrator privileges, and thus cannot install software. Please contact LeafLabs for secure exchange of account details with root privileges.

3.3 Network Confguration

A pre-configured workstation from LeafLabs has two NICs - one built-in to the motherboard (labelled *Internet*) and a second as a PCI card (labelled *Willow*) - to allow internet access while communicating with the Willow datanode. Each NIC is configured for one purpose or the other but not both. Thus, it is important to establish correct connectivity by following the labels.

In case the labels are missing or illegible, correct connectivity can still be determined. To begin, in the case of reversed connectivity, WillowGUI will display no connectivity to DataNode (even though the datanode can be pinged at 192.168.1.2) and no IP address will be issued by DHCP to NIC intended for internet (as seen by running *ipconfig* at a command prompt). To restore proper function, swap the ethernet cables connected to the workstation. WillowGUI will display no connectivity to Datanode (see Figure 5.1.) even though the datanode can be pinged at 192.168.1.2, and no IP address will be issued by DHCP to NIC intended for internet (as seen by running *ipconfig* at a command prompt). To restore proper function, swap the ethernet cables connected to the workstation.

To support high data rate network traffic, it is important configure the size in bytes of network packets. If you're using a pre-configured Willow workstation, this happens automatically upon startup. In the case of a non-pre-configured workstation, a utility script is included with the *Daemon* in *willow-daemon/util*:

```
$ cd leafysd/util
$ ./expand_eth_buffers.sh <interface>
```

where <*interface*> is the name of your network interface (default is eth0).

Note: This script requires sudo privileges. See *Login to the Workstation*.

3.4 GUI Startup

On a pre-configured Willow workstation, you can start WillowGUI by clicking the WillowGUI icon on the Unity Launcher side panel.

Alternatively, you can start WillowGUI from the top-level folder of the local repository (e.g. *willowgui*) with

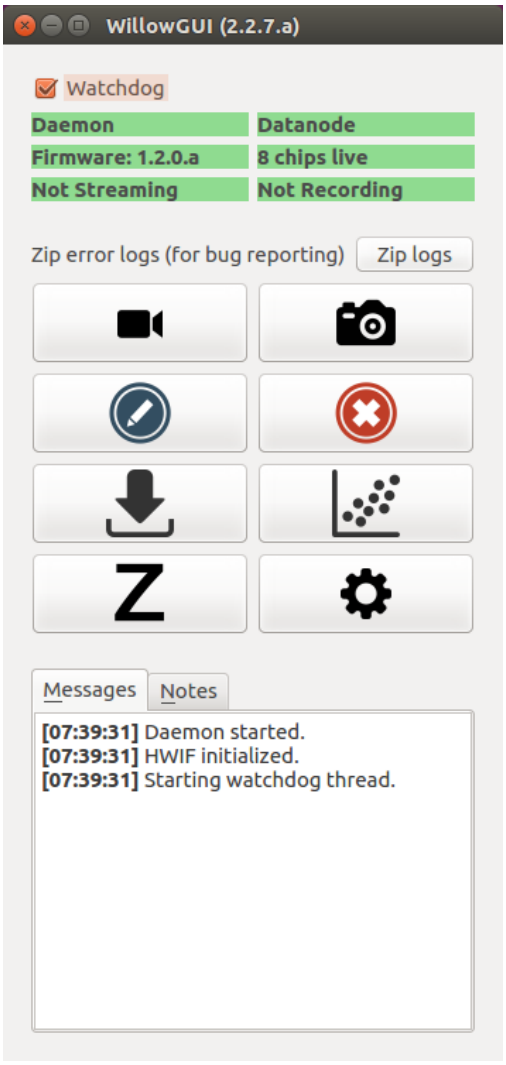
```
$ src/main.py
```

The first time you run the GUI, a Configuration Wizard will appear. Follow the instructions and click Finish.

If the setup was successful, the WillowGUI control panel will appear on the screen.


Willow GUI


WillowGUI Configuration Wizard
WillowGUI Configuration Wizard
WillowGUI has detected that config.json is missing. Probably this means that you are starting the application for the first time on this system. This wizard will help you configure your setup. You only need to do this once per installation.
< Back
Next >
Cancel


WillowGUI (2.2.7.a)
Watchdog
Daemon
Datanode
Firmware: 1.2.0.a
8 chips live
Not Streaming
Not Recording
Zip error logs (for bug reporting)
Zip logs
Messages
Notes
[07:39:31] Daemon started.
[07:39:31] HWIF initialized.
[07:39:31] Starting watchdog thread.

3.5 GUI Usage

This section is offered as a quick-start guide to basic GUI usage. For more in-depth documentation including installation instructions, please refer to the *Willow Software User Manual*.

1. Stream live data:

Click this button to launch a Stream....UPDATE WITH LATEST.

Add figure

2. Take a snapshot:

Click this button to take a snapshot, which is a short (1-10 second) sampling of all 1024 channels, stored in an HDF5 data file. A dialog box will appear in which number of samples to collect and the target filename can be entered. By default, the snapshot will be 1 second long (30000 samples) and have a timestamp-specific filename of the form:

```
snapshot_YYYYMMDD-hhmmss.h5.
```

Selecting "Plot When Finished" to open the snapshot data in a Plot Window (see Section 6).

3. Start recording:

Click this button to start recording to the Datanode.

> **Warning:** Each recording will start at the beginning of the disk, overwriting any previously recorded experiments. To ensure that no data is lost, make certain that any important experiments have been transferred off the Datanode (see Transfer Experiment below) before recording. While recording, the recording label in the status bar will turn red and list the current disk usage.

4. Stop recording:

Click this button to stop recording. The 'recording' status bar will turn back to its green "Not Recording" state.

5. Transfer experiment:

Click this button to transfer an experiment (i.e. previous recording) to the workstation. YA dialog box will appear in which the number of samples to transfer (or entire recording), and the target filename can be selected. Select "Name Automatically" to name the file with the UNIX time-stamp from when the recording was started.

6. Plot data:

Click this button to plot data from a previously acquired recording or snapshot. A file browser will appear in which the desired file can be selected. A subsequent dialog box will appear in which the amount of data desired for import can be selected. Upon selection, a Plot Window will open in which channel traces can be viewed as line plots. (Fig. 8)

Click "Waterfall" near the top-right of the Plot Window to open a Waterfall Plot – a 2D spectrogram-like visualization with channel count on the y-axis, and time on the x-axis. (Fig. 9)

ADD FIGURES

3.6 Retrieve Data from SSD Using SATA-to-USB cable

Data can be pulled from SSDs at any time using the *sata2hdf5* program that comes installed with the *Daemon* software. If not already done, first *Install* the daemon software. Then, using a SATA-to-USB adapter (one is included with every Willow system) attach to the PC one SSD, by first powering down the Willow Datanode and ejecting the SSD.

Note: To ensure that the PC operating system quickly registers the SSD, connect the SSD to adapter first, then connect adapter to USB connector on PC.

Then,

1. Determine the device name assigned to USB-to-SATA adapter by running *ls /dev/sd*‘* before and after attaching the adapter. The new device name is the desired one.
2. Determine the start time of data (recording on SSD), i.e. the UNIX time on the laptop when recording started.

Example:

```
$ sudo ~/src/willow-daemon/build/sata2hdf5 -c 30000 -o 0 /dev/sdd test.h5
Starting cookie: 0x57000886
Starting board id: 0x3746f441
Copied 30000...
Copied 30000 board samples.

0x57000886 = 1459619974 seconds since epoch = 2016-04-02 13:59:34+00:00 EST.
```

3. From snapshots (which have time embedded in filename), calculate the sample offset (which determines where in SSD to start extracting data). So, for data at 15:00 EST: sample offset = 15:00-14:00 = 60 minutes = 3600 sec = 3600*30e3 samples = 108000000.
4. Decide how many samples we want, e.g. 1 min = 60 sec*30e3 samples/sec = 1800000.
5. Finally, run sata2hdf5:

```
$ sudo ~/src/willow-daemon/build/sata2hdf5 -c 1800000 -o 108000000 /dev/sdd data_
↪offset_1hr_length_1min.h5
```

Note: At this time obtaining a subset of the channels (e.g. 64 out of 1024) is not possible unfortunately, neither through WillowGUI or from *sata2hdf5*.

Figure3.3.: SATA-to-USB adapter with SSD attached.

CHAPTER

FOUR

WILLOW HARDWARE USER MANUAL

Contents

4.1 System Overview

The Willow system is comprised of both hardware and software components. On the hardware side, there is the Datanode, which is the main data acquisition and storage computer. The Datanode collects data from the headstages, which are modular amplifier and digitizer boards. Each headstage acquires from 128 channels, and each Datanode can collect from as many as 8 headstages. The Datanode receives commands from the workstation - a computer running Linux. The Datanode and the workstation communicate over a high-speed Ethernet connection.

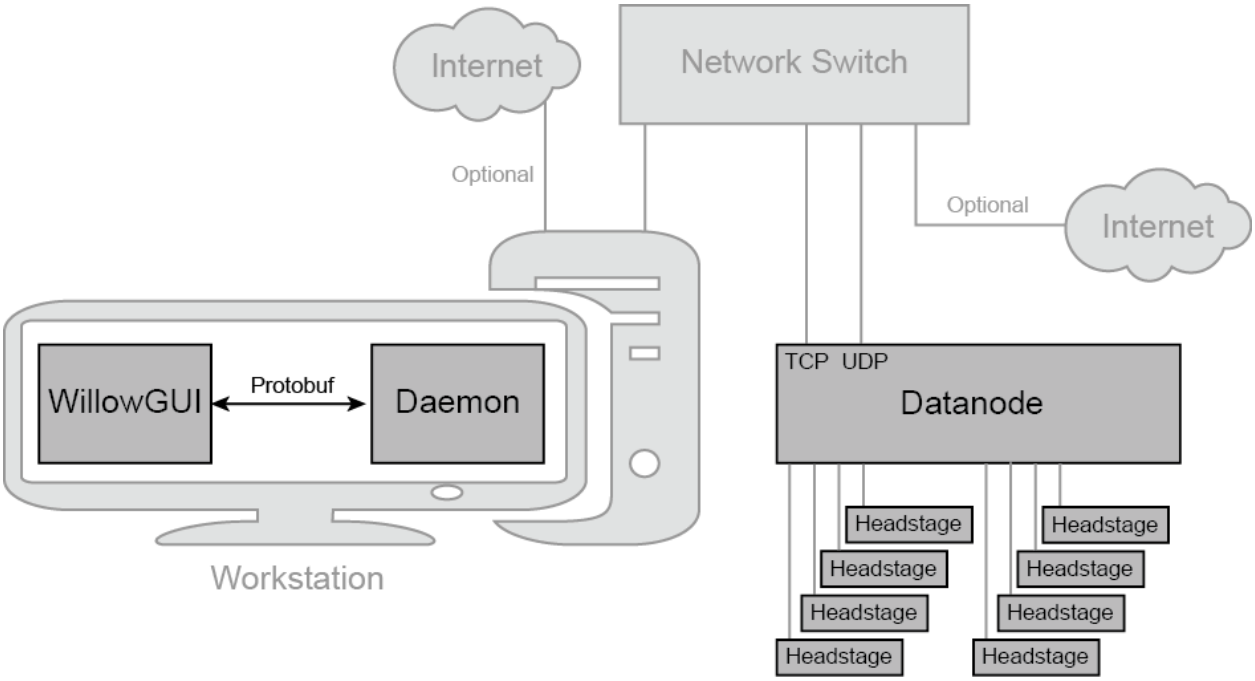
The workstation itself runs a multi-layered software stack. The *Daemon* handles the low-level communication with the Datanode, data collection, routing, and I/O. The daemon exposes a control interface based on Google's Protocol Buffers (protobuf). The *WillowGUI* software wraps this control interface in some higher-level functions that allow the user to record, stream, an analyze data with the convenience of a graphical user interface.

Note: As of Willow 2.0, Linux is the only supported operating system.

4.2 Precautions

Our headstages incorporate four RHD2000 Series Digital Electrophysiology Interface Chips. According to the datasheet for these chips:

> "All CMOS integrated circuits are susceptible to damage by exposure to electrostatic discharge (ESD) from charged bodies. Electrostatic charges of greater than 1000 V can accumulate on the human body or test equipment and can discharge without detection. All RHD2000 chips incorporate protection circuitry to guard against mild ESD events. However, permanent damage may occur on devices subjected to high



energy electrostatic discharges. It is important for users to understand the nature of the ESD protection circuitry used on the chip.

On-chip passive elements (diodes and resistors) are used for ESD protection at the input to each amplifier. Diodes are connected to GND and VESD, and are used to bleed off charge quickly to prevent the voltage on the series capacitors from exceeding damaging levels. Small series resistors (370 Ω and 200 Ω) create voltage drops in response to large ESD currents, further protecting the amplifiers."

Warning: Permanent damage may occur on devices subjected to high-energy electrostatic discharges. To avoid damaging the headstages, ensure the handler is grounded and thus free of electrostatic charge before handling any headstage. When not in use, headstages should be stored in an antistatic bag.

4.3 Performance Specs

Each Willow Datanode can acquire data from 1024 channels at 16-bit depth at 30 kHz. In addition, the Datanode has sufficient bandwidth to sample three auxiliary channels which can be chosen arbitrarily from any of the various registers in the RHD2132 chip, or from the current state of 16 GPIO channels on the DAQ module. By default the FPGA is programmed to automatically record the state of all GPIO pins at 2 kHz sample rate. In aggregate, this amounts to 1120 channels at 30 kS/s and 16-bit resolution.

The total data rate to acquire 1024 channels of data, including auxiliary data, meta-data and zero padding (each described in detail in the paper linked below), is 122.88 MB/s. At this data rate, a 512 GB hard drive, for example, can store up to 70 min of data from 1024 channels.

For more details, see A direct-to-drive neural data acquisition system.

4.4 How to Connect Hardware

1. Connect Datanode Power Supply to port labeled 12VDC on right side of Datanode.

Figure4.1.: Datanode 12VDC power port and on/off switch.

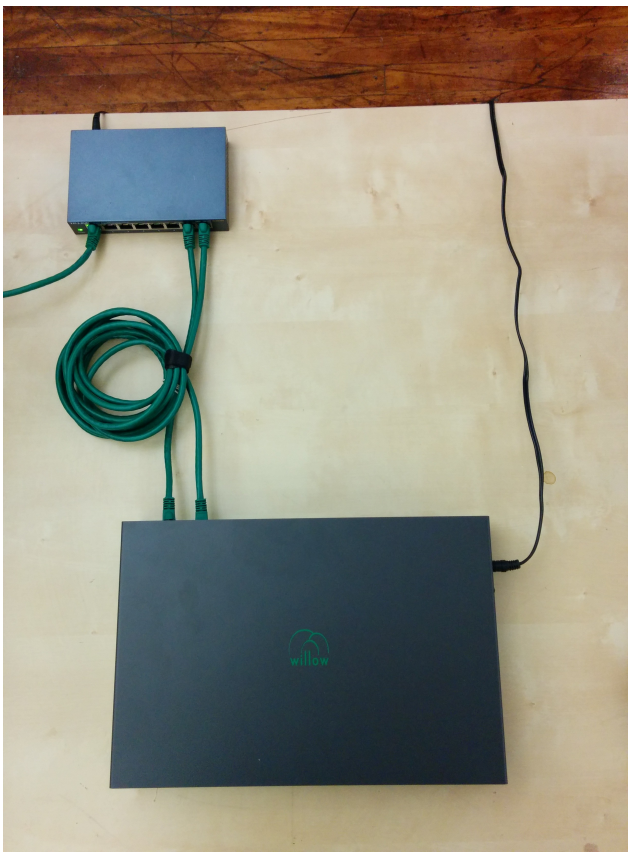
Warning: All Willow Datanodes come with a floating 12VDC power supply, and are thus *not grounded*. This configuration allows the experimenter to chose how the Willow system is grounded for best signal quality, e.g. close to the animal. However, it also means that the experimenter must take care not to operate the Datanode in an ungrounded state such as with no headstages attached, lest static charge accumulate on the Datanode. Switching to a grounded power supply resolves any grounding concerns (since the Datanode is always grounded) at the risk of inferior signal quality due to increased noise.

2. Connect TCP port and UDP port on Datanode to network switch (any two ports will do) using two Ethernet cables.
3. Connect workstation to the same network switch via Ethernet cable. For information about networking with pre-configured workstations from LeafLabs, see *Network Configuration*.
4. Connect Datanode to headstages via HDMI cables. Connect any Datanode MISO/MOSI paired-ports to respective MISO/MOSI paired-ports on any headstage. Repeat desired number of paired-port connections between Datanode and headstages, as needed. Arbitrary configurations of headstage connectivity at the MISO/MOSI interface is allowed, provided that at least one headstage is connected to the Datanode. If zero headstages are connected, then a snapshot will result in a timeout error.
5. Ensure that an SSD is in the drive bay.
6. Slide power switch on side of Datanode to ON position to power Datanode. Wait until white LEDs on headstages flash twice (takes about 15 seconds) and then remain dark. At this point, the connection topology should look as shown in *System Overview*.

Note: The status of LEDs (lit vs not) is random before completion of initialization.

4.5 Network Configuration

A pre-configured workstation from LeafLabs has two NICs - one built-in to the motherboard (labelled *Internet*) and a second as a PCI card (labelled *Willow*) - to allow internet access while communicating with the Willow datanode.



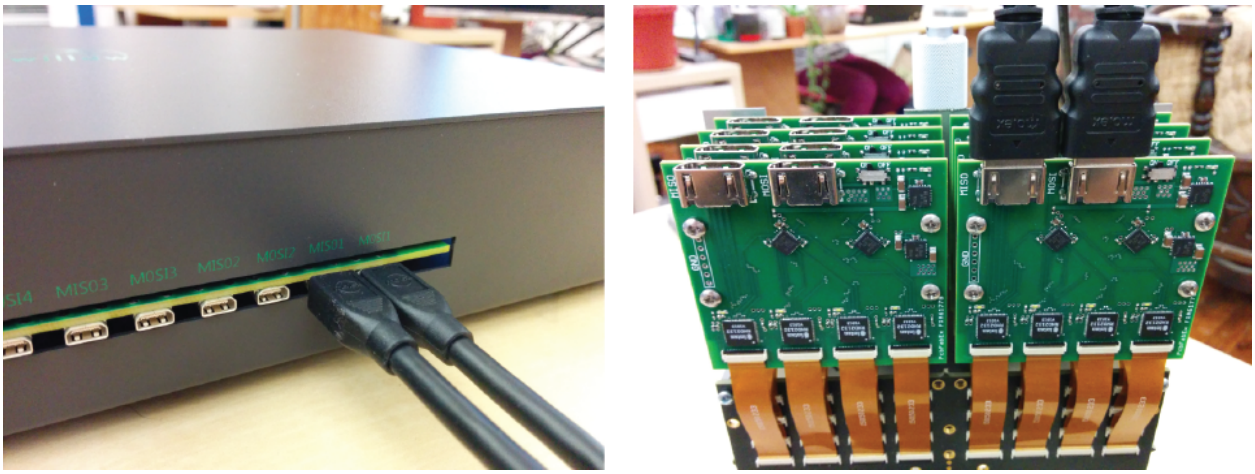


Figure4.2.: Example of Datanode paired-ports MISO1 and MOSI1 (left) connected to a headstage's MISO/MOSI ports (right).

Each NIC is configured for one purpose or the other but not both. Thus, it is important to establish correct connectivity by following the labels.

In case the labels are missing or illegible, correct connectivity can still be determined. To begin, in the case of reversed connectivity, WillowGUI will display no connectivity to DataNode (even though the datanode can be pinged at 192.168.1.2) and no IP address will be issued by DHCP to NIC intended for internet (as seen by running *ipconfig* at a command prompt). To restore proper function, swap the ethernet cables connected to the workstation. WillowGUI will display no connectivity to Datanode (see Figure 5.1.) even though the datanode can be pinged at 192.168.1.2, and no IP address will be issued by DHCP to NIC intended for internet (as seen by running *ipconfig* at a command prompt). To restore proper function, swap the ethernet cables connected to the workstation.

To support high data rate network traffic, it is important configure the size in bytes of network packets. If you're using a pre-configured Willow workstation, this happens automatically upon startup. In the case of a non-pre-configured workstation, a utility script is included with the *Daemon* in *willow-daemon/util*:

```
$ cd leafysd/util
$ ./expand_eth_buffers.sh <interface>
```

where <*interface*> is the name of your network interface (default is eth0).

Note: This script requires sudo privileges. See *Login to the Workstation*.

4.6 Troubleshooting

4.6.1 In WillowGUI, streaming works fine but snapshots fail and yield "empty" (KB instead of MB) files

Check the *Network Configuration*. Specifically, confirm that the MTU was increased from default (1500 bytes) to 4096.

4.6.2 What happens when capacity of SSD is less than configured in Settings?

For example, suppose the Datanode Storage Capacity in *Settings* is configured for 2 TB, but the SSD is 0.5 TB. What happens during a recording when the SSD fills up? The Datanode will present an error condition which will show up in the WillowGUI status bar. To clear the (hardware) error, the Datanode must be restarted by cycling its power.

4.6.3 If the WillowGUI or daemon software or even the workstation fails, what happens to the Willow Datanode?

Ideall, the Willow software suite is not supposed to fail, nor is the workstation, of course. Nevertheless, by design of the system, the WillowGUI and daemon can be killed without affecting the Willow Datanode operation. For example, recording will continue even as programs are killed. In fact, the workstation can be (if needed) re-booted without affecting the Datanode. After re-boot, simply start WillowGUI to launch daemon and re-connect to Willow DataNode.

4.6.4 Electrical noise is higher than expected.

Experiences electrophysiologists know that electrical noise can arise from any of a number of possible causes. The first thing to check is whether or not the Willow system is grounded. All Willow Datanodes come with a floating 12VDC power supply, and are thus *not grounded*. This configuration allows the experimenter to chose how the Willow system is grounded, e.g. close to the animal, for best signal quality.

However, make no assumption about whether the DC supply from any other power supply is connected to ground or not. Always measure the impedance from DC to AC on any power supply before using. It could be that the power supply has no connection to ground (so floating DC output), a sub-ohm (short-circuit) to ground, or some intermediate impedance to ground

Why should Willow users care about power supplies and connectivity to ground? If using a floating supply, then a ground connection must be added to headstages and/or sample to reduce noise. On the other hand, adding a ground connection when using a grounded power supply creates a ground loop which may increase noise. And power supplies with any impedance other than “zero or infinite” are unlikely to achieve low noise levels and should be avoided.o

4.6.5 No LEDs are lit on headstages

Check that the MISO/MOSI cables are correctly configured MISO to MISO and MOSI to MOSI. Accidentally connecting MOSI to MISO and vice versa will not damage the system, but neither will it work properly until properly connected.

4.6.6 Are headstages hot-swappable?

With the Willow Datanode powered-on, can headstages be dis/connected to the Datanode? No. Not only is there a risk of damage to sensitive circuitry caused electrical current transients, but also any newly-connected headstages will be uninitialized. Proper initialization of the neural chips on the headstages happens within 15 seconds of powering-on the Datanode only. Any headstages that are either powered-on or connected after Datanode boot will exist in an undefined state.

4.6.7 Network IP address conflicts

The Willow local area network domain is hard-coded in the FPGA on the Datanode to be 192.168.1.XXX. However, this same network domain is not uncommon for LANs. Thus, there is the potential for IP address conflict. Please contact us if your Willow Datanode needs a different IP address.

4.6.8 Are there indicator LEDs on Willow Datanode?

Yes, there are indicator LEDs on the Datanode PCB but the enclosure makes it all but impossible to see them. Since the Willow does not come with a warranty, feel free to remove the 8 screws holding the enclosure top to reveal the PCB and LEDs.

1. Green LED - located opposite corner of the board from the power connection - lit when 12 VDC power is supplied.

2. Blue LED - located on red microcontroller board - blinks when microcontroller is functioning correctly

3. Orange LEDs - located WHERE? - one LED indicates SOMETHING about user-controlled GPIO, and the other LED indicates an error in the FPGA logic

In addition, the TCP module (ADD PHOTO) has two (yellow and green) LEDs visible without removing the enclosure top.

4.6.9 The orange LED on the Datanode PCB is lit.

Whether due to accident, unexpected conditions, or undiscovered bugs in the system, an error condition may arise on the data node. This will be indicated by a solid orange LED staying lit near the USB connector (between the blue and green LEDs). There is a secondary orange LED which is user controlled GPIO and should not be confused with the error LED. Otherwise, an error flag has gone high somewhere in the FPGA logic, and needs to be cleared. The WillowGUI currently does not have a simple mechanism for clearing an error state in the hardware, so it is best to reset the FPGA by cycling the power.

4.6.10 The data node is not functional after powering on.

Just after power-on, the FPGA needs to self-program ("configure") from a relatively slow flash memory chip; this process takes more than 10 seconds, during which the only indicator lights will be the microcontroller board and the green power LED (on the opposite corner of the board from the power switch).

If the green power LED does not come on when the power switch is in the On position, check that the power supply is the correct voltage (12v DC) and plugged in, and check for any shorts of the external power connectors.

If the green power LED does come on, but none of the other status LEDs on the board turn on (there should at least be a blinking blue LED near the set of buttons), even after waiting 30 seconds, it is possible that the FPGA configuration has been lost or than an earlier upload failed, and the configuration file needs to be re-uploaded over the USB port. This condition has never been observed.

4.6.11 WillowGUI cannot connect to data node.

Check network configuration on the host (must have static addressing configured and be on the same subnet as the datanode).

Check that the microcontroller has not frozen (reset it if the blue LED is not blinking on the red microcontroller board).

Check that the TCP module is powered up and connected to the network. Most ethernet switches will have a "link up" LED for each port; check that the port connected to the TCP module is active.

Note that the TCP module and microcontroller may take up to 10 seconds to respond to requests after a power on or microcontroller reset.

If both LEDs (yellow and green) on the TCP module are blinking at the same time, there has been a failure of communication between the module and microcontroller. Try resetting the microcontroller several times, or in the worst case power down the whole data node for 60 seconds before re-trying (and also notify LeafLabs of the problem!).

4.6.12 Channel data is all zeros.

This may be due to a headstage becoming disconnected or powered down. Valid data for a connected channel should never be all zero; the floating "zero voltage" should be around 32768 (half of 2^16) and have at least a couple bits (~5-10 counts) of noise even when the probe is grounded or tied to a stable voltage source. Check the "chip alive" metadata bits for the corresponding channels.

If reading back from disk, check that the experiment cookie is as expected (you may be reading back old empty data).

4.6.13 Headstage chips initialize correctly (blinking white LEDs), but the data node reports that headstage chips are not alive.

This condition can be caused by longer or shorter than normal cables between the datanode and headstages, resulting in valid commands sent to the headstages, but mangled data in the replies. Compensation is possible using FPGA compile-time flags; contact LeafLabs for details. The expected HDMI cable length is about 1.5 meters.

CHAPTER

FIVE

WILLOW SOFTWARE USER MANUAL

Contents

5.1 System Overview

The Willow system is comprised of both hardware and software components. On the hardware side, there is the Datanode, which is the main data acquisition and storage computer. The Datanode collects data from the headstages, which are modular amplifier and digitizer boards. Each headstage acquires from 128 channels, and each Datanode can collect from as many as 8 headstages. The Datanode receives commands from the workstation - a computer running Linux. The Datanode and the workstation communicate over a high-speed Ethernet connection.

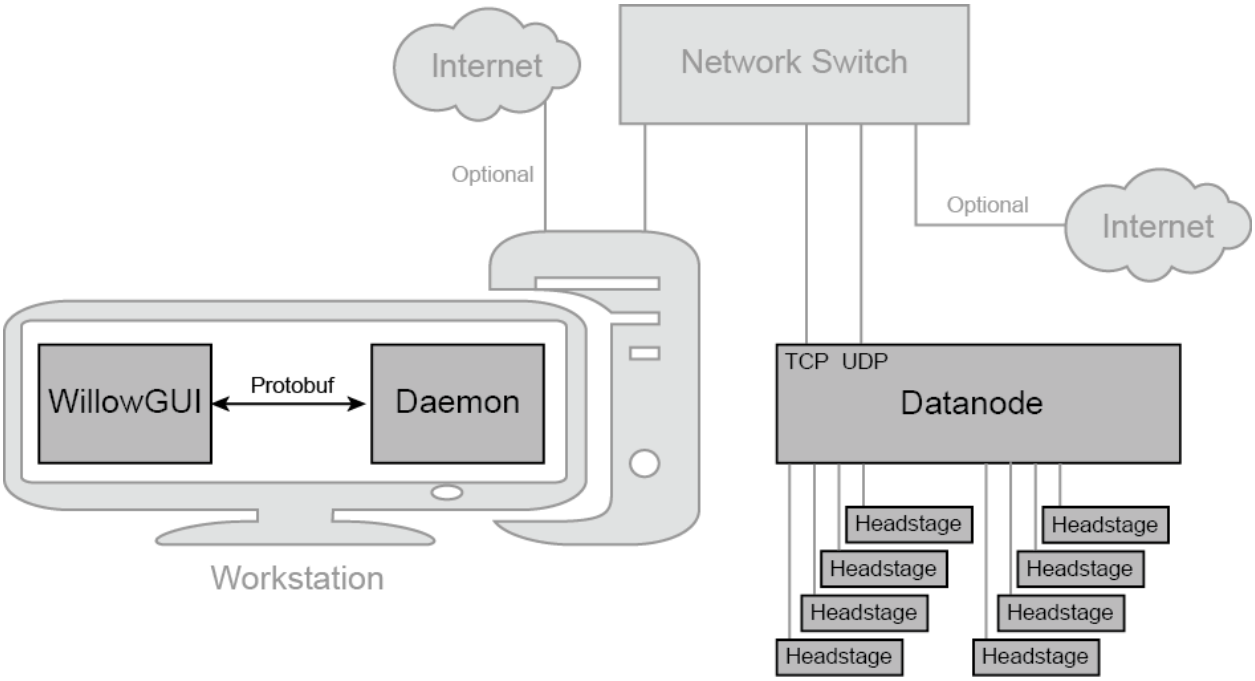
The workstation itself runs a multi-layered software stack. The *Daemon* handles the low-level communication with the Datanode, data collection, routing, and I/O. The daemon exposes a control interface based on Google's Protocol Buffers (protobuf). The *WillowGUI* software wraps this control interface in some higher-level functions that allow the user to record, stream, an analyze data with the convenience of a graphical user interface.

Note: As of Willow 2.0, Linux is the only supported operating system.

5.2 Daemon

5.2.1 Install

Skip to the *Network Configuration* section if the Willow system is pre-configured from LeafLabs or the workstation has already been setup for Willow.



Note: This guide assumes an apt-based Linux distribution (Debian, Ubuntu, Mint, etc.) with standard packages like git and Python 2.7 installed. WillowGUI has only been tested on Ubuntu 16.04.

First, download the daemon source code:

```
$ git clone https://github.com/leaflabs/willow-daemon.git
$ cd willow-daemon
```

Open *willow-daemon/README.txt*, and install all mandatory dependencies described therein. Then, compile the daemon using *scons*:

```
$ cd willow-daemon
$ scons EXTRA_FLAGS='-O2'
```

The resulting executable should be in *build/*.

5.2.2 Network Configuration

A pre-configured workstation from LeafLabs has two NICs - one built-in to the motherboard (labelled *Internet*) and a second as a PCI card (labelled *Willow*) - to allow internet access while communicating with the Willow datanode. Each NIC is configured for one purpose or the other but not both. Thus, it is important to establish correct connectivity by following the labels.

In case the labels are missing or illegible, correct connectivity can still be determined. To begin, in the case of reversed connectivity, WillowGUI will display no connectivity to DataNode (even though the datanode can be pinged at 192.168.1.2) and no IP address will be issued by DHCP to NIC intended for internet (as seen by running *ipconfig* at a command prompt). To restore proper function, swap the ethernet cables connected to the workstation. WillowGUI will display no connectivity to Datanode (see Figure 5.1.) even though the datanode can be pinged at 192.168.1.2, and no IP address will be issued by DHCP to NIC intended for internet (as seen by running *ipconfig* at a command prompt). To restore proper function, swap the ethernet cables connected to the workstation.

To support high data rate network traffic, it is important configure the size in bytes of network packets. If you're using a pre-configured Willow workstation, this happens automatically upon startup. In the case of a non-pre-configured workstation, a utility script is included with the *Daemon* in *willow-daemon/util*:

```
$ cd leafysd/util
$ ./expand_eth_buffers.sh <interface>
```

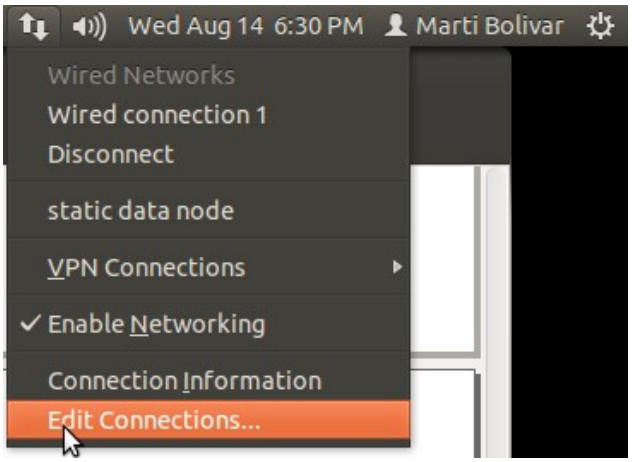
where <*interface*> is the name of your network interface (default is eth0).

Note: This script requires sudo privileges. See *Login to the Workstation*.

On a pre-configured Willow workstation, the network configuration is already set. Alternatively, to manually set up the network interface for Willow, start Network Connection Editor, either from a terminal:

```
$ nm-connection editor
```

or (on Ubuntu) by clicking "Edit Connections" in the Network Connections dropdown in the menu bar. Add a new Ethernet connection, and call it "Willow". In the IPv4 Settings tab, set the "Method" to "Manual", then add a new address with Address=192.168.1.98, Netmask=255.255.0.0, and Gateway=192.168.1.98. Click "Save", and close the Connection Editor.

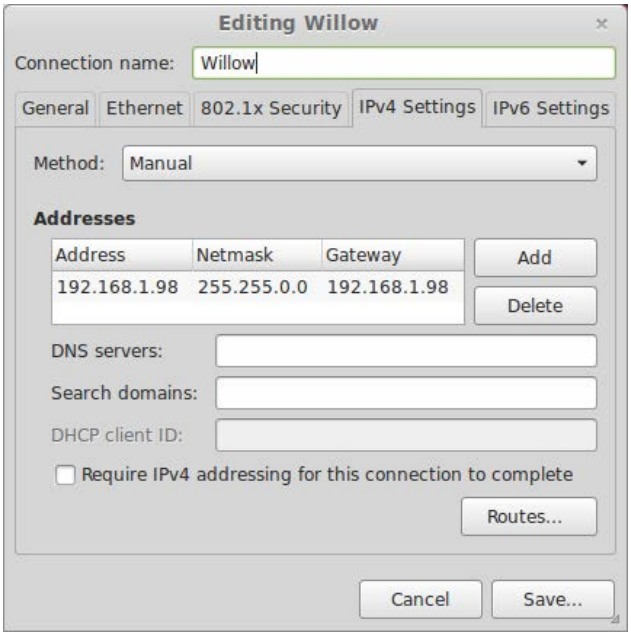


Note: The Willow local area network domain is hard-coded in the FPGA on the Datanode to be 192.168.1.XXX. However, this same network domain is not uncommon for LANs. Thus, there is the potential for IP address conflict.

5.3 WillowGUI

5.3.1 Introduction

This is the user guide for the Willow GUI software: a desktop application for interacting with Willow, a 1024-channel electrophysiology system developed by LeafLabs in collaboration with MIT's Synthetic Neurobiology Group (SNG). This document, contains information for end-users. Developers should check out the developer's guide.



5.3.2 System Overview

The Willow board communicates over TCP and UDP connections with a workstation computer running a daemon server called *willow-daemon*. All interactions with the hardware go through the daemon, which exposes an interface based on Google's Protocol Buffers (protobuf). The daemon repository includes command-line utilities for sending and receiving protobuf messages to and from the daemon. The GUI was designed to replace the command-line workflow with a more accessible and streamlined graphical approach.

This setup guide assumes you have already installed the daemon and configured your network; if not, refer to the Willow 2.0 User Guide for instructions on how to do this.

5.3.3 Install

To access the GUI repository, clone it from GitHub:

```
$ git clone https://github.com/leaflabs/willowgui.git
```

This will create in the current directory a local copy of the repository in a folder named *willowgui*. Before running the GUI, you'll need to install these mandatory dependencies:

```
$ sudo apt-get install python-numpy python-matplotlib python-qt4 python-tk \
      python-git python-scipy python-pyqtgraph python-h5py hdfview
```

5.3.4 GUI Startup

On a pre-configured Willow workstation, you can start WillowGUI by clicking the WillowGUI icon on the Unity Launcher side panel.

Alternatively, you can start WillowGUI from the top-level folder of the local repository (e.g. *willowgui*) with

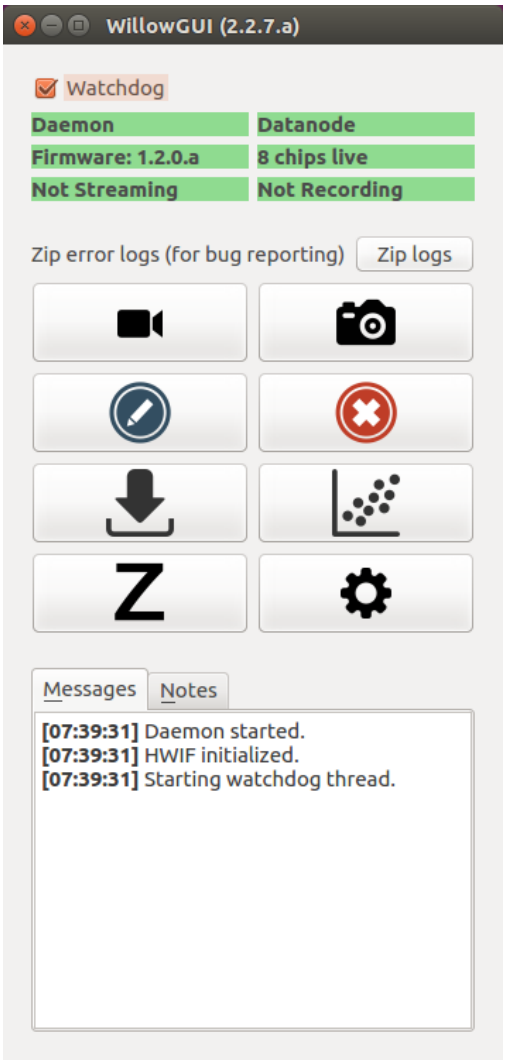
```
$ src/main.py
```


Willow GUI


WillowGUI Configuration Wizard
WillowGUI Configuration Wizard
WillowGUI has detected that config.json is missing. Probably this means that you are starting the application for the first time on this system. This wizard will help you configure your setup. You only need to do this once per installation.
< Back
Next >
Cancel

The first time you run the GUI, a Configuration Wizard will appear. Follow the instructions and click Finish.

If the setup was successful, the WillowGUI control panel will appear on the screen.



5.3.5 Usage

The Willow Control Panel (pictured above) consists of the status bar, the button panel, and the message log. The daemon is automatically started in the background, with *stderr* and *stdout* redirected to *log/eFile* and *log/oFile*, respectively. The message log should display a line indicating that the daemon was started successfully.

Status Bar

The status bar is an array of labels at the top of the main window, which indicates the status of the daemon and the hardware. The status bar is updated by a watchdog process which pings the Datanode once every second, and evaluates the response to determine daemon status, Datanode status, firmware version, hardware (i.e. neural chip) status, streaming state, and recording state. The general color code for the labels is: green for "good/normal state", orange for "bad state, something is wrong", and gray for "unknown state". For example, if the Datanode is unresponsive, its

label will turn orange and the labels below it will turn gray to indicate an unknown state. Similarly, if the daemon is unresponsive, its label will turn orange and all labels below it will turn gray. These conditions are shown below.

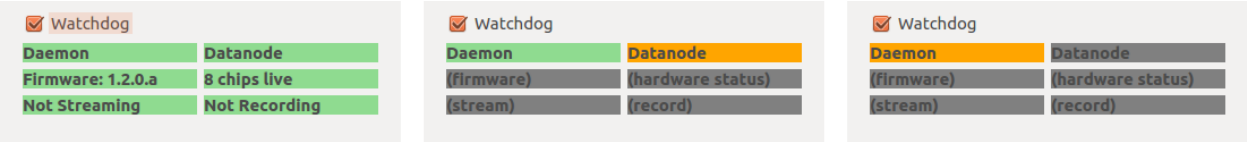
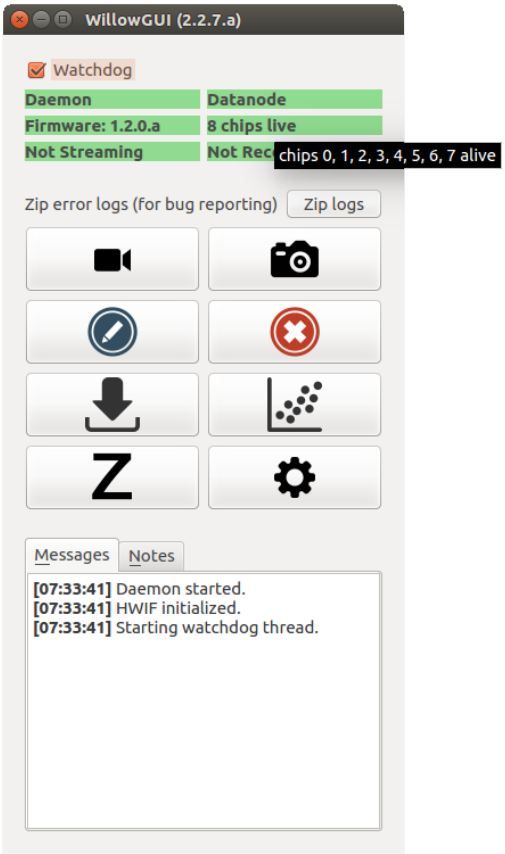


Figure5.1.: Status bar in three states: Daemon up and connected to Datanode, showing firmware version, number of chips alive, and streaming/recording states (*left*); Daemon up but Datanode unresponsive (*center*); Daemon down or unreachable, hardware state unknown (*right*).

The firmware label, when green, will show the firmware version as a 32-bit hexadecimal number. The hardware status label beside it indicates the number of active neural chips that were found. Hovering over the label will display the id's (0-based) of the active chips. For example, MISO/MOSI1 corresponds to chips 0,1,2,3; MISO/MOSI2 corresponds to active chips 4,5,6,7; etc. Note chips are numbered from left to right on the headstage as viewed in Figure 3.2..



Note: Hardware-related bug reports can be sent to info@leaflabs.com, and should include both the firmware version and the error state bitmask as context.

The *Stream* label indicates the streaming state of the Datanode: green for idle, and blue for streaming. Similarly, the *Record* label beside it is green for idle, but turns red while recording and additionally indicates the current disk usage as a percentage in real time. These states — streaming and recording — are determined from the hardware registers on the Datanode, and are therefore robust to daemon and/or GUI crashes.

The Watchdog process can be disabled by un-checking the *Watchdog* box at the top of the status bar. This should rarely be necessary, but may free up some bandwidth during streaming or transfer operations on slow workstations. The status bar will no longer be updated while the Watchdog is disabled.

Message Log

At the bottom of the main window is the message log, which displays and logs messages from the GUI. Examples of messages include success/failure reports from requested actions, status updates from background threads, and general errors. Each message is prepended with a timestamp of the form *HH:mm:ss*. The message log can be cleared, or saved to a log file using the buttons at the top-right. Message logs are saved to the *log/* subdirectory by default, but can be saved anywhere by navigating the presented file dialog.

The message log is intended to serve as a supplement to the experimentalist's lab notebook, logging the actions taken — and their results — to help contextualize the data for future analysis.

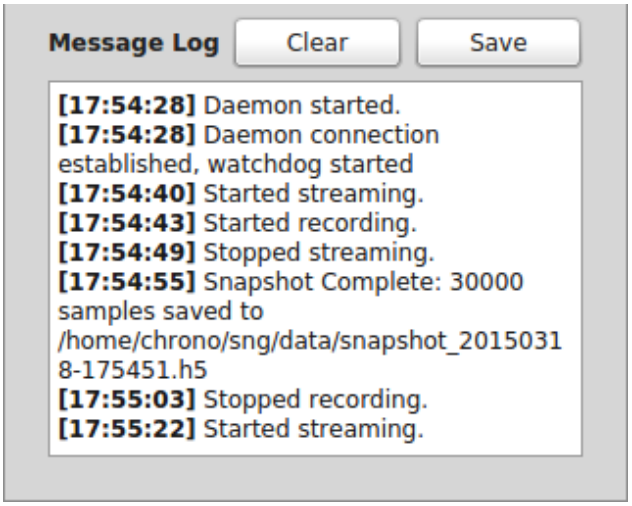


Figure5.2.: The message log, showing a list of actions taken, with timestamps.

Button Panel

At the center of the main GUI window is the button panel, containing 8 buttons with symbolic icons. These buttons are your starting point for all the capabilities offered by the GUI. Mousing over a button will display a tooltip which describes its action.

Streaming

UPDATE

Snapshots

UPDATE

Recording

Recording is the acquisition of data directly to the SATA disk. It is the only acquisition mode that is guaranteed to be robust, and is intended for scientific experiments. To start a recording, click the *Start Recording* button in the Button Panel. The corresponding label in the status bar will turn from green to red, and the label will begin displaying the disk usage. Accurate reporting of disk usage is depending on the *Datanode Storage Capacity* parameter in your settings. To stop recording, click the *Stop Recording* button.

Warning: Every time you click *Start*, the system will start recording from the beginning of the disk, over-writing any previous experiments you may have recorded. In this sense, the disk acts as a temporary (non-volatile) buffer for experimental data. If you've just recorded an experiment and want to preserve the data, it is recommended that you transfer the data over to your workstation immediately after stopping recording. If storage space is an issue on your workstation, an alternative is to remove the SATA disk and label it, thus repurposing the disk buffer as an archival system.

Note: Filling the SSD with data to 100% capacity (as defined by Datanode Storage Capacity in *Settings*) is allowed. In this case, WillowGUI will briefly indicate that 100% capacity was achieved (REF FIGURE), will report the final sample index (THAT WAS LAST WRITTEN?), and will stop the recording. Importantly, the criterion for stopping the recording is that Datanode Storage Capacity was reached, not necessarily that the SSD actually is full. Thus, normal operation of WillowGUI is expected when Datanode Storage Capacity is less than or equal to the actual capacity of the SSD. For details on what to expect when SSD capacity is less than Datanode Storage Capacity, see *Troubleshooting*.

Transferring Experiments

ADD CONTENT.

Experiments cannot be transferred while recording. The recording must be stopped before the experimental data can be transferred.

Plotting Data

ADD CONTENT.

Impedance Testing

The Intan headstages offer impedance testing functionality which can be accessed from the GUI. Click the *Run Impedance Test* button on the button panel — this will present a dialog where you can choose to measure the impedance of a single channel, or of all channels in sequence. If you choose *Single Channel*, the process will take about 10 seconds, and the result will be posted on the message log. If you choose *All Channels*, the loop will run for about 3 minutes, and the result will be saved in a time-stamped *.npy* file, also reported in the message log. In *All Channels* mode, you also have the option to *Plot When Finished*, which will bring up a plot window showing the impedance results across all channels (see Figure 5.3.). All impedance measurements are taken at f = 1/kHz.

Note: Impedance cannot be measured while recording. The Intan chips used in our headstages simply do not allow this.

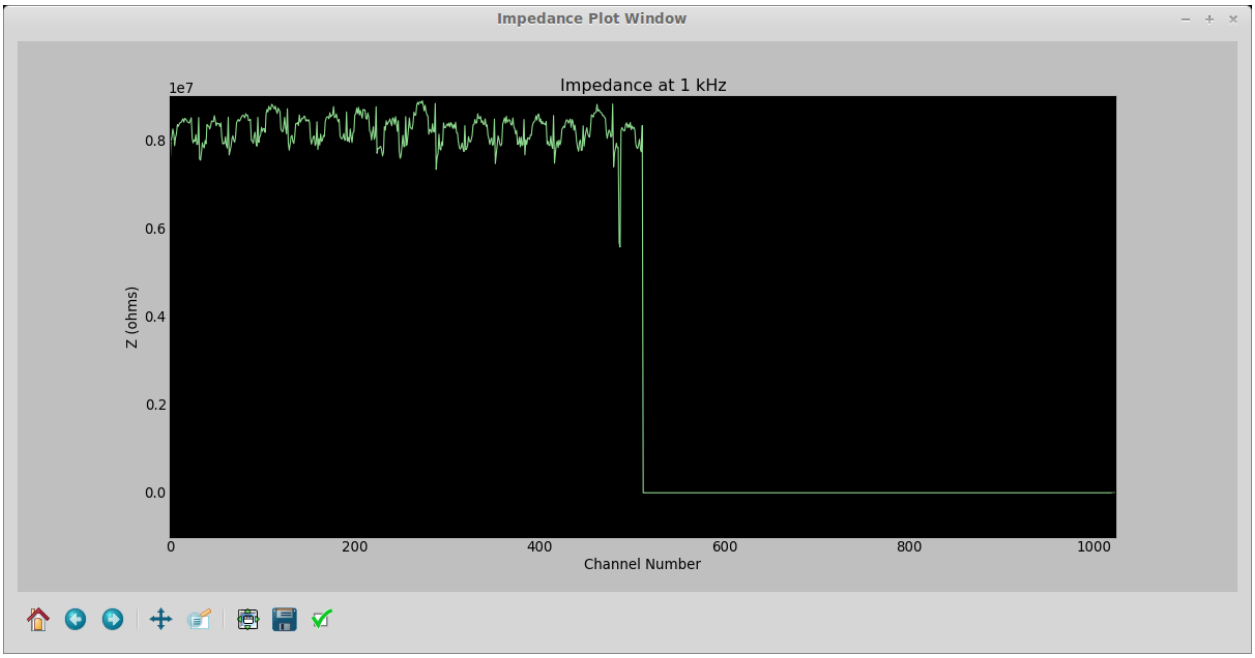


Figure5.3.: UPDATE FIGURE Impedance results from a measurement taken with 4 of 8 headstages bare headstages connected. As expected, channels 0-511 show an approximate, 32-channel periodicity from the PCB traces, and channels 512-1023 are dead.

Settings

The GUI preferences can be viewed and modified by clicking the *Configure Settings* button on the button panel. This will present a key-value dialog (see figure below) for the various configuration parameters. These values are stored in *src/config.json*, and persist across restarts.

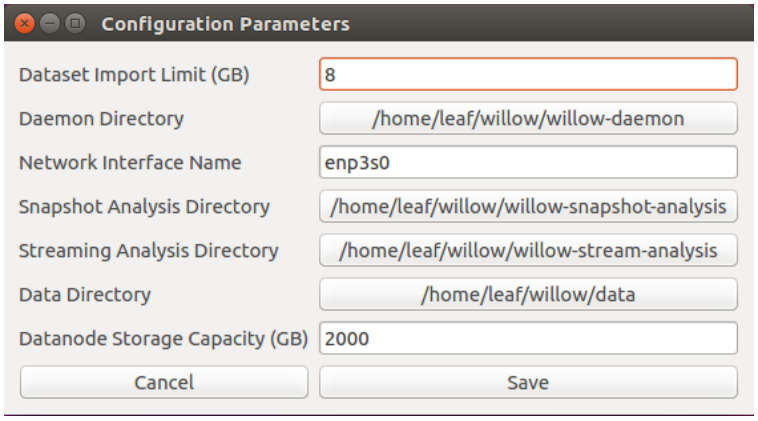


Figure5.4.: Settings window, showing default configuration parameters.

5.3.6 Troubleshooting

dffdsfgds

5.3.7 Working with Different Probe Geometries

dffdsfgds

5.4 Plugin Analysis Scripts

Describe Plugin Architecture of WillowGUI. Mention existing Plugin scripts.

5.4.1 Install

To access the snapshot analysis repository, clone it from GitHub:

```
$ git clone https://github.com/leaflabs/willow-snapshot-analysis.git
```

Likewise, to access the stream analysis repository, clone it from GitHub:

```
$ git clone https://github.com/leaflabs/willow-stream-analysis.git
```

ADD NOTE ABOUT DEPENDENCIES, IF ANY.

5.4.2 For Each Script, Walk through Functionality

dffdsfgds

5.4.3 Adding User Scripts

To add an analysis program to WillowGUI, create a new folder in the top level and put inside of it an executable named 'main'. This executable can be anything - a MATLAB script, a Python GUI, a compiled program, etc. The only requirement is that is take as its first command-line argument the absolute filename of the snapshot it is to analyze.

To make this work with WillowGUI, set the *Snapshot Analysis Directory* parameter in the Config Wizard, or else in the Settings menu during runtime. Then, any subdirectory containing a 'main' executable will appear as an option for custom analysis in the Snapshot Dialog. WillowGUI will launch the executable as a subprocess, with its working directory set to that of the executable - so metadata (e.g. probe mappings) or library files can be kept in the same folder alongside the exectuable. stdout and stderr will be piped to 'oFile' and 'eFile' in the working directory, so avoid using these filenames for metadata, as they will get overwritten upon execution.

Place code or data that you'd like to re-use across different analysis programs in the lib/ directory. Then programs can access this code or data using relative pathnames, e.g. in Python:

```
sys.path.append('../lib/py')
```

Stay D.R.Y.; Don't Repeat Yourself!

5.4.4 Working with Different Probe Geometries

dffdsfgds

5.5 Working with Data

The data on the SSDs is stored in raw form without a filesystem. This design choice simplifies the programming of the FPGA, but obviously is inconvenient and may change in future versions of Willow.

To read data off the SSD and into a file on a PC, the user has two options...

5.5.1 Retrieve Data from SSD Using WillowGUI

Immediately after an experiment ends (and before the Willow Datanode has been powered down!), data can be pulled from the SSD to the laptop (or whatever computer is controlling the Willow) using the WillowGUI.

1. From WillowGUI with Datanode connected, click *Transfer* (down-pointed arrow)
2. Under *Experiment* click *Subset* and select the time window, in seconds from start of recording

---

Note: After the Willow Datanode has been power cycled, the WillowGUI cannot be used retrieve data from an SSD. WHY NOT?!

5.5.2 Retrieve Data from SSD Using SATA-to-USB cable

Data can be pulled from SSDs at any time using the *sata2hdf5* program that comes installed with the *Daemon* software. If not already done, first *Install* the daemon software. Then, using a SATA-to-USB adapter (one is included with every Willow system) attach to the PC one SSD, by first powering down the Willow Datanode and ejecting the SSD.

Figure5.5.: SATA-to-USB adapter with SSD attached.

Note: To ensure that the PC operating system quickly registers the SSD, connect the SSD to adapter first, then connect adapter to USB connector on PC.

Then,

1. Determine the device name assigned to USB-to-SATA adapter by running *ls /dev/sd*‘* before and after attaching the adapter. The new device name is the desired one.

2. Determine the start time of data (recording on SSD), i.e. the UNIX time on the laptop when recording started.

Example:

```
$ sudo ~/src/willow-daemon/build/sata2hdf5 -c 30000 -o 0 /dev/sdd test.h5
Starting cookie: 0x57000886
Starting board id: 0x3746f441
Copied 30000...
Copied 30000 board samples.

0x57000886 = 1459619974 seconds since epoch = 2016-04-02 13:59:34+00:00 EST.
```

3. From snapshots (which have time embedded in filename), calculate the sample offset (which determines where in SSD to start extracting data). So, for data at 15:00 EST: sample offset = 15:00-14:00 = 60 minutes = 3600 sec = 3600*30e3 samples = 108000000.
4. Decide how many samples we want, e.g. 1 min = 60 sec*30e3 samples/sec = 1800000.
5. Finally, run sata2hdf5:

```
$ sudo ~/src/willow-daemon/build/sata2hdf5 -c 1800000 -o 108000000 /dev/sdd data_
↪offset_1hr_length_1min.h5
```

Note: At this time obtaining a subset of the channels (e.g. 64 out of 1024) is not possible unfortunately, neither through WillowGUI or from *sata2hdf5*.

5.5.3 Finding a Snapshot in a Continous Recording

The situation may arise that a snapshot taken during an experiment contains particularly interesting data, and we wish to find that snapshot in a large continuous recording. To do so, see section *Retrieve Data from SSD Using SATA-to-USB cable*.

5.5.4 HDF5 version dependency

WillowGUI compiles without difficulty in Ubuntu 16.04 which include libhdf5 packages at version 1.8.16. Note that older versions of libhdf5 (e.g. version 1.8.11 as ships in Ubuntu 14.04) cause a compilation failure for WillowGUI (unless the signature for H5D_scatter_func_t in H5Dpublic.h is modifies with the addition of a const qualifier to the first argument).

Furthermore, Ubuntu 17.04 ships with version 1.10.0 of libhdf5. This is a major release jump, and this page: https://support.hdfgroup.org/HDF5/ warns that although HDF5-1.10 will read files created with earlier major releases, older releases such as HDF5-1.8 might not be able to read files created with HDF5-1.10. This could potentially cause headaches in the future, if e.g. a customer were to run sata2hdf5 on a system running Ubuntu 17.04, and then process the resulting file on a system running an older distro.

5.5.5 Opening HDF5 Files

The latest version of Ubuntu (16.04.2), which comes installed on any pre-configured workstation with the Willow system, includes HDFView for inspecting HDF5 files. Simply double-click on an HDF5 file to open the file and inspect its contents including all datasets and their attributes.

In addition, the WillowGUI can be used to interactively view HDF5 *Snapshots*.

HDF5 files in MATLAB

To open a snapshot HDF5 file in MATLAB, `plot_snapshot.m` is a script that accepts as input a filename and channels to plot. For example, to plot channel 128 from snapshot_20160402-161433_planetEarth.h5

```
plot_snapshot('snapshot_20160402-161433_planetEarth.h5',[128])
```

Generates as output

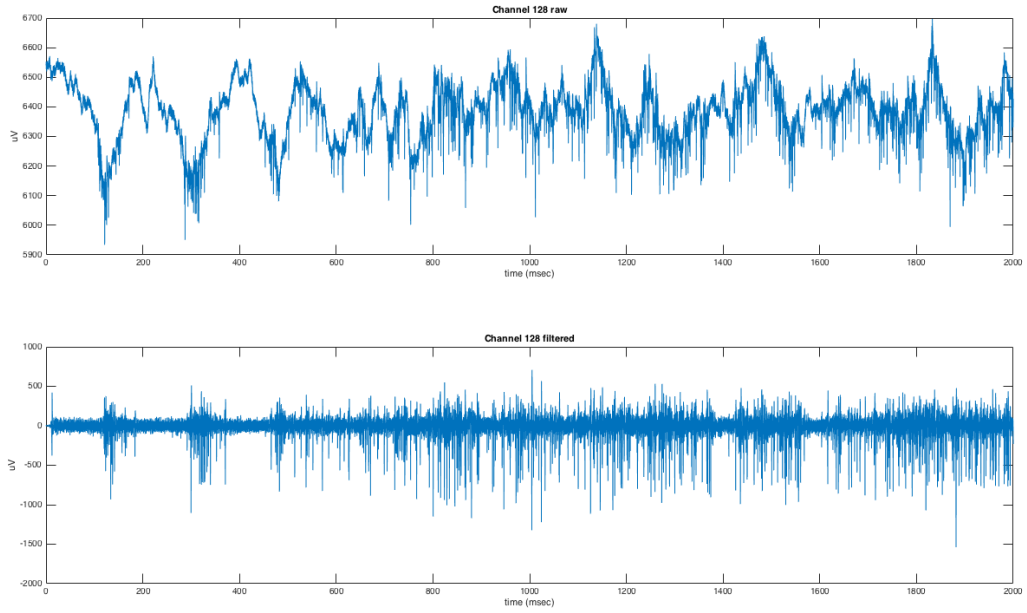


Figure5.6.: Plots of raw (top) and high-pass filtered (bottom) data for channel 128 in plot_snapshot('snapshot_20160402-161433_planetEarth.h5.

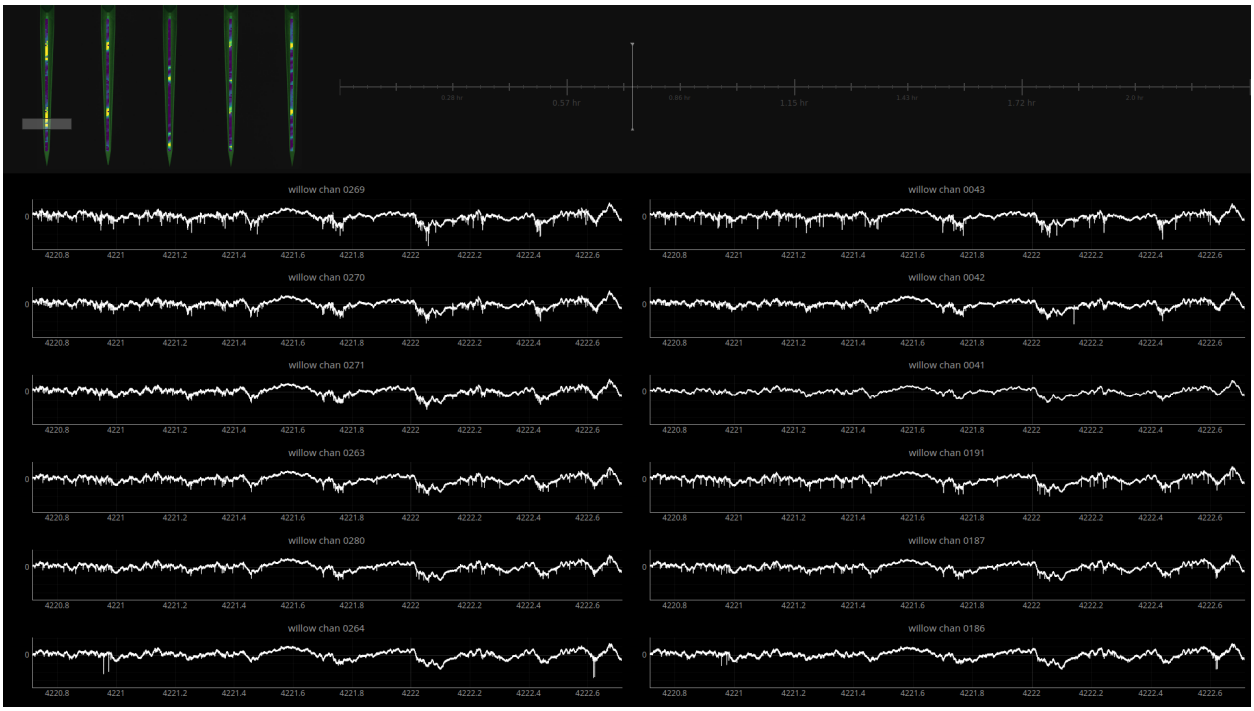
To open an impedance file, for example, `impedance_20160402-130530.h5`, in MATLAB run:

```
>> impedance = h5read('impedance_20160402-130530.h5','/impedanceMeasurements');
>> size(impedance)
ans =
      1024           1
```

Warning: In MATLAB, indexing of arrays is 1-based. In other words, the first element of any array (or along any dimension of a multi-dimensional array) is assigned an index of 1 and the last of N elements has an index of N. In contrast, the numbering and indexing of channels and chips in the Willow software suite is 0-based. This distinction is important when seeking to establish a correspondence between channels in Willow and other software packages such as MATLAB. Specifically, *channel j* in Willow is channel *j+1* in MATLAB.

Snapshots with Aspen WDX

To interactively explore 1024 channel data sets, Aspen WDX utilizes a client-server interface to enable researchers to explore large TB-scale data sets using a thin client interface.



5.5.6 Channel Maps

To preserve the geometrical arrangement of recording sites on the probe when plotting neural data, a channel map is needed that describes which shank, row and column each channel occupies. Channel maps can be found in the willow-snapshot-analysis repository. For a more detailed treatment of channel maps see the probe-channel-maps repository in gitlab (MOVE OUT OF GITLAB). The nomenclature used in these channel maps is defined in this schematic `(PDF)`:

5.6 Pre-configured Workstation

On a pre-configured workstation from LeafLabs, all willow-related software (the Willow Suite) is installed in *~/willow*.

By default, WillowGUI saves data in *~/willow/data*. Data can be saved to a different directory by configuring the *Settings*.

Documentation is contained in *~/willow/willow-docs*.

All other subfolders of the ~/willow folder contain one component of the Willow software suite. Software components include willow-gui, willow-daemon, willow-docs, willow-snapshot-analysis, and willow-stream-analysis. Each component is a clone of a github repository.

To update any software component, cd into the subfolder of component, and run:

```
git pull origin master
```

If git responds that there were merge conflicts, it means that changes have been made to source code. In that case, you must resolve the conflicts before proceeding.

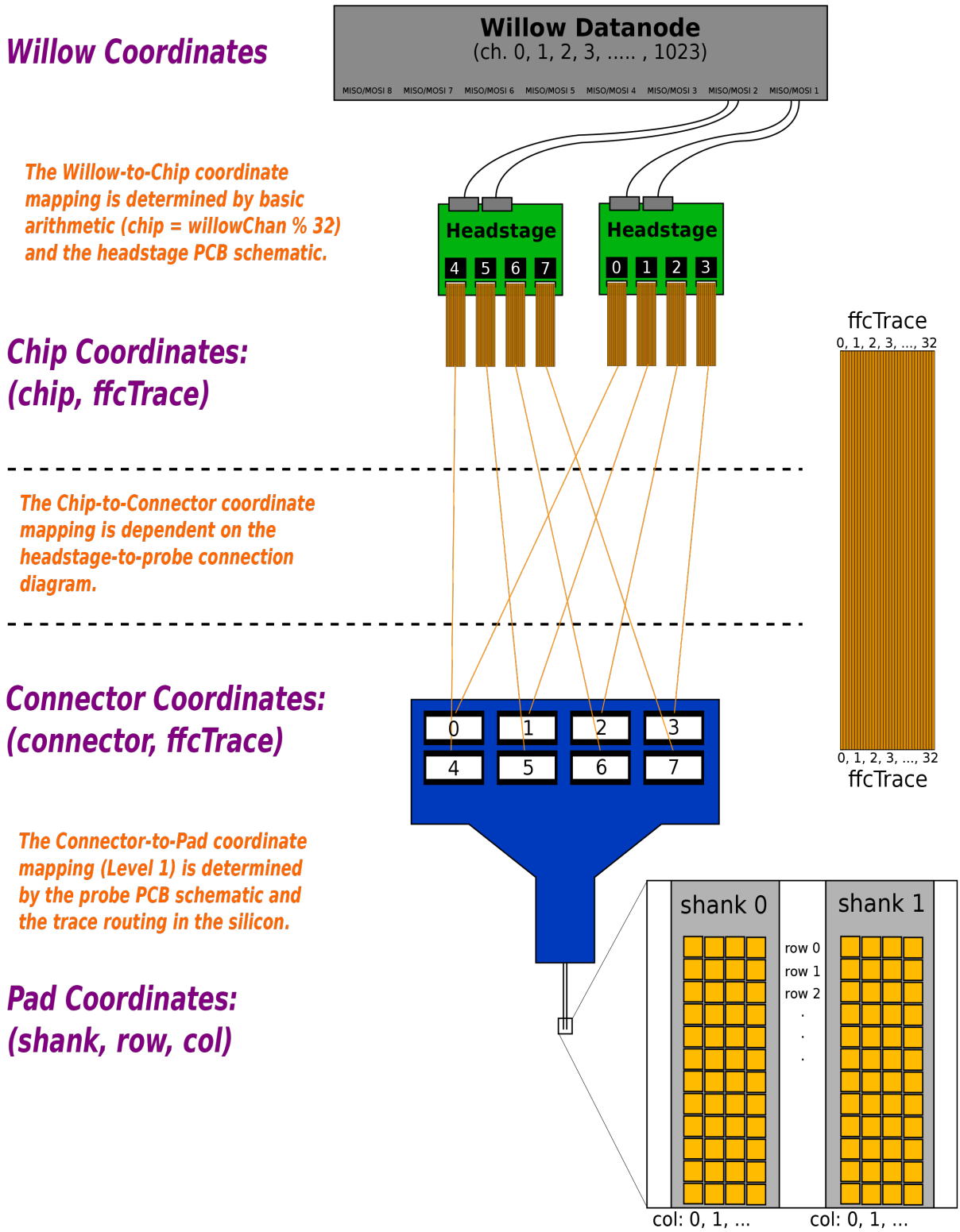


Figure5.7.: Nomenclature used to define channel maps.

5.7 Frequently Asked Questions

5.7.1 Is there a way to update all Willow-related git repos at once?

Currently, no. But it would not be hard to write a bash script to do so, assuming that there were no changes made to local copy of the source code and no merge conflicts to resolve.

CHAPTER

SIX

PROBES

Contents

6.1 Precautions

Warning: Probes are very delicate. When handling any probe, only touch the PCB board.

New users sometime break probes due to lack of handling/insertion experience. To gain experience without risking the most valuable probes, consider training first with low-quality probes (available upon request).

6.2 Performance Specifications

For a detailed explanation of the fabrication steps, physical dimensions, and electrical performance of our probes, see:

Scholvin, J., Kinney, J.P., Bernstein, J.G., Moore-Kochlacs, C., Kopell, N., Fonstad, C.G. and Boyden, E.S., 2016. Close-packed silicon microelectrodes for scalable spatially oversampled neural recording. IEEE Transactions on Biomedical Engineering, 63(1), pp.120-130.

6.3 Cleaning and Care

Adapted from NeuroNexus:

1. Store probes in their shipping box.

2. After withdrawal from tissue after an experiment, immediately rinse with a gentle stream of distilled water. Soak the probe (only the silicon shank; avoid soaking the epoxy or PCB) in a protein-dissolving detergent or enzyme such as contact lens solution or diluted surgical instrument detergent. This process may take a few hours.
3. After in vivo use, isopropyl alcohol (e.g. 70% IPA) can be used for cleaning AFTER the protein dissolving procedure (See Step 2).

Warning: Without first dissolving the residual tissue from the probe, alcohol could cause protein to stick to the electrode sites.

4. **DO NOT** use ultrasonic cleaners as this may cause damage.
5. **DO NOT** autoclave probes as this may cause damage.

6.4 Impedance Testing

The signal quality of each channel is determined in part by its impedance. To measure the impedance of individual channels in a probe, see *Impedance Testing*.

6.5 Rejuvination

With regular practice a neurosurgeon can reach a level of proficiency in handling and usage of probes so as to not break them and instead allow repeat experiments with a single probe. In this case, to preserve the low impedance of the individual recording sites and thereby the signal-to-noise ratio and quality of the data, it is important to follow the cleaning and care instructions described in *Cleaning and Care*.

Nevertheless, even with proper care, single-site impedance rises with repeated usage. It is unclear to us whether impedance is rising do to fouling (accumulation of foreign material on the sites), or loss or structural rearrangement of the plating material. Regardless, deep cleaning and re-electroplating may be necessary to lower impedance and re-establish excellent signal quality.

6.6 Channel maps

The geometrical arrangement on each probe of all channels is captured in a collection of *Channel Maps*.