WWU
MÜNSTER

Interactive USB measurement device controlling with Python

Benedikt Bieringer

DPG-Frühjahrstagung 2023

Online version with code: https://2xb.github.io/usb4research

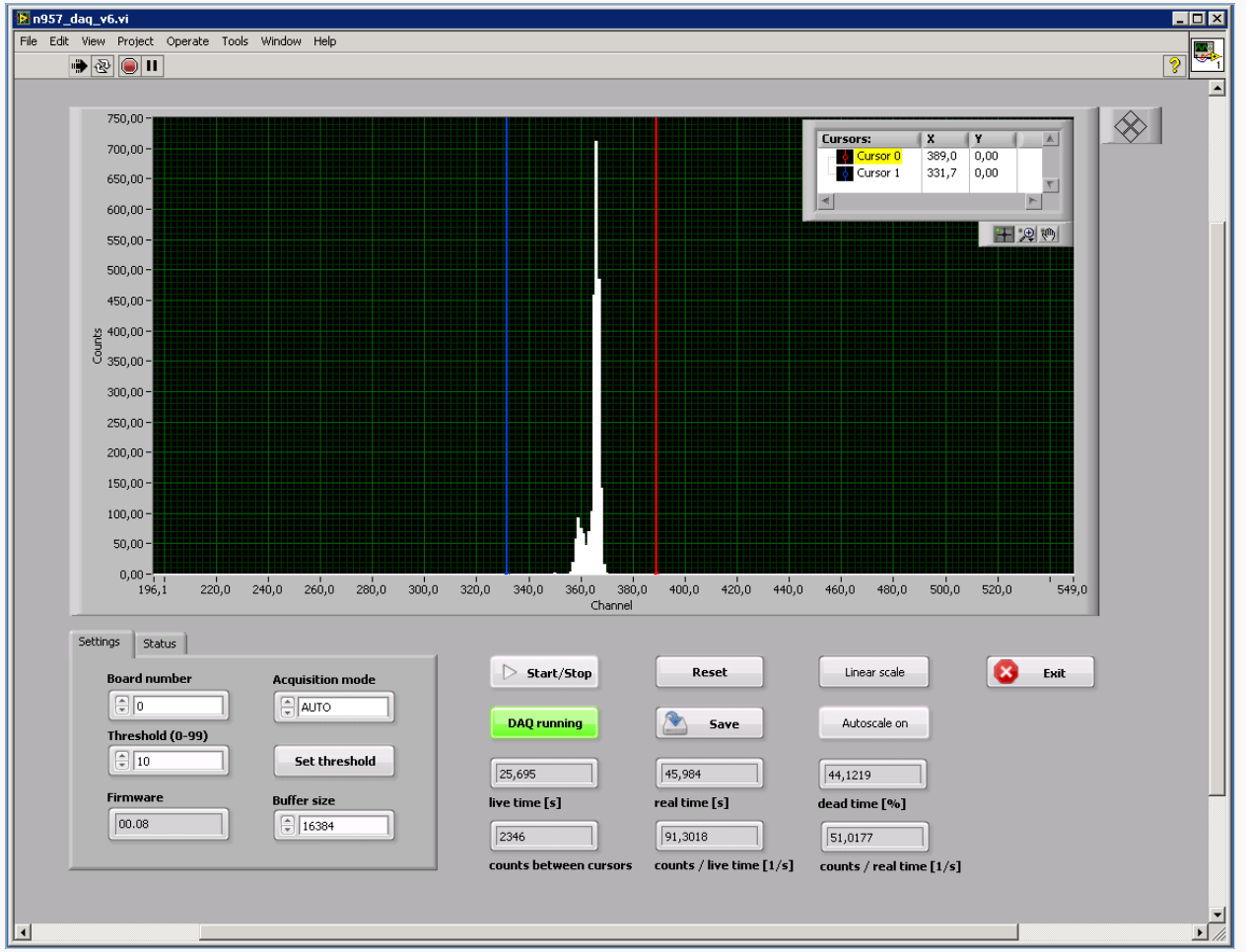
living.knowledge

Institut für Kernphysik

Outline

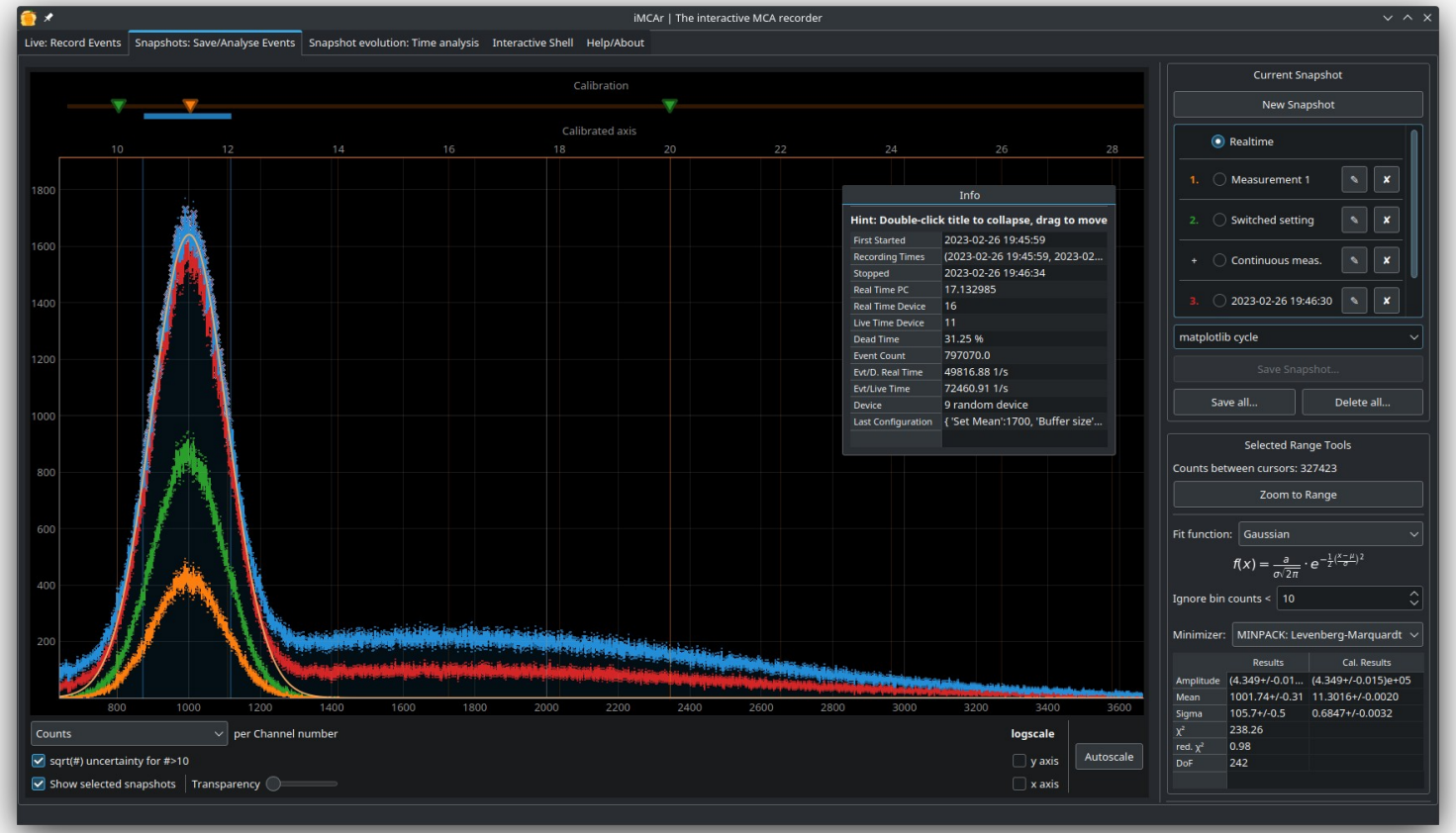
- **Introduction**
- How to write Graphical User Interfaces
- How to connect to devices – how manufacturers intend it
- How to connect to devices – writing own drivers

How it started...



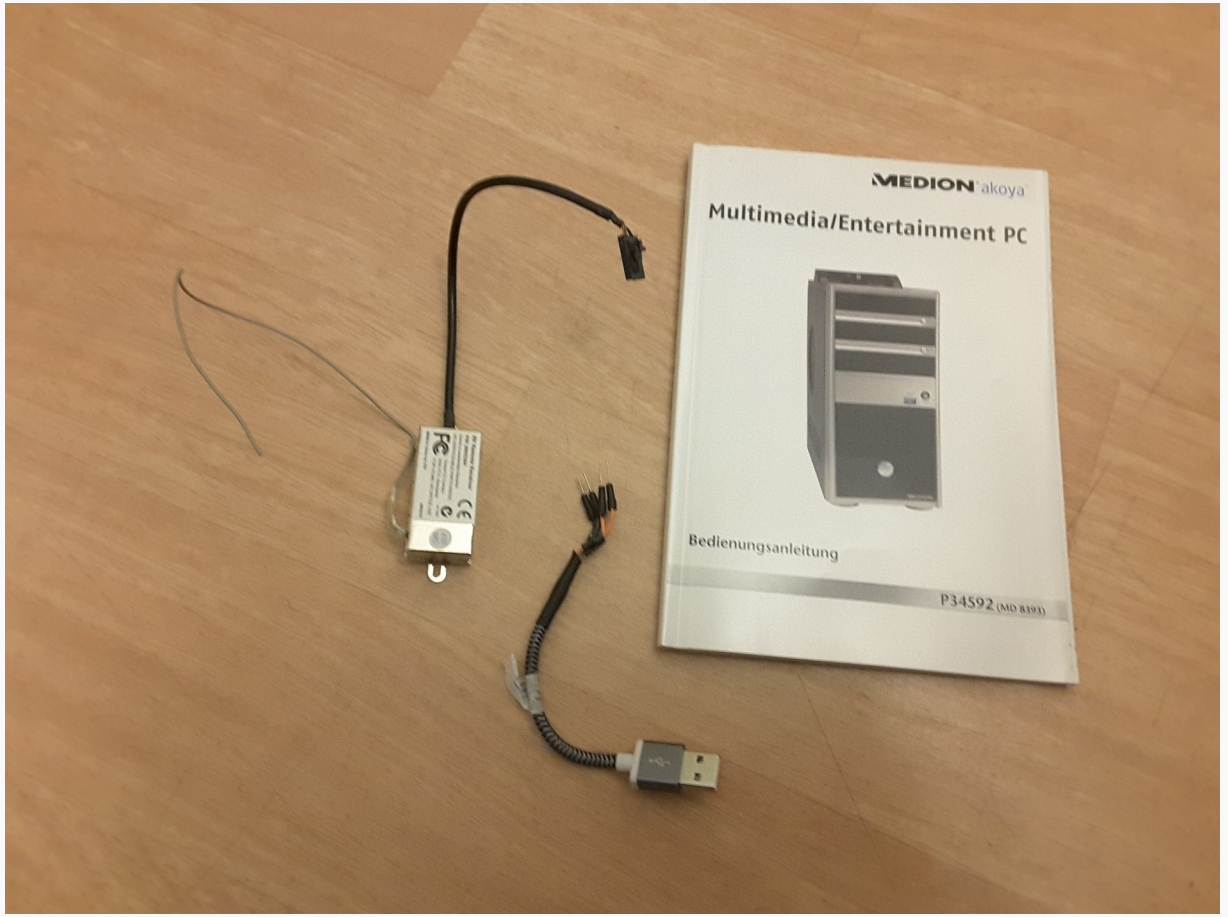
(Source: Volker Hannen)

Example: iMCAr, the interactive MCA recorder software



- Self-written GUI & drivers
- Supports:
  - CAEN N957 8K MCA
  - ORTEC EasyMCA 8K
  - ORTEC ADCAM 926
- Supports automatically repeated measurements
- Uses faultguard to avoid data loss (https://github.com/2xb/faultguard)

USB is everywhere



There's lots of software already

Don't re-invent the wheel (except for a good reason)!

- Look for already existing solutions
- For data storage & UI, Grafana + InfluxDB for time series enables users to combine data across devices
- Many USB devices already have generic drivers as part of multiple operating systems
  - Some libraries for such drivers exist (e.g. gphoto2 for cameras)

Outline

- Introduction
- **How to write Graphical User Interfaces**
- How to connect to devices – how manufacturers intend it
- How to connect to devices – writing own drivers

Graphical User Interface | It’s about users

Things to know about UI design:
Generally speaking:

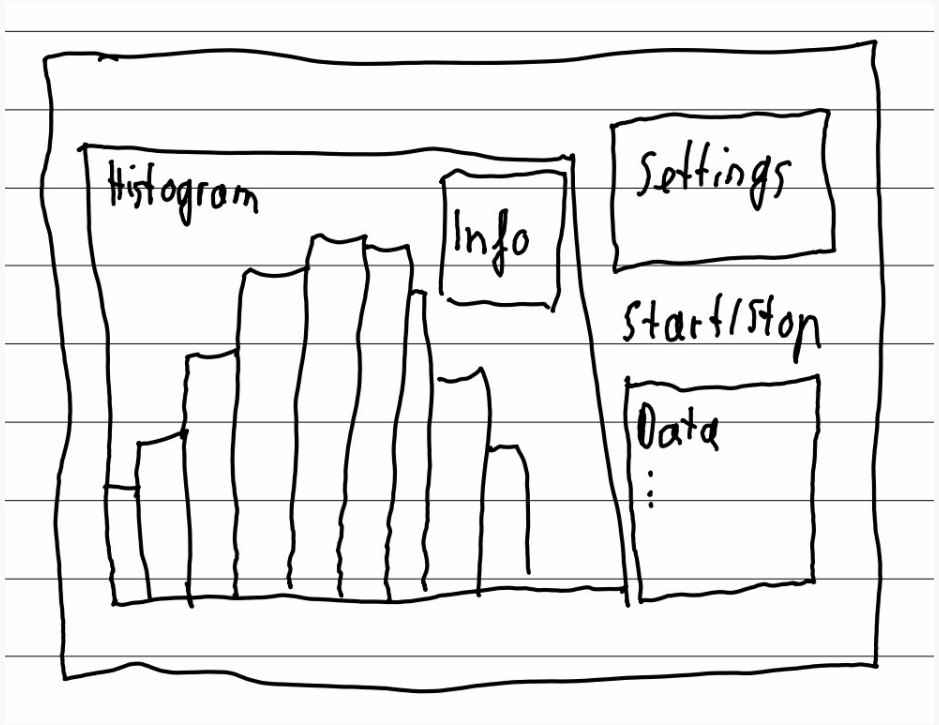
- Users combine features in unexpected ways *(that may break your code)*
  ⇒ **Work together with other users early**
- Users ignore unintuitive features
  ⇒ **Make everything as simple as possible**

General rule: You want users to think about their physics, not your software.

Graphical User Interface | Concept

1) Concept (sketch)

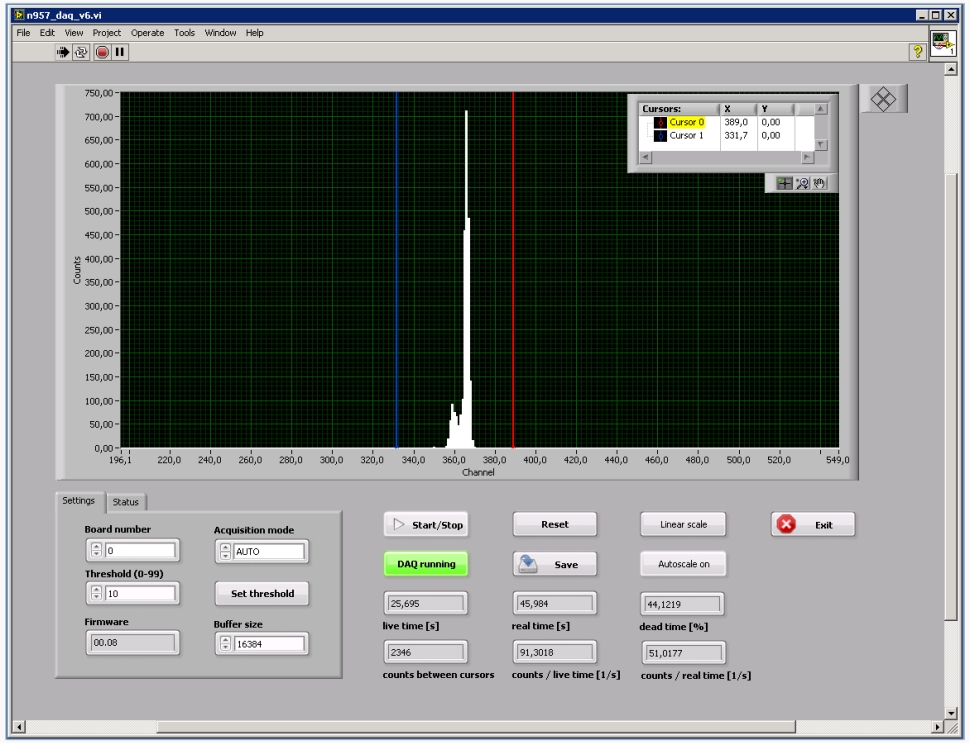
- Grab a paper, start sketching
- Show the sketch to colleagues/users, ask them how they'd intuitively use this software – be prepared for surprises!



Graphical User Interface | Rapid prototype

2) Rapid prototype

- First implementation is always dirty
- Might as well acknowledge that, get it running with compromises as quickly as possible
- Use e.g. LabView or libraries here that help you get results quickly, even if they have a lot of papercuts for end users
- Discuss the prototype with users!



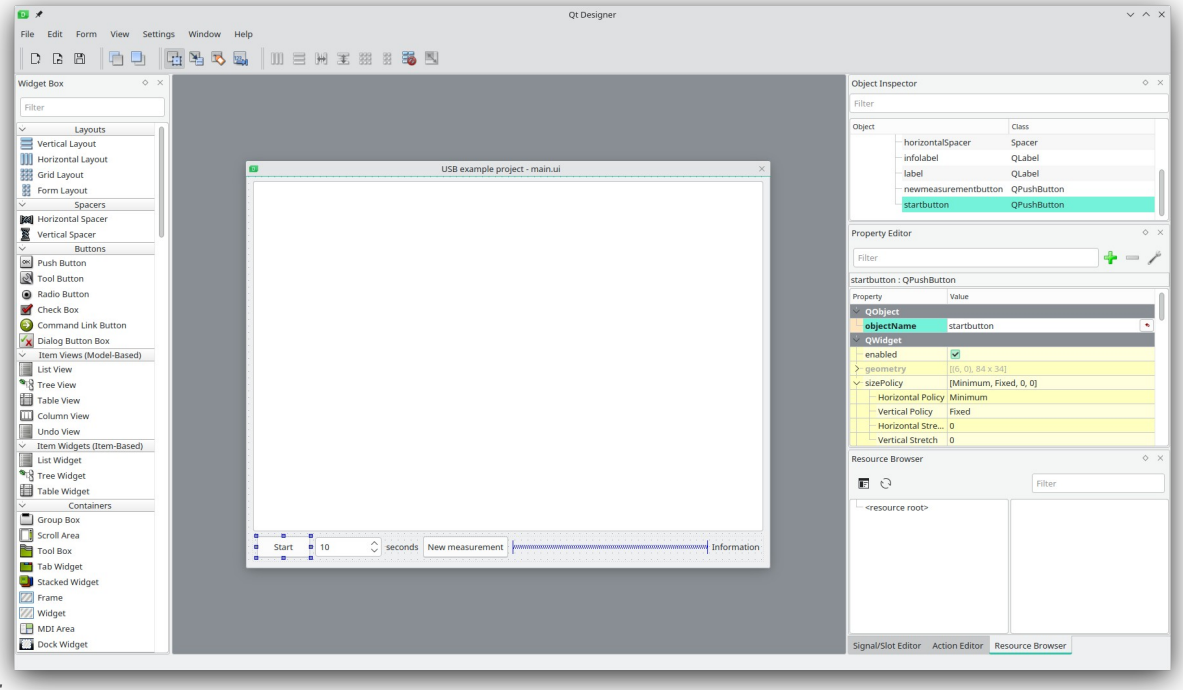
(Source: Volker Hannen)

Graphical User Interface | Implement

3) Implement – My go-to approach:

- Design UI in Qt Designer, save .ui file
- Use e.g. PyQt or PySide to load .ui file
  (PyQt & PySide differ in their license)
- Use PyQtGraph for interavtive plotting widgets
  Note: Never trust any plotting library without testing!

  Embed pyqtgraph plots using Qt Designer:
  https://pyqtgraph.readthedocs.io/en/latest/getting_started/how_to_use.html

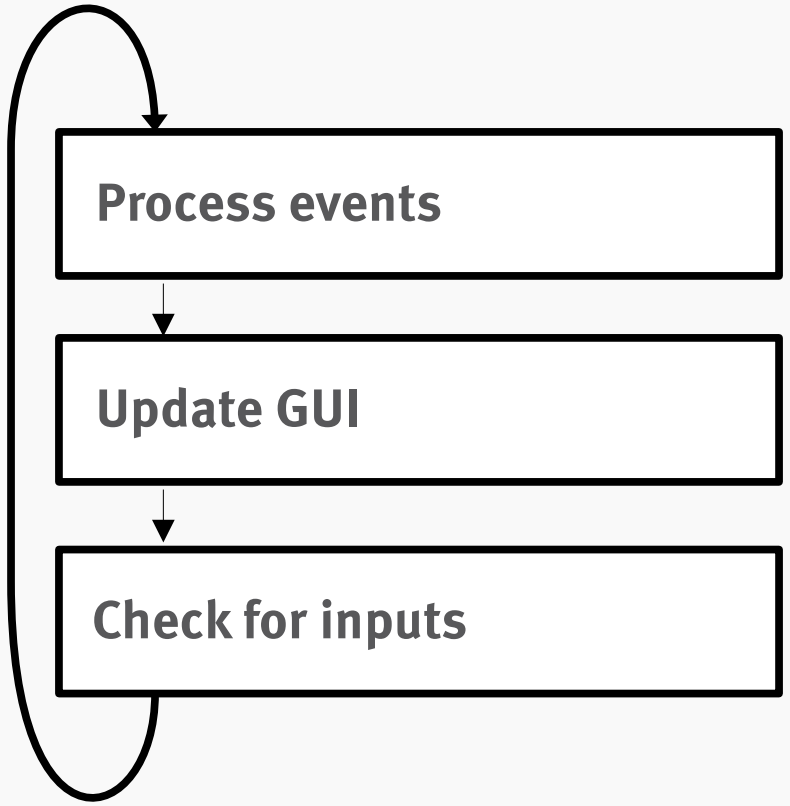


Graphical User Interface

1) Concept (sketch)
2) Rapid prototype
3) Implement

Consider these steps for other projects as well!

Help, my GUI freezes! | How GUIs handle events



Everything is running one line after the other!

If no second thread is used, every wait time freezes the GUI

Second thread can communicate with main thread event loop

In Qt:
Events = Signals
Event processors = slots

https://realpython.com/python-pyqt-qthread/

WWU
MÜNSTER

Demo

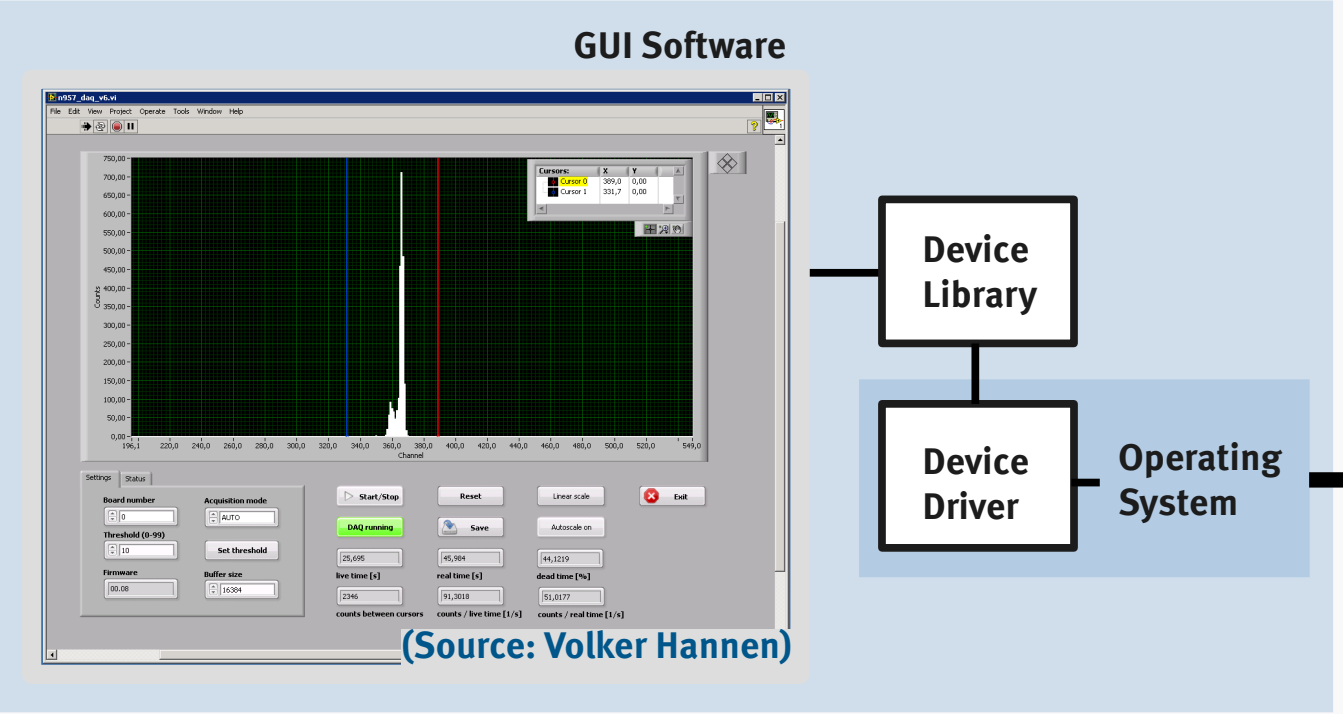
living.knowledge

A GUI is nice – but how does it interact with my device?

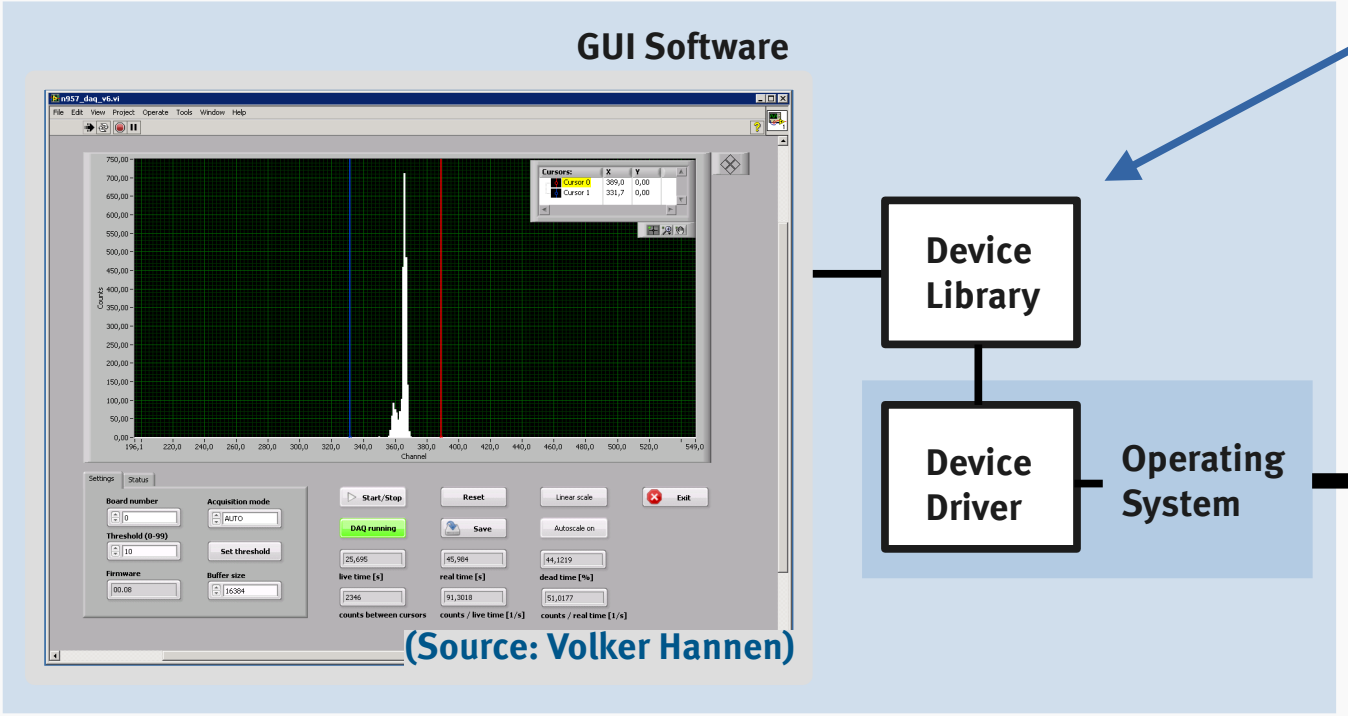
Outline

- Introduction
- How to write Graphical User Interfaces
- **How to connect to devices – how manufacturers intend it**
- How to connect to devices – writing own drivers

USB communication under the hood



USB communication under the hood



USB communication using existing library

1) Find documentation for the library

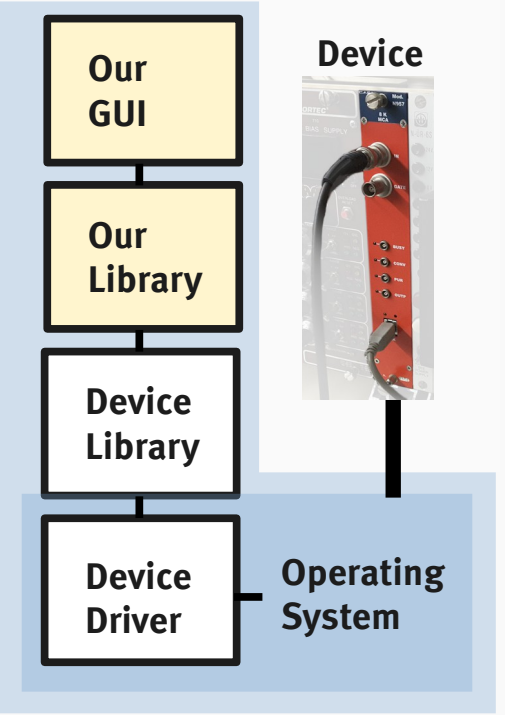
Usually provided by manufacturer

2) Interface with that library in whatever language it supports (e.g. C)

Manufacturer sometimes provides example code

3) If required: Write a wrapper for the language you want to use (e.g. Python)

For sending large amounts of data between Python & C/C++, use NumPy: https://pythonextensionpatterns.readthedocs.io/en/latest/cpp_and_numpy.html



But if the manufacturer's library doesn't work for me?

- **Requires old software (remember ActiveX?)** [Yudong Sun, deRSE23]
- **Requires old/wrong operating system**
- **Difficult to install**

Potential solution: Use generic library (such libraries exist for scanners/printers/keyboards/mice/cameras/storage/...)

But if the manufacturer's library doesn't work for me?

- **Requires old software (remember ActiveX?)** [Yudong Sun, deRSE23]
- **Requires old/wrong operating system**
- **Difficult to install**

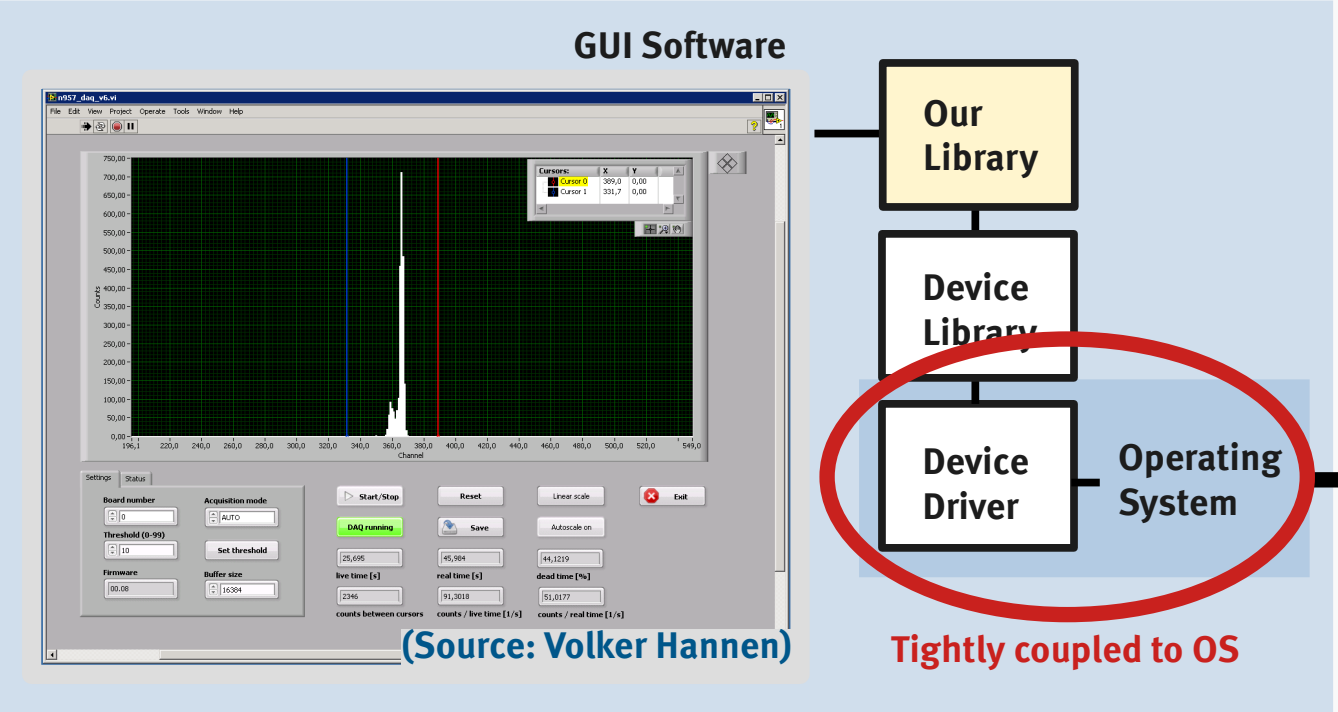
Potential solution: Use generic library (such libraries exist for scanners/printers/keyboards/mice/cameras/storage/...)

...else: Write your own drivers & library!

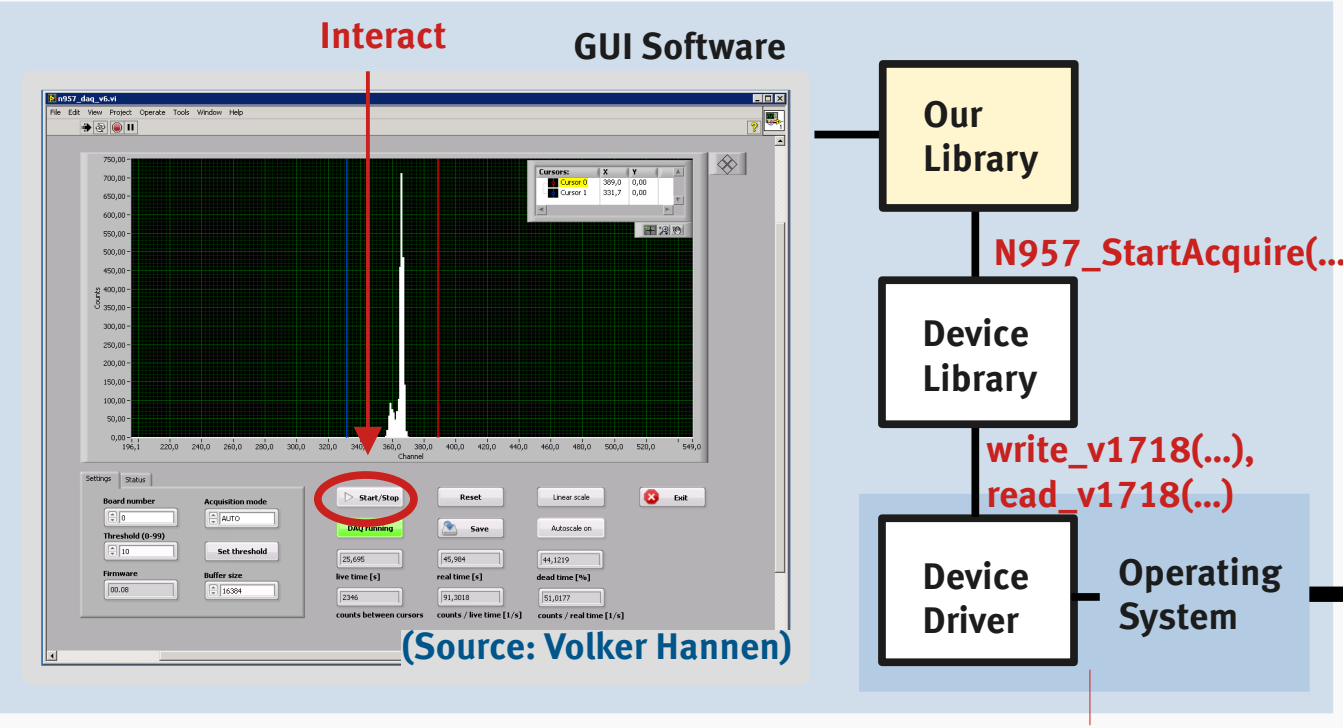
Outline

- Introduction
- How to write Graphical User Interfaces
- How to connect to devices – how manufacturers intend it
- **How to connect to devices – writing own drivers**

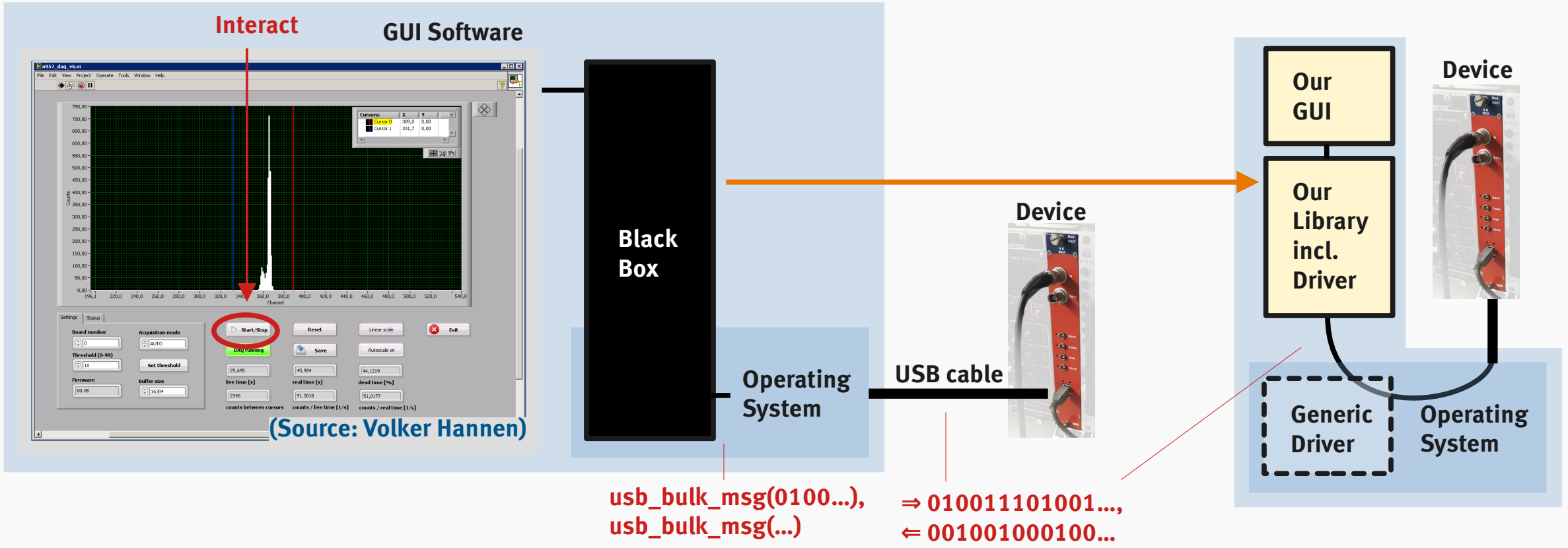
USB communication under the hood, part II



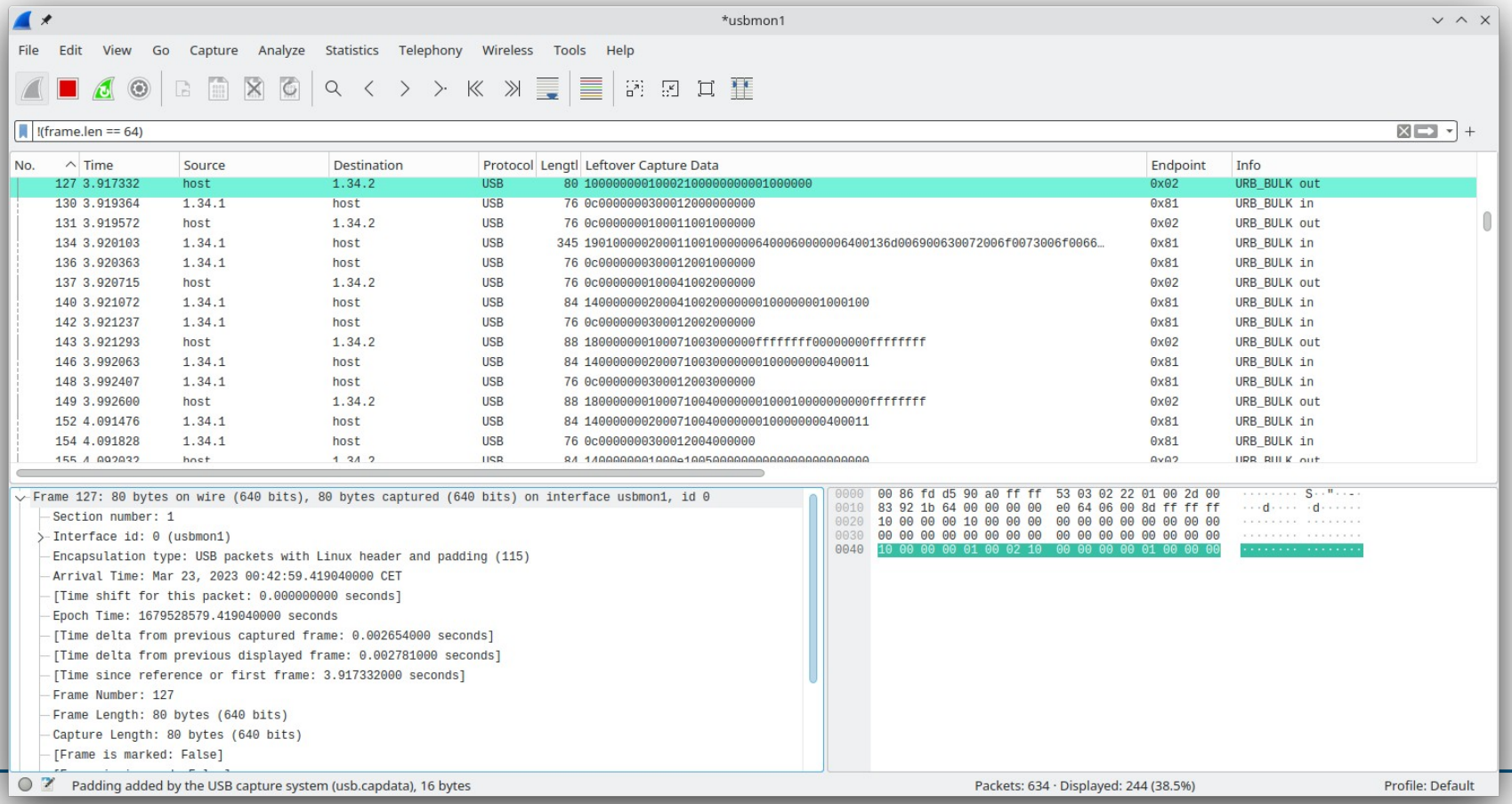
USB communication under the hood, part II



Understanding existing drivers: Reverse engineering



Understanding existing drivers: Wireshark



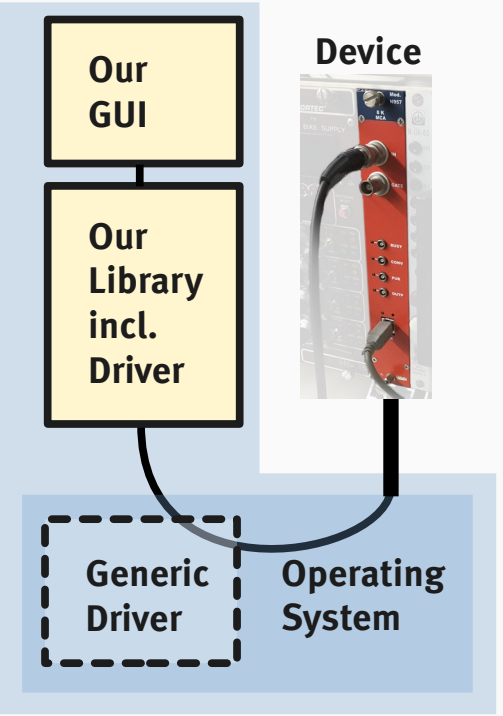
Writing a new driver: PyUSB

PyUSB allows to write user-space USB drivers

Procedure:

1) Look at binary data going over Wireshark
2) Try sending and receiving the same data
3) Look at Wireshark to understand if one is really doing the same

Beware: PyUSB doesn't integrate your driver into the operating system!
Scanner/Printer/Keyboard/Mouse/Camera/Storage… won't be usable with existing applications when writing PyUSB-based drivers
=› For Operating System integration, develop kernel-space driver (in C)

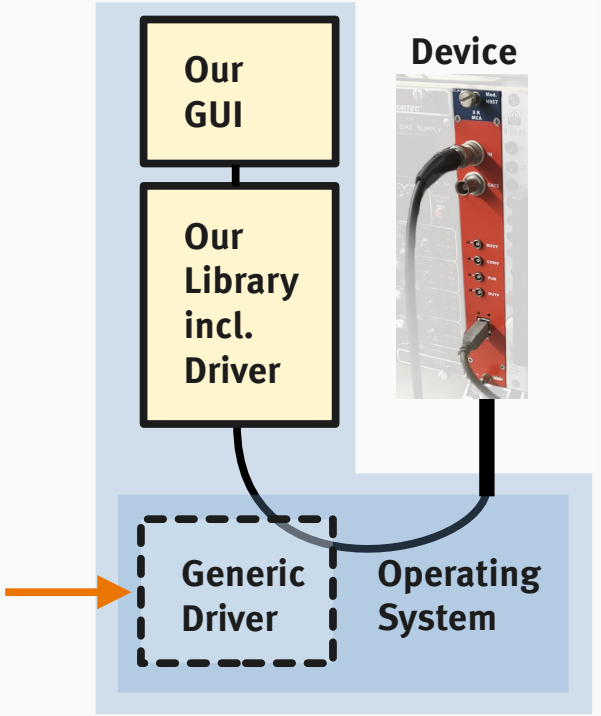


Generic drivers

Our driver should be operating system independent

=› Have the least interaction with OS possible

=› Passthrough USB driver just forwards our 0s and 1s to device



Generic drivers

How to activate/install them:

Linux:

Add a file /etc/udev/rules.d/somename.rules including the following content (vendor and product ids are listed in the table below).

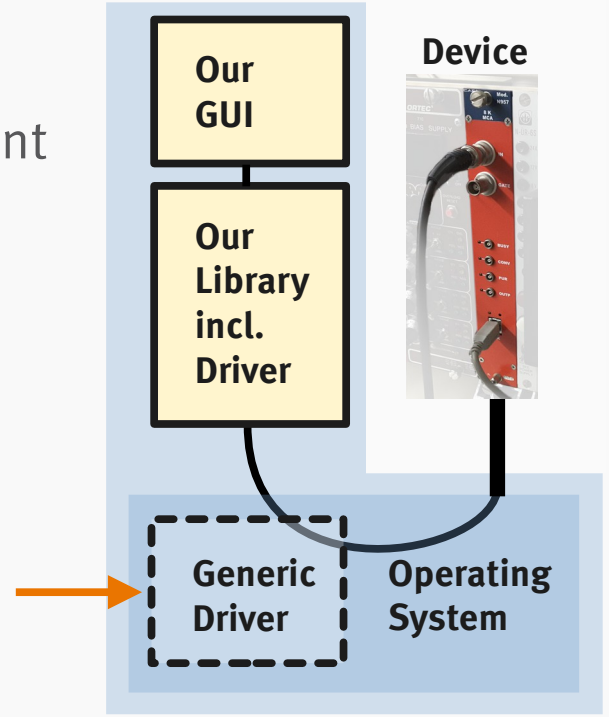
SUBSYSTEM=="usb", ATTRS{idVendor}=="...", ATTRS{idProduct}=="...", MODE="0666"

Then, run the following two commands:

sudo udevadm control –reload
sudo udevadm trigger

Windows:

Use Zadig (https://zadig.akeo.ie/) to install "libusb-win32"



WWU
MÜNSTER

Demo

Notes:
- Byte-ordering (Endianness)
- Coloring rules in Wireshark

WWU MÜNSTER

Conclusion

General

- USB devices can be used more freely than manufacturers anicipate

Graphical User Interfaces

- Writing GUIs for scientific contexts quite easy nowadays
- Discuss with other users, keep everything simple

Device interaction

- Using existing device libraries with Python possible
- Writing custom drivers requires some puzzle solving, but also works

- If you're interested in preventing memory loss, faultguard may be useful (https://github.com/2xB/faultguard)

https://2xb.github.io/usb4research

living.knowledge