



O. V. Lounasmaa  
Laboratory



Aalto University

# Software system for $^3\text{He}$ NMR experiments

Vladislav Zavjalov

Group meeting  
5.6.2017

# Hardware

RF lock-in (SR844)

power supply frame (Keysight N6700B)

- two N6762A precision modules

- two N6731B modules

generator (Agilent 33511B)

2-channel generator (Keysight 33510B)

grounding plate

oscilloscope (PicoScope 4224)

panel for current terminals

power supply (Tenma )

multimeters (Keysight 34461A)

panel with SMA connectors

microcontroller for relay control

computer

gpib to ethernet converter

50-port network switch



# Device library

<https://github.com/slazav/tcl-device>

---

## TCL language:

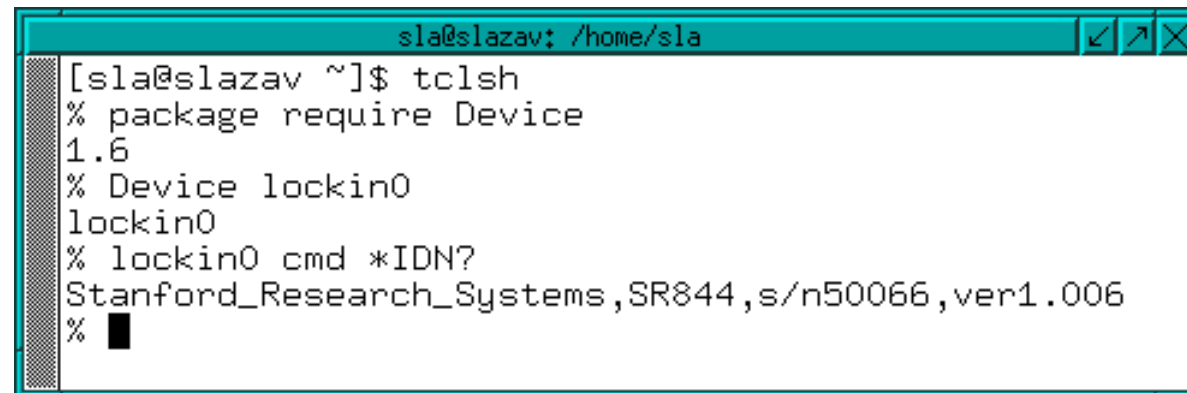
- easy to make graphical interfaces
- used in ROTA (some programs can be used)
- good for interaction between programs

Main idea: programs do not care about how devices are connected.

Program can just open a device, send a command and get an answer.

## Other features:

- error handling
- IO locks
- user locks
- timeouts
- logging



```
sla@slazav: /home/sla
[sla@slazav ~]$ tclsh
% package require Device
1.6
% Device lockin0
lockin0
% lockin0 cmd *IDN?
Stanford_Research_Systems,SR844,s/n50066,ver1.006
% █
```

# Device library – configuration

```
mc [root@slazav_exp.localdomain]:/etc
devices.txt [-M--] 60 L:[ 1+14 15/ 35] *(739 /1718b) 92 0x[*][X]
# device driver          parameters
#=====
lockin0  gpib_prologix  gpib0:8      # SR844 lock-in
gen0     lxi_scp_i_raw  gen0         # 2-ch generator
gen1     lxi_scp_i_raw  gen1         # 1-ch generator
mult0    lxi_scp_i_raw  mult0        # Keysight 34461A multimeter
mult1    lxi_scp_i_raw  mult1        # Keysight 34461A multimeter
osc0     spp      pico_rec -d ER245/039 # picoscope 4224
osc1     usbtcm   /dev/tek_osc0 # Tektronix TDS2014B oscilloscope

ps0      lxi_scp_i_raw  ps0          # Keysight PS frame
ps1      tenma_ps  /dev/tenma_ps0 # tenma PS

lockin   gpib      -board 0 -address 6 -trimright "\r\n"
mult_ag  gpib      -board 0 -address 17 -trimright "\r\n"
mult_hp  gpib      -board 0 -address 22 -trimright "\r\n"
capbr    gpib      -board 0 -address 28 -timeout 1000

db       spp      graphene -i
db_local spp      graphene -i -d .

1Help 2Save 3Mark 4Replac 5Copy 6Move 7Se~ch 8Delete 9PullDn10Quit
```

## Device library – using programs as devices, remote access

```
sla@slazav: /home/sla
$ device -d lockin0
#SPP001
#OK
*idn?
Stanford_Research_Systems,SR844,s/n50066,ver1.006
#OK
*i?
#Error: Read timeout: ssh slazav_exp device -d lockin0
```

```
mc [sla@slazav.localdomain]:/etc
devices.txt [-M--] 0 L:[ 10+ 3 13/ 32] *([*][X])
ps1 spp ssh slazav_exp device -d ps1
osc0 spp ssh slazav_exp pico_rec -d ER245/039
osc1 spp ssh slazav_exp device -d osc1
db spp graphene -i
db_exp spp ssh slazav_exp graphene -i
db_local spp graphene -i -d .
sweep1 spp ssh slazav_exp sweeper -ps_dev1 ps0:1H
sweep2 spp ssh slazav_exp sweeper -ps_dev1 ps0:3
sweep3 spp ssh slazav_exp sweeper -ps_dev1 ps0:4
sweep4 spp ssh slazav_exp sweeper -ps_dev1 ps1
1Help 2Save 3Mark 4Re~ac 5Copy 6Move 7Se~ch 8De~te
```

# Graphene database

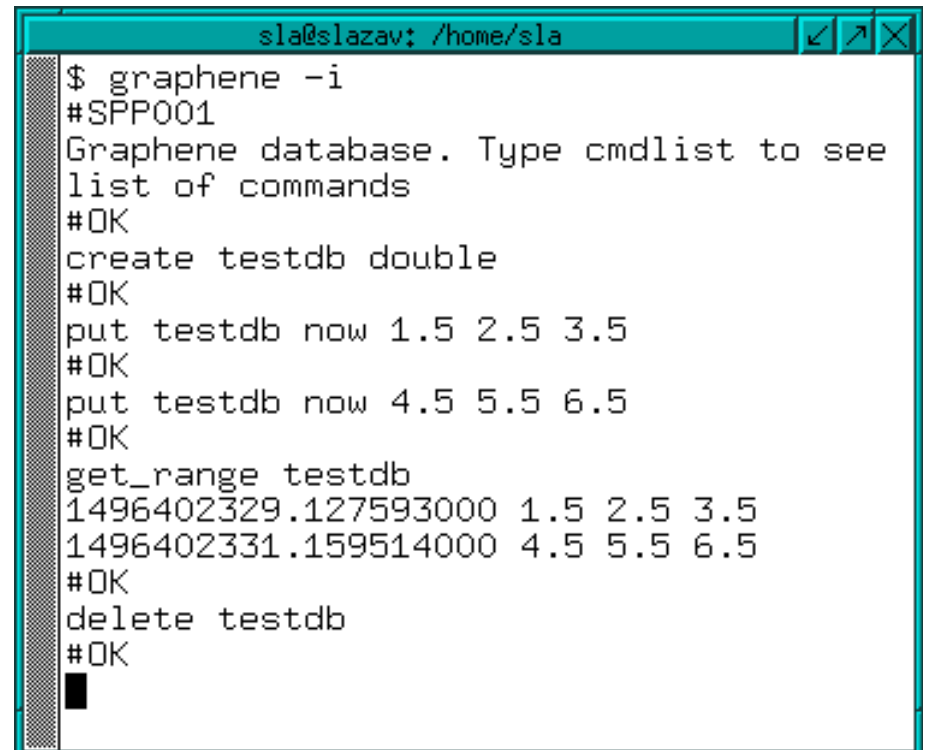
<https://github.com/slazav/graphene>

---

Main idea: you can put a few numbers or text with a timestamp into a database. Then you can extract data for any time range

Features:

- based on BerkleyDB
- integer, floating point or text values
- nanosecond-precision timestamps
- multi-column numerical values
- fast access to data, interpolation, downsampling
- command line interface
- http interface for web-applications (Grafana viewer)



```
sla@slazav: /home/sla
$ graphene -i
#SPP001
Graphene database. Type cmdlist to see
list of commands
#OK
create testdb double
#OK
put testdb now 1.5 2.5 3.5
#OK
put testdb now 4.5 5.5 6.5
#OK
get_range testdb
1496402329.127593000 1.5 2.5 3.5
1496402331.159514000 4.5 5.5 6.5
#OK
delete testdb
#OK
█
```

# DeviceRole library

[https://github.com/slazav/tcl-device\\_role](https://github.com/slazav/tcl-device_role)

---

Main idea: program can use a device in some simple role, without a knowledge about its model and command set.

Program can just open a device "as a voltage source", and run "set voltage" method.

Existing roles and supported devices:

power\_supply – a power supply with constant current and constant voltage modes

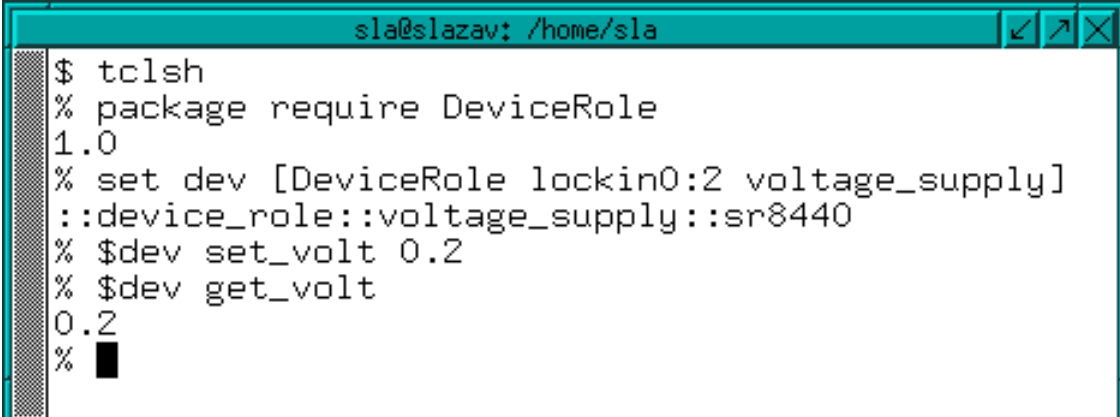
- \* Keysight N6700B frame with N6762A or N6762A modules
- \* Korad/Velleman/Tenma 72-2550 power supply

voltage\_supply – a simple DC voltage source

- \* Korad/Velleman/Tenma 72-2550 power supply
- \* SR844 lock-in (auxiliary outputs)
- \* Keysight 33511B generator (1 channel)
- \* Keysight 33510B generator (2 channels)

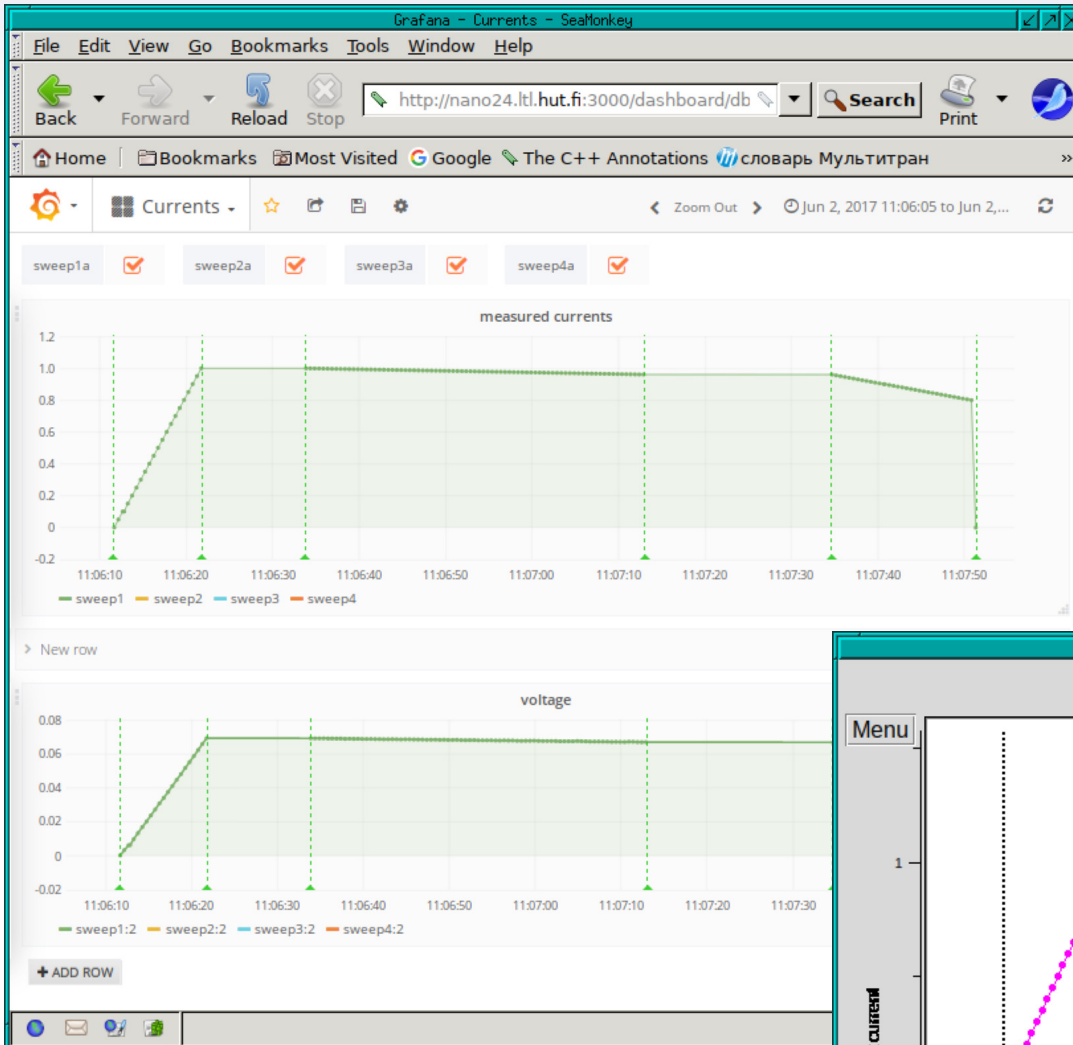
gauge – a gauge device

- \* SR844 lock-in



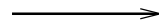
```
sla@slazav: /home/sla
$ tclsh
% package require DeviceRole
1.0
% set dev [DeviceRole lockin0:2 voltage_supply]
::device_role::voltage_supply::sr8440
% $dev set_volt 0.2
% $dev get_volt
0.2
% █
```

# sweeper device



Grafana web-interface

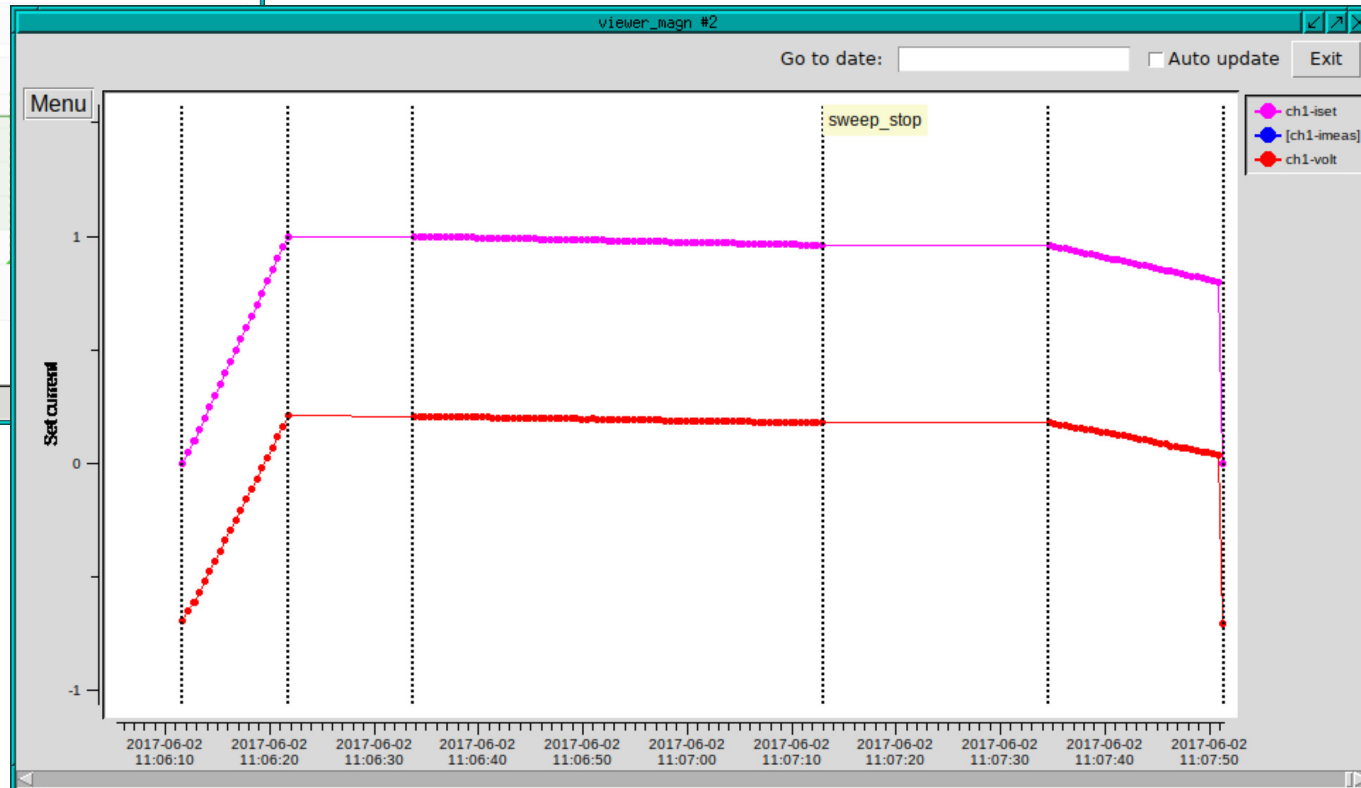
GrapheneViewer



parameters:

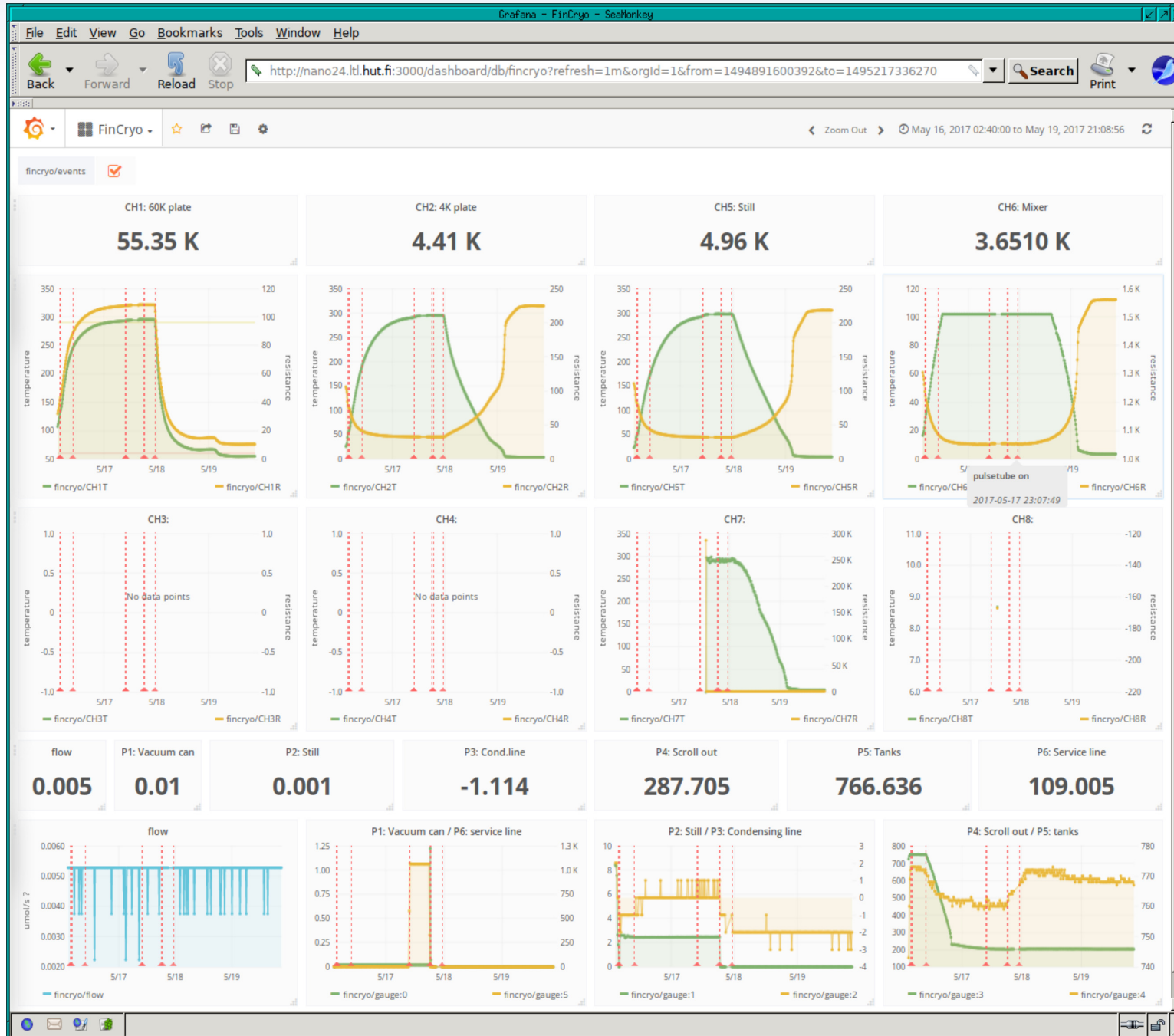
- power supply device
- 2nd power supply
- gauge device
- database device, database name

```
sla@slazav: /home/sla
sweep 1 0.1
#OK
sweep 0 0.001
#OK
sweep_stop
#OK
get_mcurr
0.96052838
#OK
sweep 0 0.01
#OK
get_mcurr
0.91052764
#OK
get_mcurr
0.8951434614
#OK
```

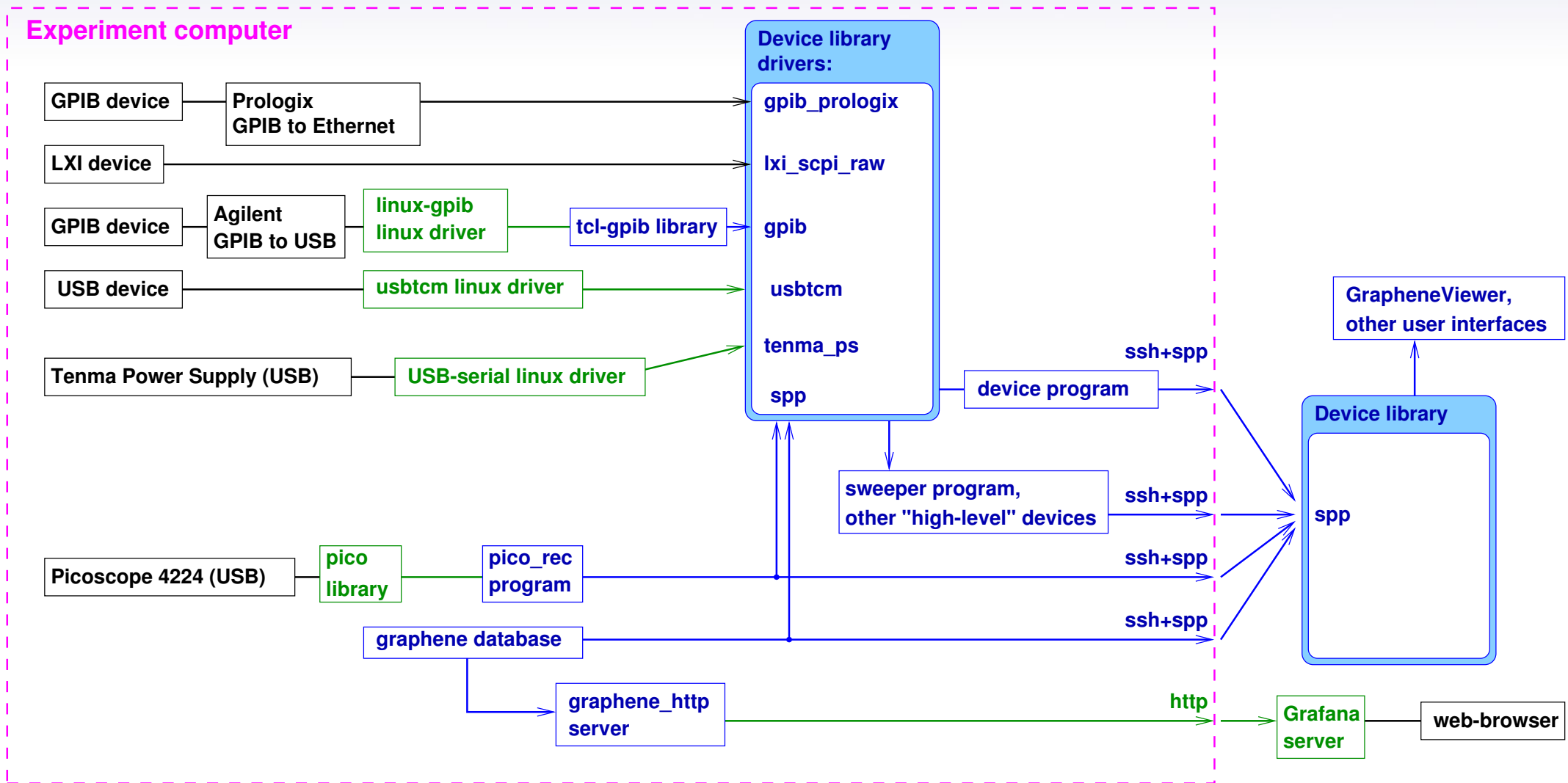




# BlueFors data



# All components



# All components

---

Device library:

<https://github.com/slazav/tcl-device>

DeviceRole library:

[https://github.com/slazav/tcl-device\\_role](https://github.com/slazav/tcl-device_role)

Graphene database:

<https://github.com/slazav/graphene>

pic\_osc – program for controlling oscilloscope and processing signals:

[https://github.com/slazav/pico\\_osc](https://github.com/slazav/pico_osc)

bf2gr – script for synchronizing graphene database with BlueFors logs:

<https://github.com/slazav/tcl-bf2gr>

GrapheneViewer – tcl viewer for graphene database:

<https://github.com/slazav/tcl-grview>

GrapheneMonitor – tcl frame for measurement modules:

<https://github.com/slazav/tcl-grmon>