Lab::VISA

VISA, GPIB, etc.

- Instruments can be connected in various ways:
  Serial port, GPIB, VXI, TCP/IP, etc.

- **GPIB** (hardware and software)
  - GPIB (IEEE488): Standard by Hewlett-Packard
  - Physical layer IEEE488.1
  - Command layer IEEE488.2
  - SCPI (Standard Commands for Programmable Instruments)

- **VISA** (software)
  - Virtual Instrument Software Architecture
  - VXI, GPIB, serial, or computer-based instruments
  - NI-VISA library is one implementation of the VISA standard

Lab::VISA

- Lab::VISA is software to control instruments in the lab via VISA (e.g. voltage sources, multimeters, etc.)
- Alternative to LabView, Gpplus, etc.
- But very different: no GUI, just API

Lab::VISA design goals

- **Flexible**
  - Allow any kind of measurement procedure
  - Control anything that has GPIB or serial connection
- **Safe**
  - Make sure you never drive voltage sources to fast and destroy your gates
- **Helpful**
  - Automatically collect additional information about the data (“metadata”)
  - User should have to type in additional information only once and then never again

Lab::VISA is in Perl

- Lab::VISA is written in Perl and C
- To be used from programs written in Perl

- Perl: interpreted scripting language
- Runs on almost every OS
- Extremely good in reading data files, manipulating data, etc.
- Allows to write quick and dirty scripts that get the job done
- Also allows to write clean fullsize programs

=> ideal for experimental physics

Lab::VISA architecture

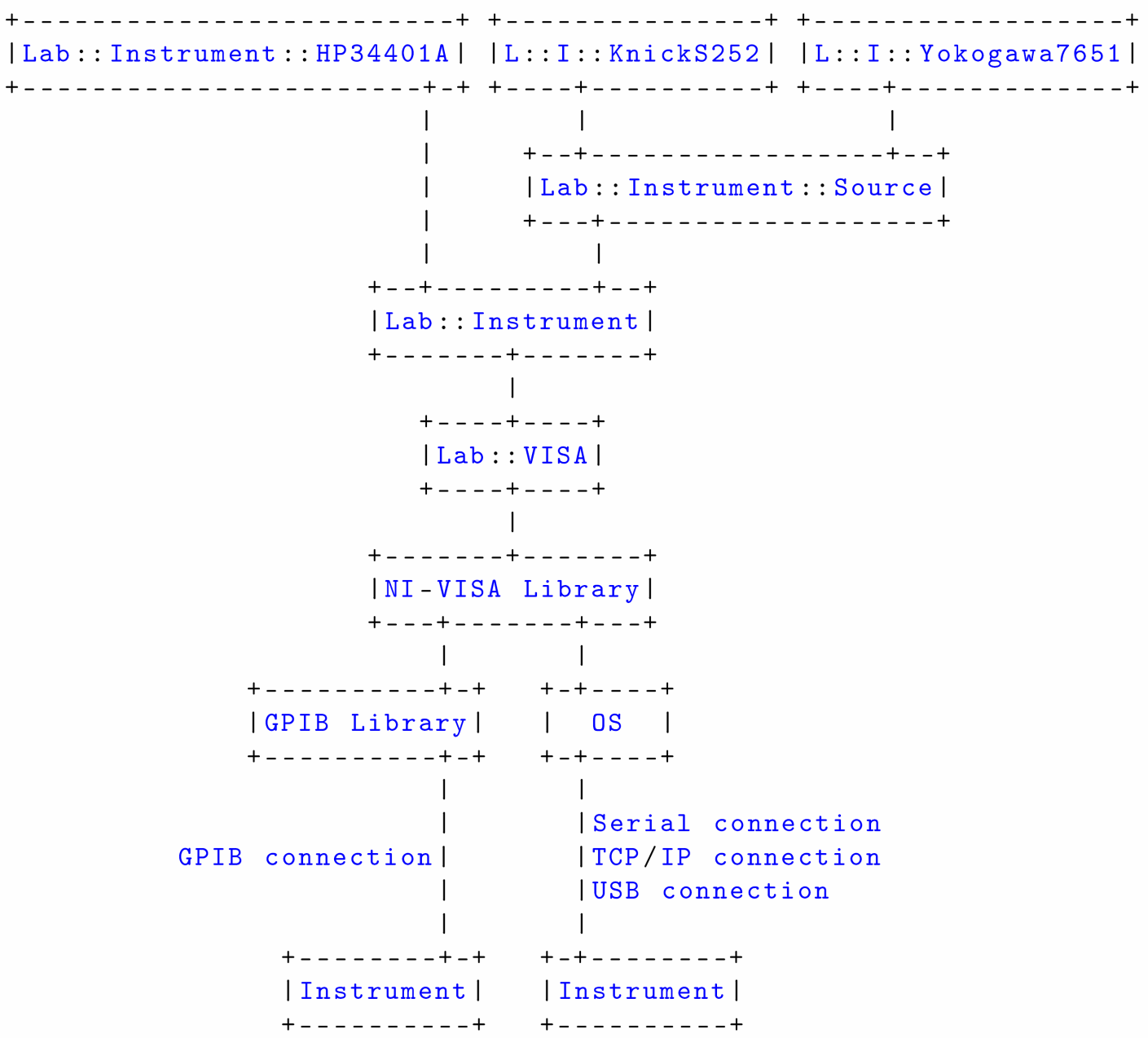
- Divided into three parts
- Build on top of each other
- Provide increasing comfort
- Measurement scripts can be based on any of these stages

- **Lab::VISA** is the lowest layer. It makes the NI-VISA library accessible from perl and therefor allows to make any standard VISA call

- **Lab::Instrument** package makes communication with instruments easier by silently handling the protocol involved

- **Lab::Tools** is the highest abstraction layer. Provides support for writing good measurement scripts. Offers means of saving data and related meta information to disk, plotting data, etc.

Lab::VISA architecture



Example 1

- lab-visa/raw_visa.pl
  - looks like example programs from manual
  - protocol overhead is pain in the neck

Example 2

- lab-instrument/query_id.pl
  - much nicer!

Example 3

- lab-instrument/make_sweep.pl
  - already at the level of GPplus

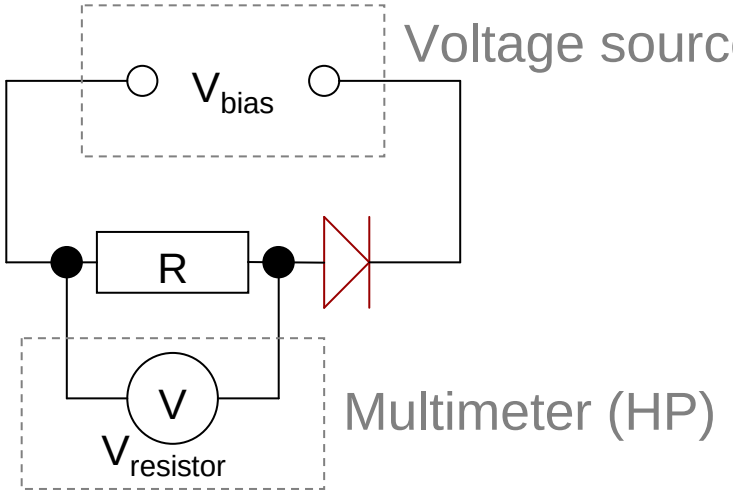
Lab::Tools

- Provides additional tools to write better measurement scripts

- Store metadata alongside data
  - date and time
  - settings of additional instruments
  - ratio of voltage divider
  - color of the shirt you are wearing
  - everything that might be important for a later interpretation of the data

- Don‘t repeat yourself
  - Use above collected information automatically
  - Automatically plot data with correct axes, scaling, labels etc.

Metadata philosophy



Constants

- R = 1000 Ohm

Columns

- C_1: V_{bias}
- C_2: $V_{resistor}$
- unit
- description…

1.5	+8.19300500E-01
1.4	+7.23413000E-01
1.3	+6.28083900E-01
…	…

Axes

- Axis “bias voltage” (C_1)
- Axis “diode current” (C_2 / R)
- Axis “diode resistance” (R * (C_1 / C_2 – 1))
- unit
- expression
- description…

Plots

- Plot “diode current” axis “diode current” over axis “bias voltage”
- Plot “diode resistance” axis “diode resistance” over axis “bias voltage”
- logscale
- grid
- ranges…

Example 4

- lab-measurement/lab-measurement.pl
  - metadata
  - plotter.pl
    - live plot
    - list avaible plots
    - create postscript

Example 5

- real-world/ladediagramm_qpc.pl
  - there is room for further improvements
  - standard measurements should be factored out to dedicated classes like Lab::Measurement::Ladediagramm
  - collection of metadata could be automated further

Other Lab::VISA features

- "**Gate protect**" safety mechanism
  - Makes sure that no voltage is changed to fast
  - Big voltage steps are automatically split into small, slow steps

- **Date/time** handling
  - Date/time column can contain timestamp for every data point
  - Plot data as function of time

- Measurements with **higher dimensionality**
  - Each trace/sweep/line is a "block"
  - Two-dimensional plots, selections of traces, etc.

Online resources

It‘s far from being complete...

Source Code

- http://www.nano.physik.uni-muenchen.de/svn/lab/trunk/

Tutorial

- http://www.nano.physik.uni-muenchen.de/svn/lab/trunk/documentation.pdf

Example 6

- goodbye/goodbye.pl