Lab::Measurement – measurement control with Perl

S. Reinhardt, M. Schambeck, E. Fabrizzi, and A. K. Hüttel

Institute for Experimental and Applied Physics, University of Regensburg, 93040 Regensburg, Germany

Flexible measurement needed?!

- Tired of following your wires in square meters of LabVIEW diagrams?
- Tired of clumsy string handling and low-level driver functions in your looong C program?
- Use a text processing language to manage your measurement! Use Perl!

```
#!/usr/bin/env perl
# Read out SR830 Lock In Amplifier at GPIB address 13
use 5.010;
use Lab::Moose;

my $lia = instrument(
    type               => 'SR830',
    connection_type    => 'LinuxGPIB',
    connection_options => { pad => 13 }
);

my $amp = $lia->get_amplitude();
say "Reference output amplitude: $amp V";

my $freq = $lia->get_freq();
say "Reference frequency: $freq Hz";

my $r_phi = $lia->get_rphi();
my ( $r, $phi ) = ( $r_phi->{r}, $r_phi->{phi} );
say "Signal: r=$r V, phi=$phi degree";
```

Currently supported hardware

Hardware driver backends:

- NI-VISA and all hardware supported by it
- LinuxGPIB and all hardware supported by it
- HTTP, TCP connection, generic network socket
- USB-TMC lightweight driver (Linux, libusb)
- VXI-11 lightweight driver (Linux, libtirpc)
- Oxford Instruments IsoBus
- Zurich Instruments LabOne API

Growing number of high-level drivers (more are *very* easy to add):

- Multimeters: HP / Agilent / Keysight / Keithley
- DC sources: Yokogawa / Keithley / Keysight
- AW generators: Rigol / Agilent / Keysight / Zurich Instruments
- Lock-in amplifiers: Stanford Research / Signal Recovery / Zurich Instruments / SyncTek
- RF / microwave sources, spectrum analyzers, VNAs: Rohde & Schwarz / HP / Agilent / Keysight / Rigol
- Oscilloscopes: Tektronix / Keysight / Rohde & Schwarz
- Temperature controllers: Lakeshore / Oxford Instruments / Bluefors
- Magnet power supplies: Oxford Instruments, American Magnetics (Bluefors)
- Quantum Measurement Systems: Nanonis Tramea

Reference / Cite as

"Lab::Measurement — a portable and extensible framework for controlling lab equipment and conducting measurements", S. Reinhardt *et al.*, Comp. Phys. Comm. **234**, 216 (2019)

Real world VNA measurement

- Measuring $|S_{21}|^2$ for increasing signal power
- Outer loop: power applied by VNA
- Inner loop: hardware-executed VNA frequency sweep, transmission measurement

```
use 5.010;

use Lab::Moose;
use Time::HiRes 'time';
use PDL;

# Record several VNA traces for increasing power

################################################

my $sample = 'NK17'; #chip name

# VNA sweep range
my $freqstart  = 4500000000;
my $freqend    = 6500000000;
my $freqpoints = 10001;

my $vna_bw = 1000;
my $vna_AVG = 1;    # number of sweeps to average

# VNA power
my $powerstart = -20;
my $powerend   = +10;
my $powerstep  = +10;

################################################

my $vna = instrument(
    type               => 'RS_ZVA',
    connection_type    => 'VISA::GPIB',
    connection_options => { pad => 20 },
);

# The power "sweep"
my $sweep_power = sweep(
    type => 'Step::Power',
    instrument => $vna,
    delay_before_loop => 5,
    from => $powerstart, to => $powerend, step => $powerstep
);

# The data file
my $datafile = sweep_datafile(
    type => 'Gnuplot',
    columns => [qw/time power f Re Im Amp phi/],
);

# The measurement procedure
my $meas = sub {
    my $sweep = shift;

    my $pw = $vna->get_power();

    say "Sweeping at power ".$pw."dBm ...";
    my $pdl = $vna->sparam_sweep( timeout => 10000,
                                  average => $vna_AVG );
    say "... done.\n";

    $sweep->log_block(
        prefix => { time => time(), power => $pw },
        block => $pdl,
        add_newline => 1,
    );
};

# Set up the VNA parameters
$vna->sense_bandwidth_resolution( value => $vna_bw );
$vna->sense_frequency_start(value => $freqstart );
$vna->sense_frequency_stop(value => $freqend );
$vna->sense_sweep_points(value => $freqpoints );

# Start the measurement
$sweep_power->start(
    measurement => $meas,
    datafile    => $datafile,
    folder => "vnasweep_$sample",
);

# Set power to low value at end
$vna->set_power( value => -20 );
```

Output files

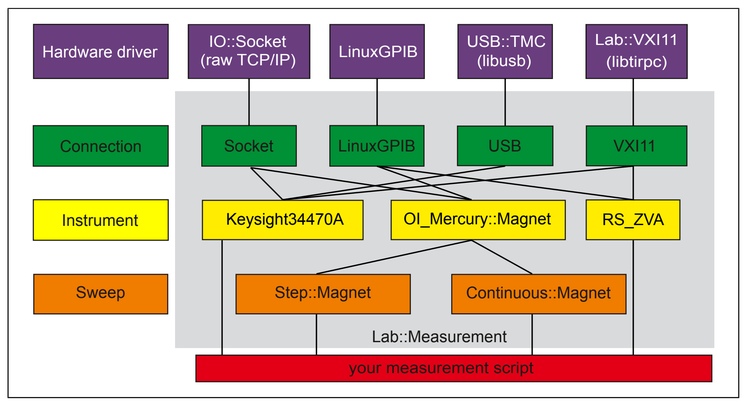
```
huettel@dilfridge ~/VNA$ ls 2023-02-16_13-22-58_vna_NK17_001/
META.yml  data.dat  vnasweep.pl
```

- *META.yml:* various metadata (host, user, date, L::M version, command line arguments)
- *data.dat:* measured data, in tab-separated Gnuplot format
- *data.png:* live plot at the end of the measurement, as a png image (optional)
- archival copy of the measurement script
- RFC3161 cryptographic timestamp attestation of data (optional)

Advanced sweep features

- Multidimensional sweeps, e.g. 3D sweep: creating one 2D datafile for each step of the outermost sweep
- Log arrays and matrices of data (PDLs). Useful for spectrum analyzers, VNAs, oscilloscopes which do fast, internally controlled sweeps.
- Extensive support for live plots via gnuplot: line plots (2D data) and color maps (3D data)
- Customizing live plots: access to all gnuplot plot and curve options via PDL::Graphics::Gnuplot

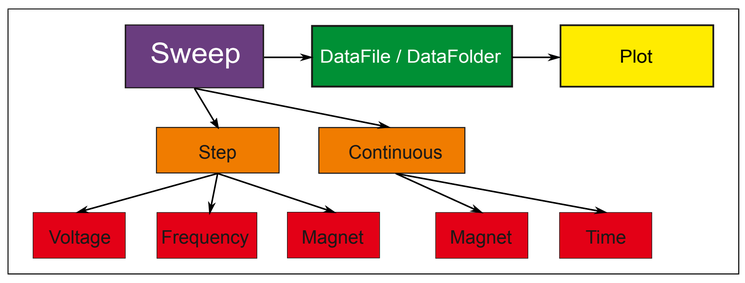
Code abstraction layers



- Modular structure. Easy to extend with new instrument drivers and connection types
- Abstract IO layer, makes instrument drivers independent of hardware backends

High-level sweep framework

- Modern Perl implementation; use state of the art object-oriented programming



- Separate classes for sweeps, datafiles, datafolders, and plots
- Most operational details of sweeps implemented in subclasses of Lab::Moose::Sweep
- High modularity: *very* easy to extend

New improvements / additions

- Bluefors dilution refrigerator temperature control
- American Magnetics AMI 430 (Bluefors) magnet power supply
- Keysight N9310A RF generator
- Zurich Instruments HDAWG arbitrary waveform generator

Very stable

- No continuous adaptions (e.g., for new Python versions or operating system versions) needed
- None of the new Perl releases over the past years have ever required changes to measurement scripts

Key facts

- Programmed in Perl 5, following the Modern Perl guidelines
- Open source / free software
- https://www.labmeasurement.de/
- License: same as Perl (GPL-1+ or Artistic)
- Releases on CPAN, development on Github
- Contributors and collaborations welcome!

We gratefully acknowlegde funding by the DFG via SFB 1277.