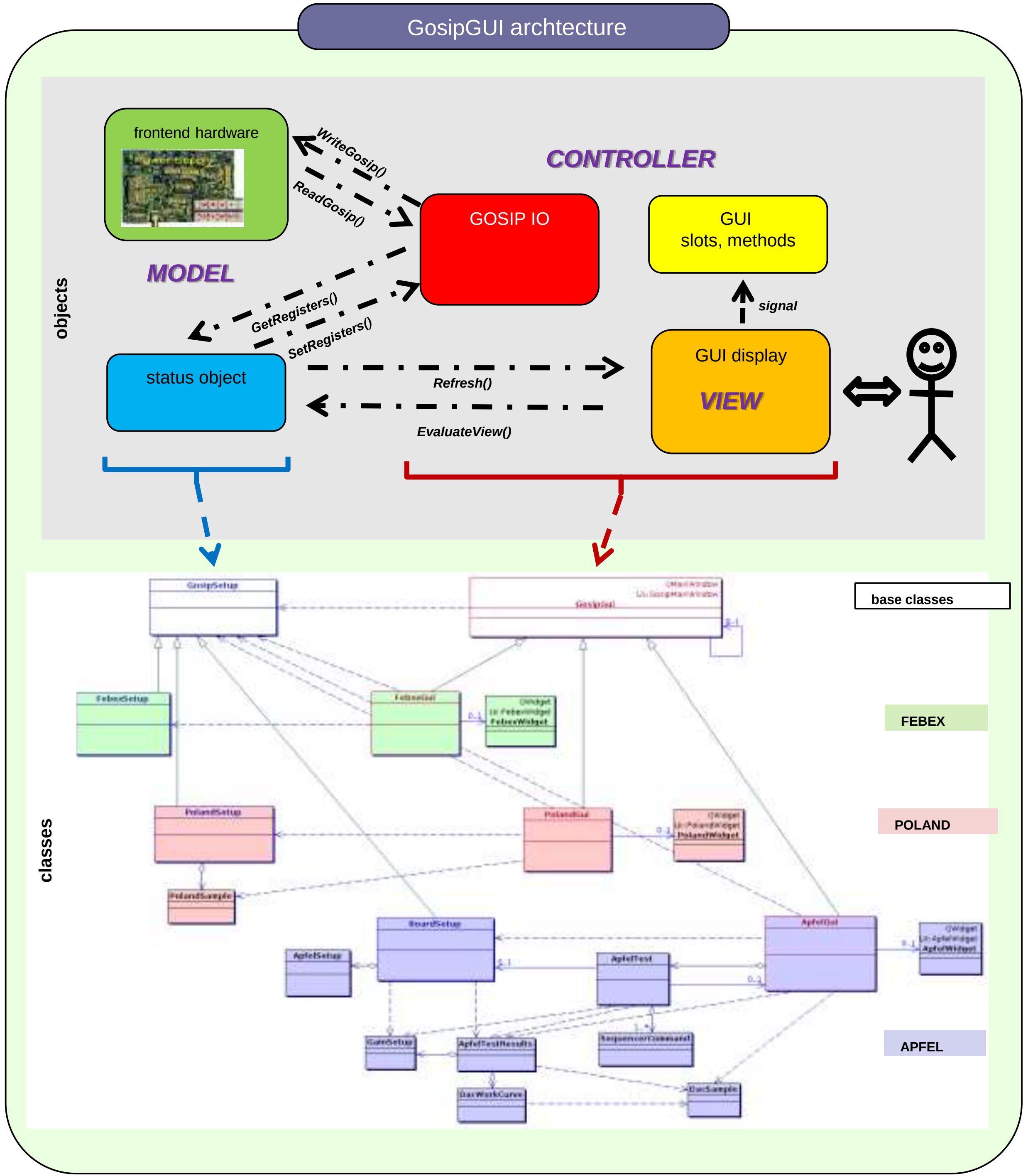
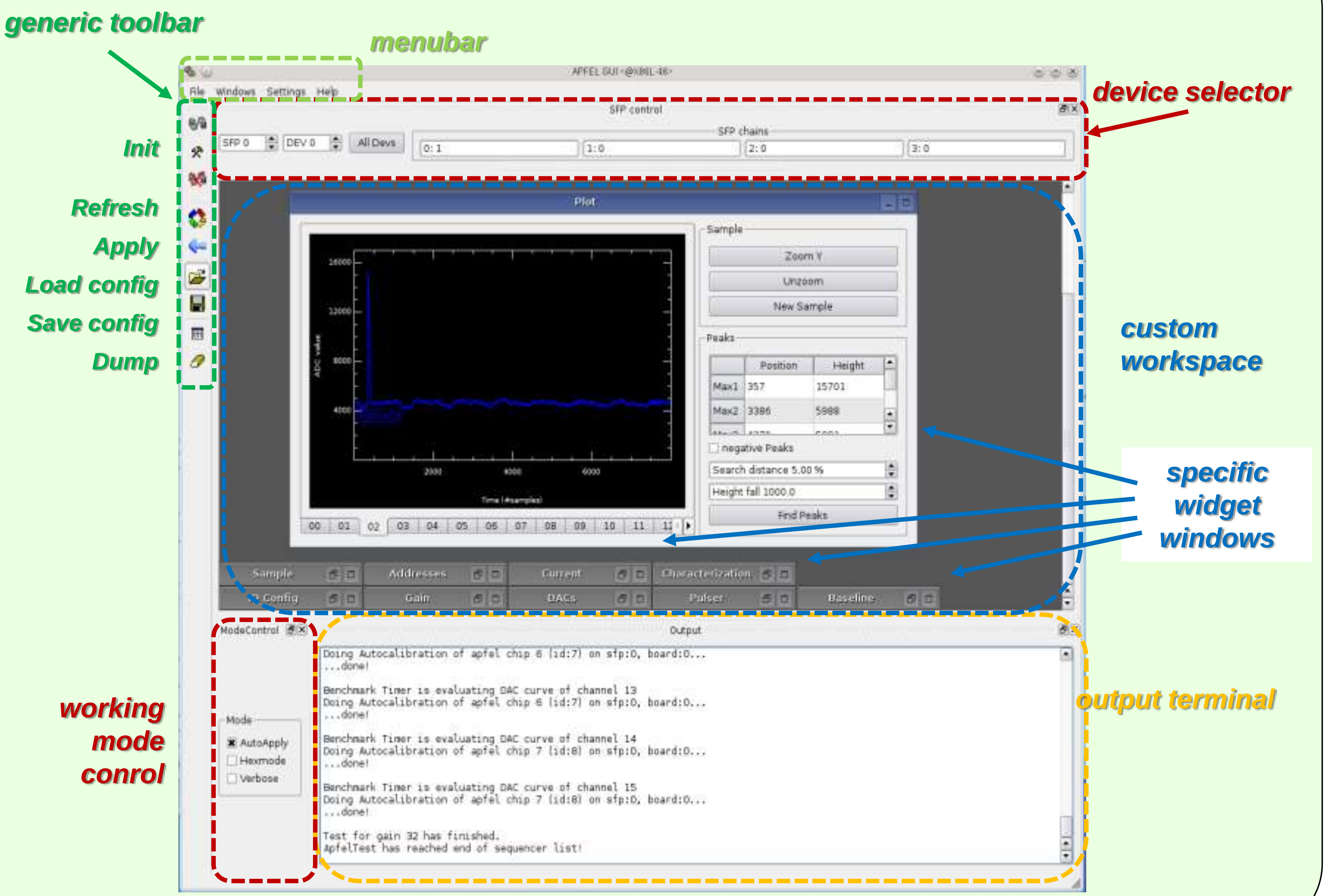
The GosipGUI framework for control and benchmarking of readout electronics front-ends

Jörn Adamczewski-Musch,
Experiment Electronics department, GSI,
Darmstadt, Germany

- FEE boards at **PCIe optical fibre chain**
- use GSI **GOSIP protocol** and drivers
- use **GUIs based on Qt5**
- C++ **software framework** to handle various devices with dedicated GUIs



Common GUI frame



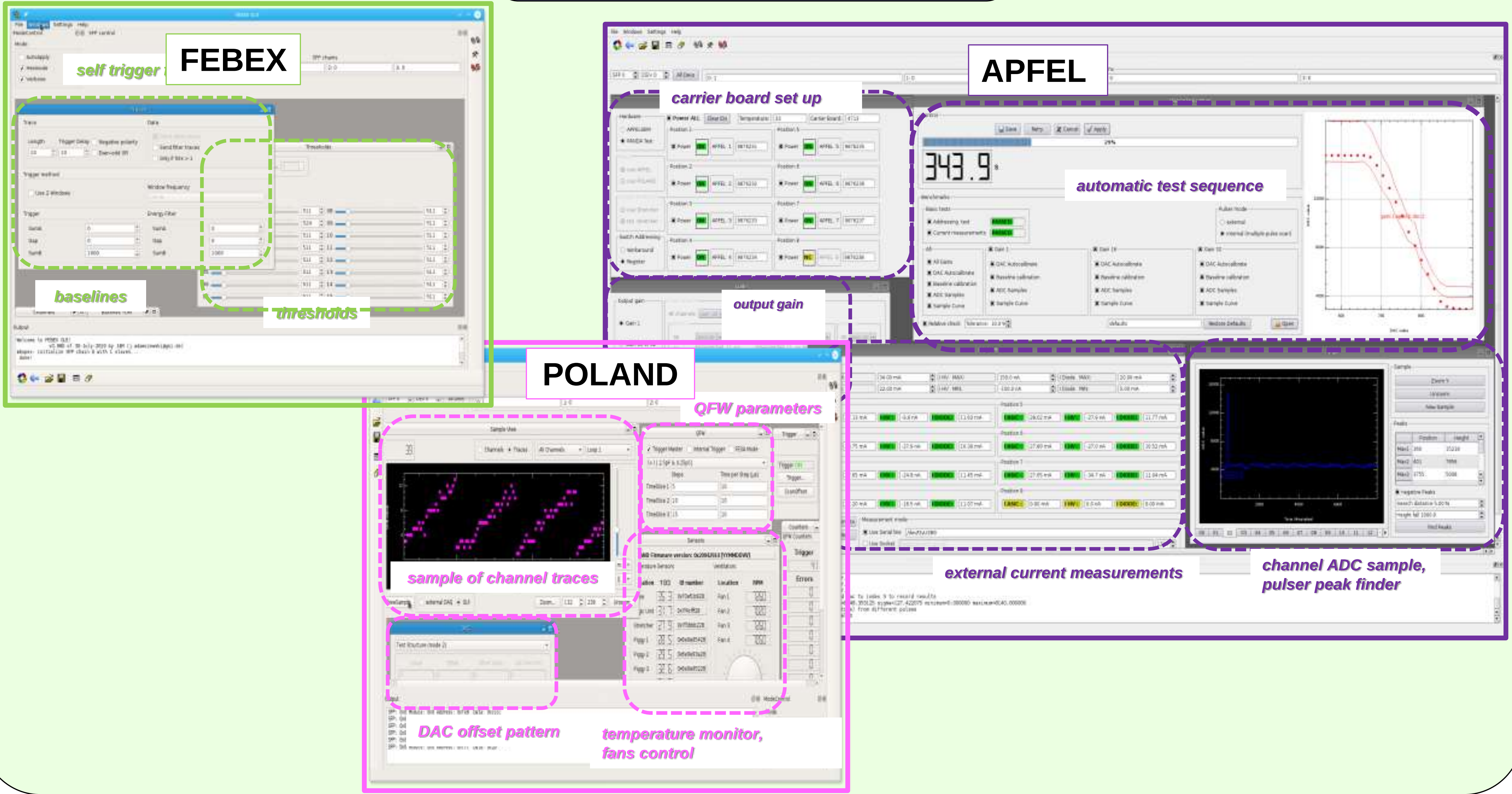
	Position	Height
Max1	357	15701
Max2	3386	5988

Search distance 5.00 %
Height fall 1000.0

Mode: [✓ AutoApply] Hexmode Verbose

```
Doing Autocalibration of apfel chip 6 (id:7) on sfp:0, board:0...
...done!

Benchmark Timer is evaluating DAC curve of channel 13
Doing Autocalibration of apfel chip 6 (id:7) on sfp:0, board:0...
...done!

Benchmark Timer is evaluating DAC curve of channel 14
Doing Autocalibration of apfel chip 7 (id:8) on sfp:0, board:0...
...done!

Benchmark Timer is evaluating DAC curve of channel 15
Doing Autocalibration of apfel chip 7 (id:8) on sfp:0, board:0...
...done!

Test for gain 32 has finished.
ApfelTest has reached end of sequencer list!
```

Application examples



The GosipGUI framework for control and benchmarking of readout electronics front-ends

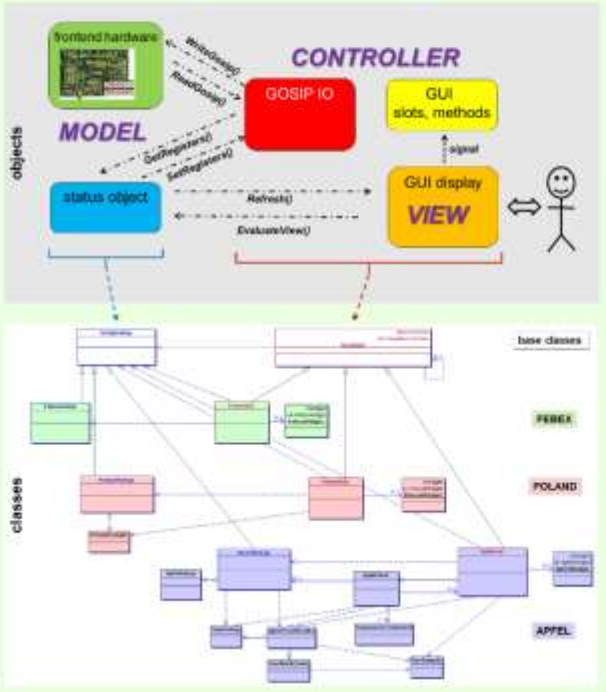
Jörn Adamczewski-Musch, Experiment Electronics department, GSI, Darmstadt, Germany

mailto: j.adamczewski@gsi.de

Abstract

The GOSIP (Gigabit Optical Serial Interface Protocol) [1] provides communication via optical fibers between multiple kinds of frontend electronics and the KINPEX PCIe receiver board located in the readout host PC. In recent years a stack of device driver software has been developed to utilize this hardware for several scenarios of data acquisition [2]. On top of this driver foundation, several graphical user interfaces (GUIs) have been developed in previous years. These GUIs are based on the Qt graphics libraries and are designed in a modular way: All common functionalities, like generic I/O with the front-ends, handling of configuration files, and window settings, are treated by a framework class GosipGUI. In the Qt workspace of such GosipGUI frame, specific subclasses may implement additional windows dedicated to operate different GOSIP front-end modules. These readout modules developed by GSI Experiment Electronics department are for instance FEBEX sampling ADCs, TAMEX FPGA-TDCs, or POLAND QFWs [3]. For each kind of front-end the GUIs allow to monitor specific register contents, to set up the working configuration, and to interactively change parameters like sampling thresholds during data acquisition. The latter is extremely useful when qualifying and tuning the front-ends in the electronics lab or detector cave. Moreover, some of these GosipGUI implementations have been equipped with features for mostly automatic testing of ASICs in a prototype mass production. This has been applied for the APFEL-ASIC component [4] of the PANDA experiment currently under construction, and for the FAIR beam diagnostic readout system POLAND. The GosipGUI framework is available under GPL at [5].

GosipGUI archtecture



Base class features

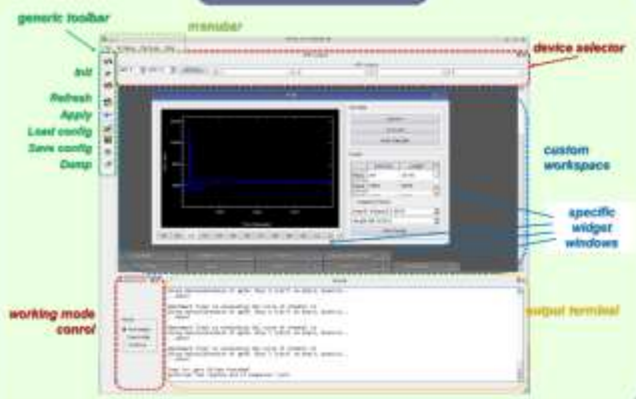
GosipGui:

- based on **Qt5 environment**, inherits *QMainWindow*, provides *QWorkspace*, uses *QSettings*
- implements **all common GUI elements**:
  - selector of currently controlled front-end
  - controls operation mode (auto-apply, number display, terminal verbosity)
  - buttons to set and retrieve all hardware registers, save and apply setup scripts
  - embedded terminal for text dump
  - status message line
- offers **central workspace** to operate specific widgets of subclass GUIs
- fully configurable **dock window toolbars** and menus
- **save and restore** window and toolbar geometry **preferences**
- implements generic **GOSIP communication methods** via device driver software
- keeps **list of GosipSetup objects** for each connected front-end slave
- **factory pattern** to create appropriate GosipSetup object in *GosipGUI* subclass
- **virtual methods interface** to use subclass functionalities with generic GUI elements

GosipSetup:

- interface for any kind of structure **representing the register state of one GOSIP slave**
- virtual method *Dump()* for optional printout to terminal

Common GUI frame



Framework interface

GosipGui virtual methods (to be re-implemented in subclass):

GosipSetup* CreateSetup()
- factory method for the setup object

RefreshView ()
- update gui display from status structure

EvaluateView ()
- put gui values into status structure

GetRegisters ()
- get register contents from hardware to status structure

SetRegisters ()
- set register contents from status structure to hardware

SaveRegisters()
- get registers and write them to config file

ResetSlave()
- reset or initialize the GOSIP slave device

DumpSlave()
- printout some device registers to terminal window

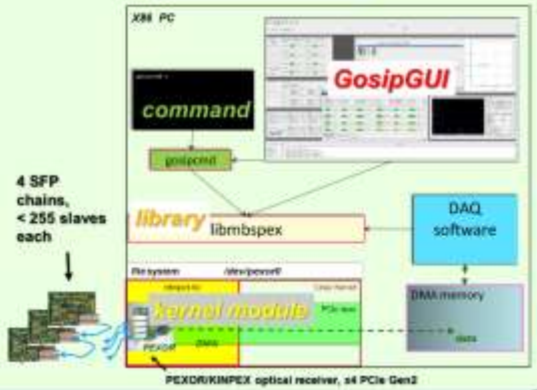
SaveConfig()
- save current hardware configuration to a gosipcmd script file

ApplyFileConfig()
- apply configuration from gosipcmd file to the hardware

Preprocessor macros (used in GosipGui subclass):

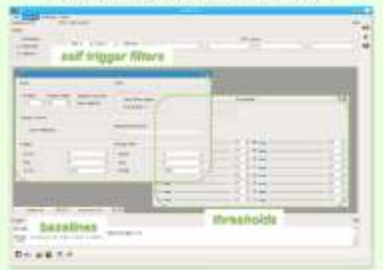
- `theSetup_GET_FOR_SLAVE(X)`
  - cast GosipSetup to actual implementation X, provides handle `theSetup->` to access special member
- `GOSIP_BROADCAST_ACTION(X)`
  - execute function X for the selected front-end; can optionally do it in a "broadcast mode" for an SFP chain, or for all chains
- `GOSIP_AUTOAPPLY(X)`
  - execute function X for selected front-end only if GUI is in "auto-apply" mode. Used for interactive tuning of single registers without writing complete setup to hardware

Device driver software

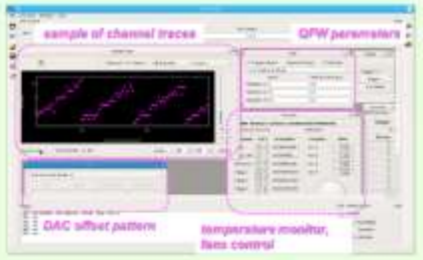


Application examples

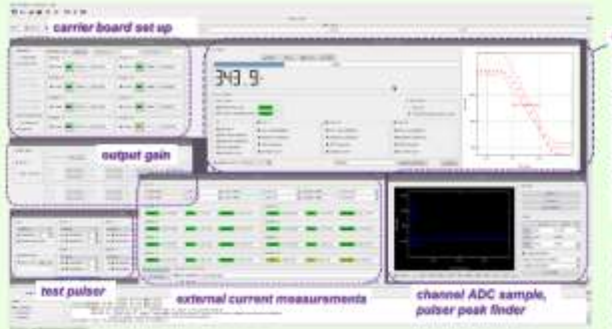
FEBEX 16 channel-pipelining ADCs



POLAND: 32 channel charge frequency converters for FAIR beam diagnostic SEM grid system



Characterization of APFEL preamplifier ASIC for PANDA experiment


carrier board set up
343.9
output gain
test pulser
external current measurements
channel ADC sample, pulser peak finder

automatic characterization sequence

- used for commissioning of APFEL ASIC
- carrier board with 8 sockets for APFEL rigid flex PCBs
- control via FEBEX front-end
- sample DAQ with MBS system
- APFEL chip id from QR code scanned by human operator
- sequencer runs all required tests
- figures of merit are stored to a database
- already 8000 APFEL PCBs for PANDA EMC-Barrel processed since 2017

*APFEL (ASIC for Panda Front-end ELectronics): integrated charge sensitive preamplifier and shaper optimized for the readout of avalanche photo diodes with large detector capacitance and high event rates.

References

[1] S. Minami, J. Hoffmann, N. Kurz, and W. Ott, *"Design and Implementation of a Data Transfer Protocol via Optical Fibre"* (PDAQ-31), presented at the 17th IEEE-NPSS RT2010, Lisbon, Portugal, May 24-28, 2010

[2] J. Adamczewski-Musch, N. Kurz, and S. Linev, *"MBSPEX and PEXORNET - Linux Device Drivers for PCIe Optical Receiver DAQ and Control"*, IEEE Trans. on Nucl. Science, vol. 65 , issue: 2 , Feb. 2018, https://doi.org/10.1109/TNS.2017.2783043

[3] S. Löchner, J. Adamczewski-Musch, H. Bräuning, J. Frühauf, N. Kurz, S. Linev, S. Minami, and M. Witthaus, *"POLAND - Low Current Profile Measurement Readout System"*, GSI Darmstadt, Germany, Sci. Rep. [Online]. 2013. Available: http://dx.doi.org/10.15120/GR-2014-1-FG-CS-13

[4] P. Wieczorek, S. Löchner, and J. Adamczewski-Musch, *"First setup for the routine tests of the APFEL-ASIC rigid flex PCBs"*, GSI Darmstadt, Germany, Sci. Rep. RESEARCH-PANDA-HAD-8, [Online]. 2017. Available: http://dx.doi.org/10.15120/GSI-2017-01856

[5] J. Adamczewski-Musch, GosipGUI framework, https://subversion.gsi.de/dabc/drivers/mbspex/gui/qt-mainwindow/