GEM SW report

Camilla Galloni for the SW team

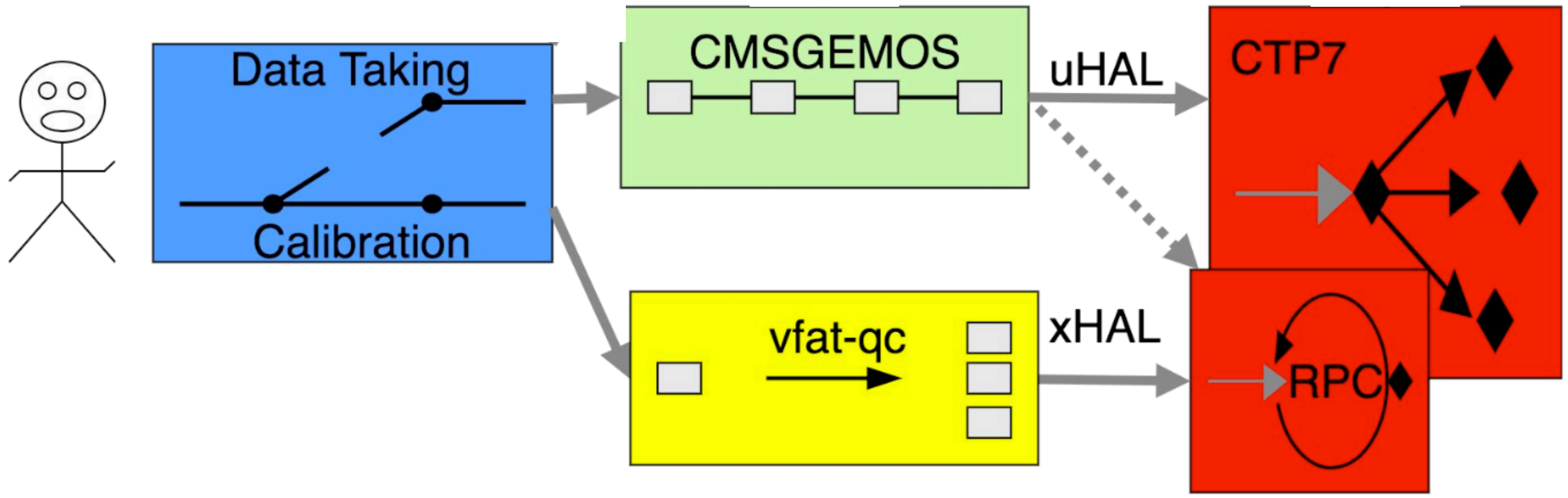
GEM Workshop XXIV
1st Oct 2019

Overview of the current SW

- Main core of the GEM online software is the package cmsgemos which is a xDAQ based framework that includes
  - HW interface with uhal::HwDevice (IPBus) member objects
  - Generic <HW>Manager classe, which controls actions for various state transitions and manages the state of multiple <HW>Device objects of the same class (e.g.,OptoHybrid)
  - GEMSupervisor class, which controls the <HW>Manager applications
    - Ensures the correct transition order depending on the HW and transition type
- Simultaneously developed python code, which is used for rapid prototyping before integration into the functionality of the C++ code
  - Became of daily use for QC and calibration operations
    - Connectivity testing
    - Calibrations of front-end chip (VFAT3)
    - Configuration of front-end electronics for data-taking

Architecture: now

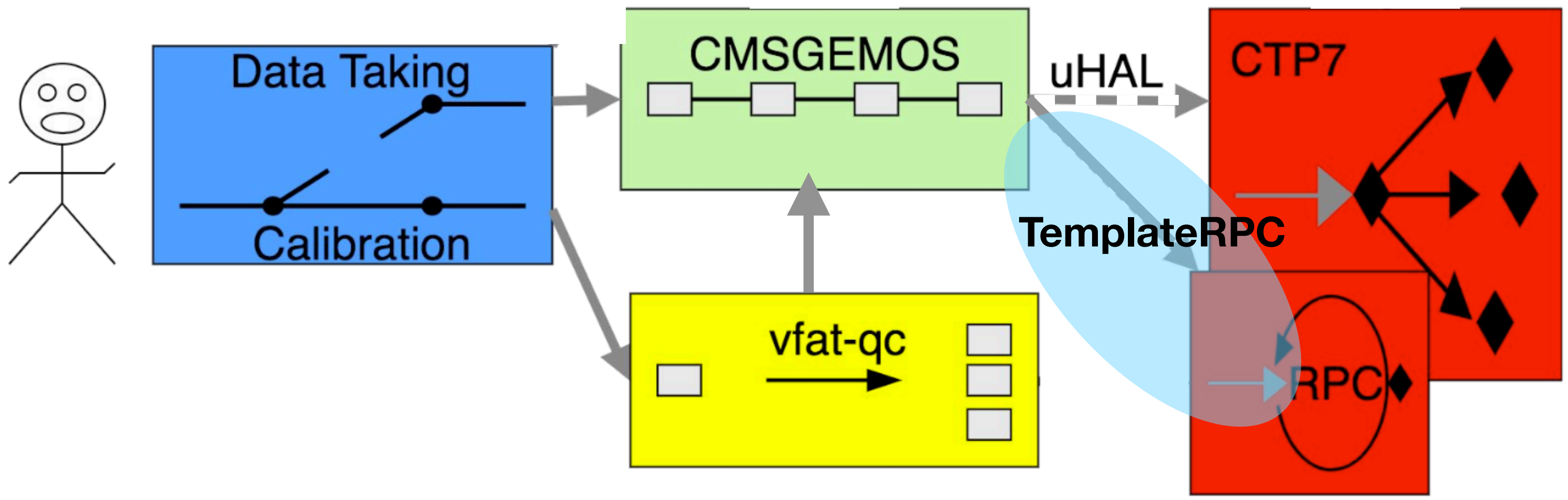
- Main core of the GEM online software is the package `cmsgemos` which is a `xDAQ` based framework that features
  - HW interface with `uhal::HwDevice (IPBus)` member objects
  - vfat-qc functionalities use `xhal` to replace the IPBus transactions with Remote Procedure Calls (RPC) to the `ctp7_modules`



Dalchenko's presentation

Architecture: now and new

- Main core of the GEM online software is the package `cmsgemos` which is a `xDAQ` based framework that features
  - HW interface with `uhal::HwDevice (IPBus)` member objects
  - vfat-qc functionalities use `xhal` to replace the IPBus transactions with Remote Procedure Calls (RPC) to the `ctp7_modules`



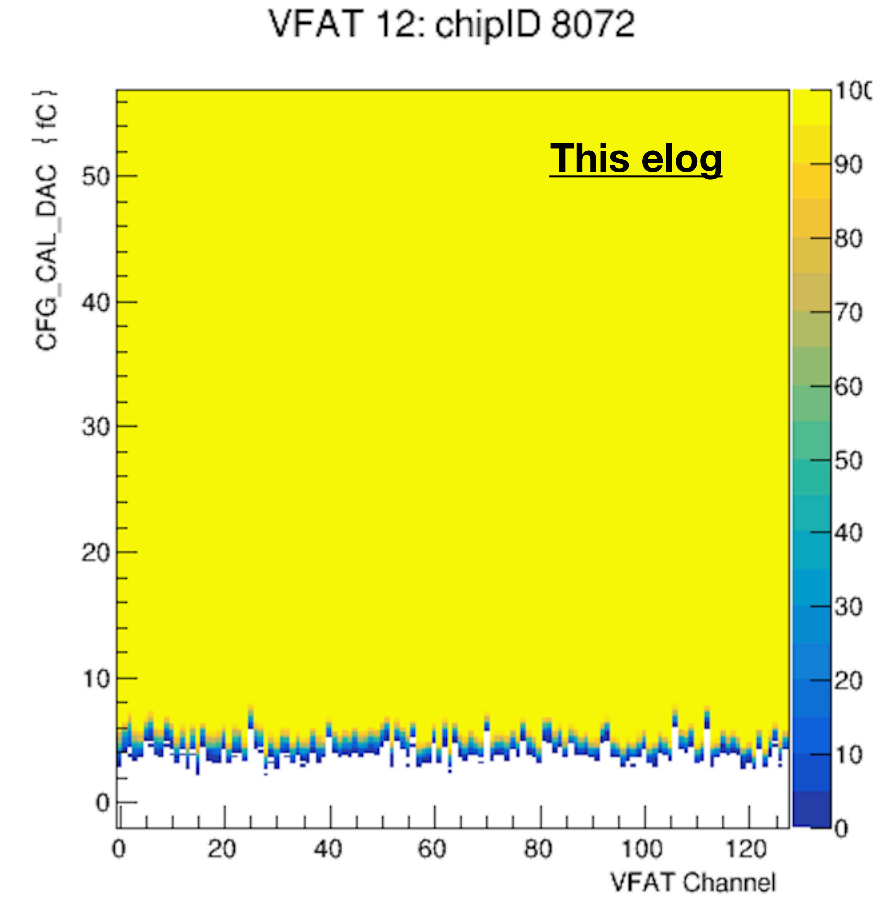
Dalchenko's presentation

- New architecture: HW interface inherits from IPBus device and `xhal` device that connect to RPC modules, which allow to move more extensively part of the code to the remote location (CTP7 Zynq CPU)
  - Core functionality is implemented in libraries executed remotely via RPC in which register actions are defined for various operations with the HW
  - `vfat-qc` functionalities imports the HW communication classes from `cmsgemos`
  - TemplateRPC are used to remove function definition duplication

Current status of the QC7/QC8 SW

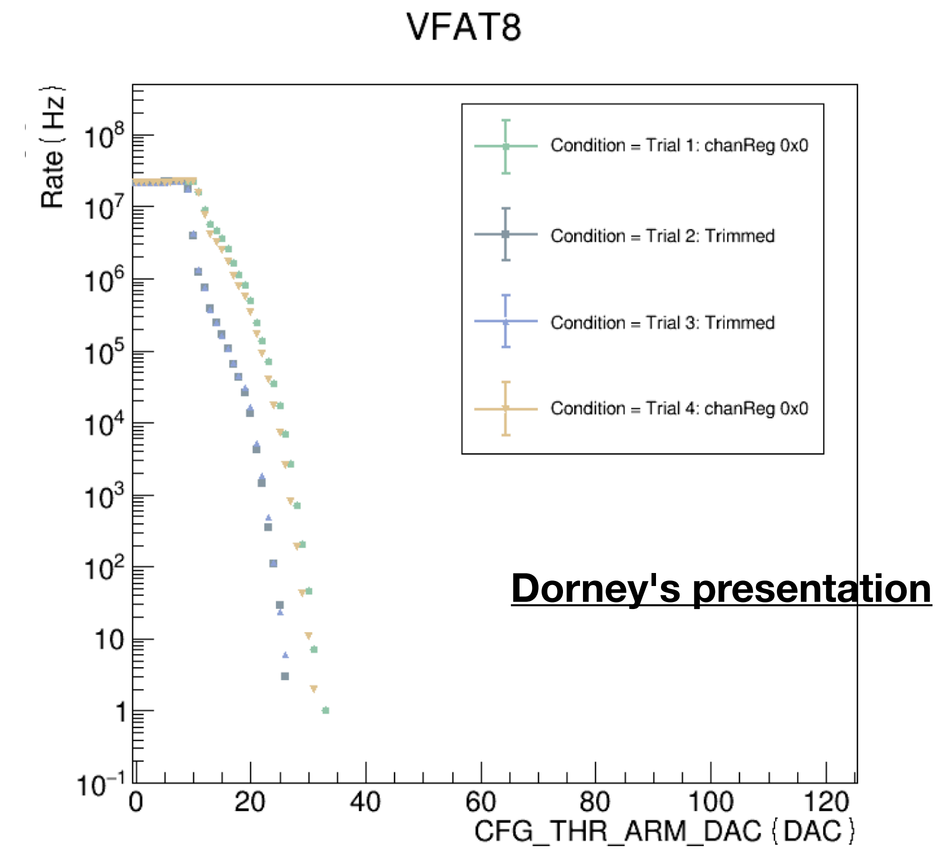
- The GEM DAQ expert at QC7 and QC8 stand, currently are able to perform (vfatqc-python-scripts):
  - Connectivity testing
  - Calibration of front end electronics parameters
  - ENC measurement with s-curves
  - Latency scans
  - CFG_THR_ARM_DAC scans for the identification of hot and dead channels and check sbit lines
  - Identification of disconnected channels with sbits

Currently in use



- Trimming is also possible:
  - Have a uniform response of the VFAT channels
  - Help in reducing the thresholds to be applied

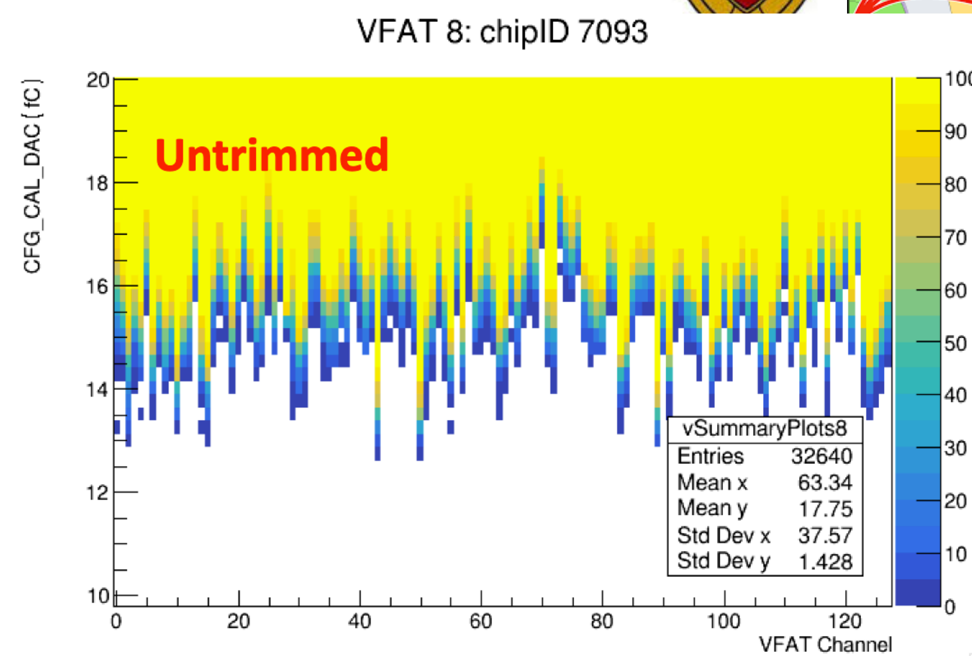
Preliminary version available, but final version under development (ready in October)

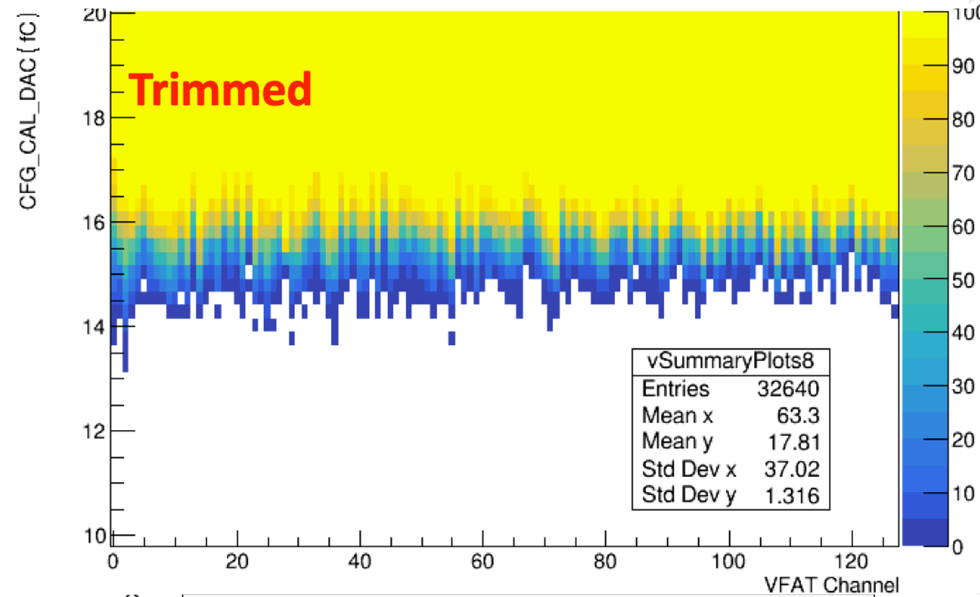


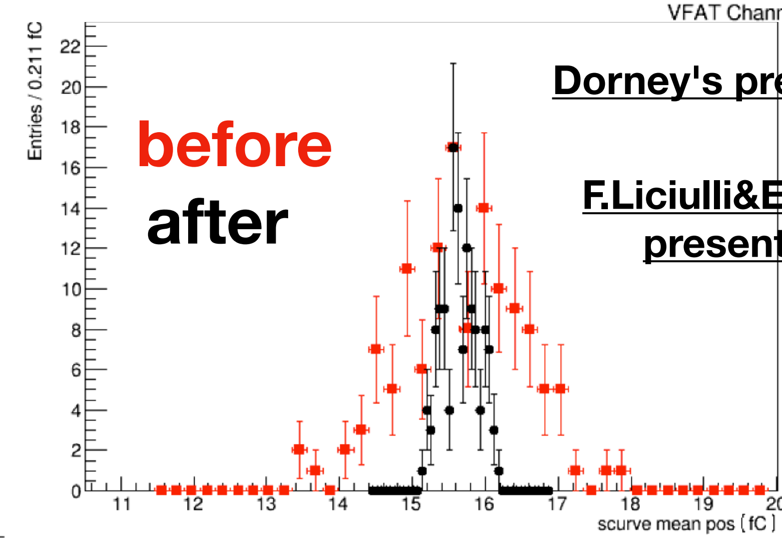
All ingredients for future operations at P5 are already in place, just need to be ported

Iterative trimming

- In order to equalize the threshold of the VFAT3 channels:
  - 1.Scurve measurement and fitting to measure the threshold value for each channel (μ_c).
  - 2.Calculate the mean value (μ_m) of the measured thresholds.
  - 3.Calculate the trimming value (v_T) for the 6 bit local DAC. For each channel apply the following formula:
    - $v_T = (\mu_m - \mu_c) * 15 \text{ (mV/fC)} / 1 \text{ mV}$
  - 4.Round the v_T in order to have an integer value, limit the obtained number to the interval [-63, 63].
  - 5.Setting the 6 bit local DAC registers to v_T
  - 6.Repeat other 2 times (no improvement seen for more iterations)
- Distribution of scurve means before trimming & after 3rd iteration: more uniform response





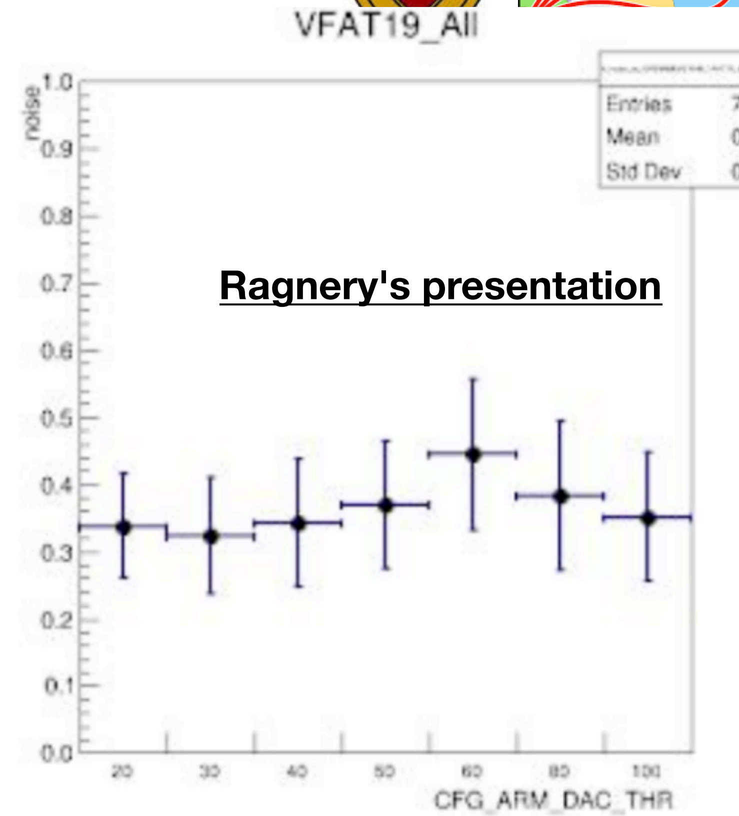


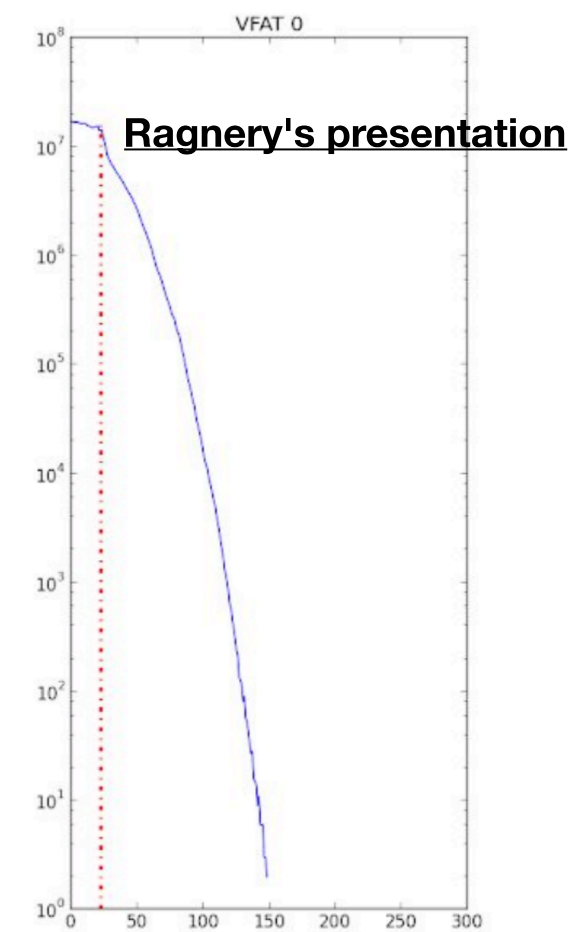
Dorney's presentation

F.Liciulli&E.Juska's presentation

Calibrations development

- **Ongoing**: studying the dependency, stability and validity of a set of trimming values upon different CFG_ARM_DAC_THR values
- The CFG_ARM_DAC_THR is also calibrated by repeating scurves for different values of the threshold on the comparator
- A possible improvement to provide a faster way to do the CFG_ARM_DAC_THR calibration is to use the S-Bit rate scan.
  - The inflection point in each S-Bit scan may be the best starting point to use in ARM DAC calibration
- **Ongoing**: development of an algorithm that exploits the gradient of the sbit curve to find this point.
- **Ongoing**: study to increase statistics in QC tools to understand better the tails

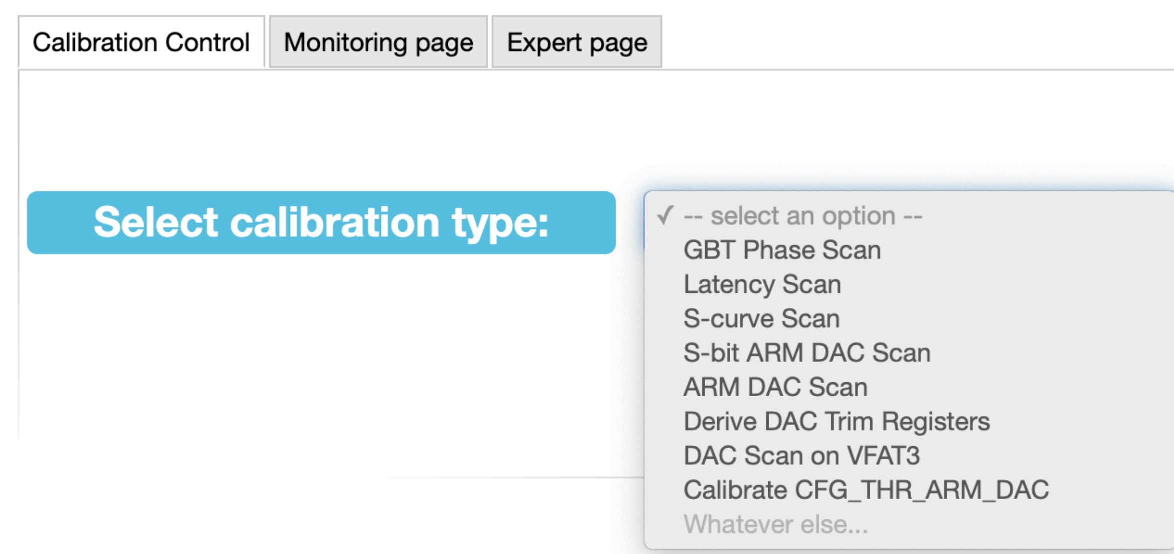




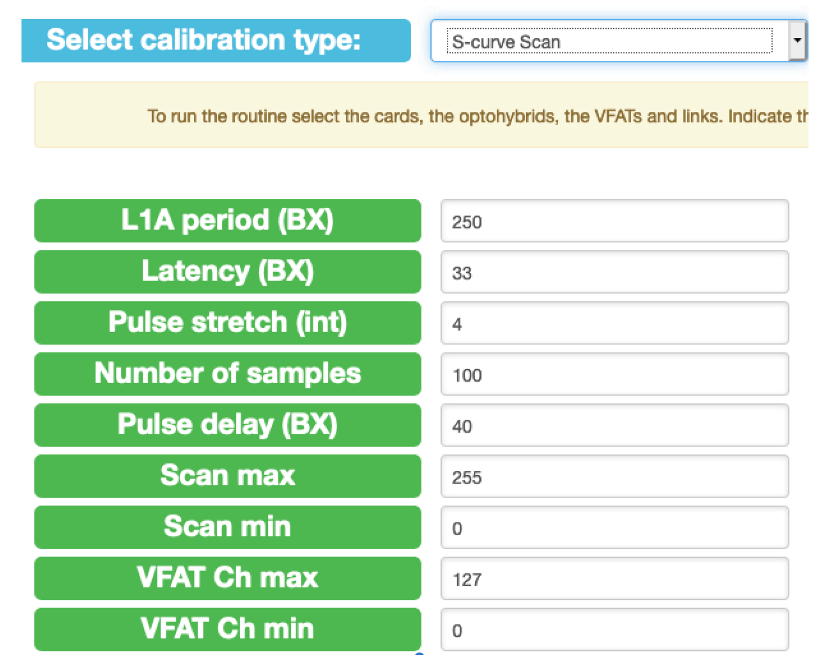
XDAQ integration for calibrations

- The current software used by the QC7 and QC8 team will have to be adapted for the operations at P5, in order to be compatible with XDAQ.
- A calibration suite is in preparation in order to be able to select the type of calibration, the main set of parameters and run the calibration routine, with few clicks.

- The calibration interface is handled by a component of `cmsgemos` (`gemcalibration`) as a plugin for XDAQ



- The user interface is almost ready
- Parameters are correctly retrieved and sent to the appropriate `<HW>Manager` applications.
- Implementation of the scan routines is ongoing and will be finalized in a couple of months (templateRPC)



TemplateRPC

- Restructuring of the RPC interface to avoid:
  - building requests by hand
  - unpack return values by hand
  - termination of the server or client for missing keys or exceptions
  - Triplication of code: `cmsgemos/xhal/ctp7_modules`
    - difficult to maintain in the long term, especially with different architectures/boards
- The templated RPC framework is part of a bigger refactoring which aims at providing an abstraction of the underlying remote call service
- the idea is to leverage the **C++ templates** in order to:
  - define the methods only once for the ctp7_`modules`
  - then invoke the function in `cmsgemos`
- All the processes of serialization, types checking and C++ exceptions catching is done under the hood by the C++ compiler.

L.Petre's presentation

L.Moureaux's presentation

TemplateRPC

- In order to complete the decoupling of the current code base from the CTP7-specific `rpcsvc` daemon, two functionalities have already been improved:
  - 1. The logging system had to be replaced: `log4cplus` was chosen(link)
  - 2. The connection to the local LMDB database, which stores the firmware registers, has been improved (link)
- **Ongoing** (1-2 months): the portage from the current system to the new framework in the `ctp7_modules` and `cmsgemos` repositories.

L.Petre's presentation

L.Moureaux's presentation

Database

- The configuration database will store the detector setting used for configuring chambers and record it for analysis.
- The layout of the configuration DB follows a scheme that was initially designed for the Pixel, that had to be understood and readapted
  - GEM specific tables were also refactored by the DB team

- In the software stack, the DB is handled by a component of `cmsgemos` (`gemonlinedb`) that has two backends:
  - one based on XML files: almost ready
  - one that connects to the DB: still in development (to be done once the DB layout is finalized)

Other developments

- Effort ongoing to provide software documentation:
    - scaffolding with sphinx (python), doxygen (C/C++) and Breath for online display of SW guide (link)
- Tree translator to convert xdaq raw data (event based) into a gemTreeStructure format (point based) to make new data format compatible with existing analysis tools (link)
- Working also at having code compatibility for the GE1/1, GE2/1 and ME0 detector type (link)

Summary and timeline

- Current work for the amelioration of the `python` SW for the QC7 and QC8 and calibration operations was presented.
  - Main functionalities are stable and available also for future operations at P5.

- Integration of calibration tools into a “calibration suite” (`cmsgemos`): ongoing
  - will deploy the current functionality in development for the QC
- Integration with configuration DB (`cmsgemos`): ongoing

- Refactor `ctp7_modules` for long-term maintainability: ongoing
  - Use a template based approach to reduce the amount of code that needs to be updated/changed whenever a module is updated/added/removed
  - Ensure modular compatibility with GE2/1 HW

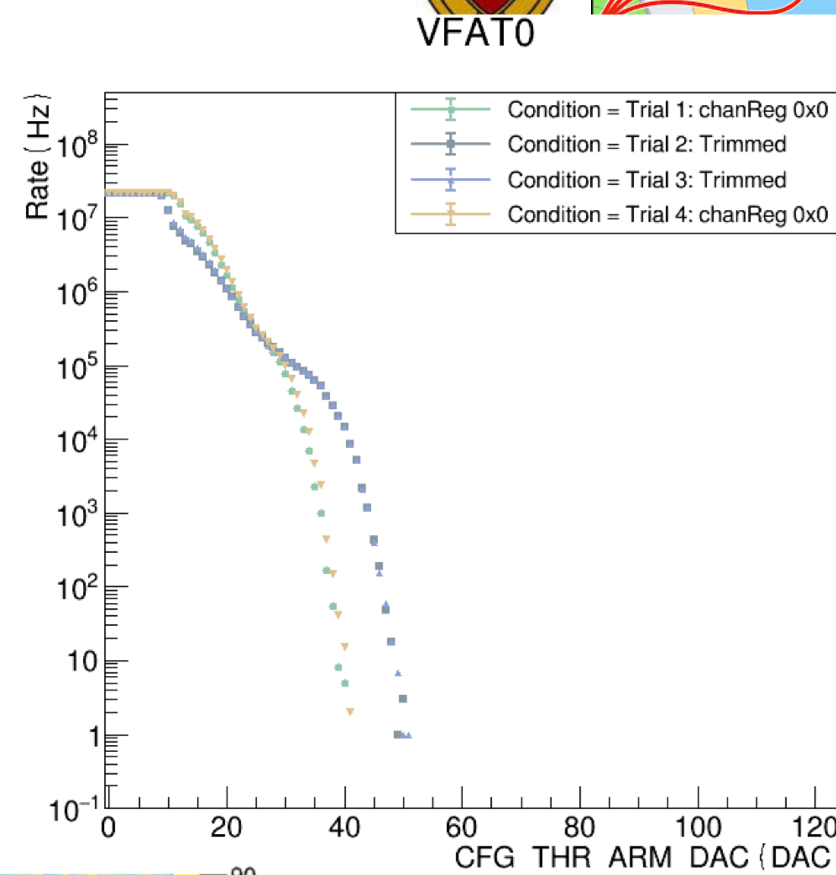
- Monitoring and alarm framework update: planned, but pending upon previous points

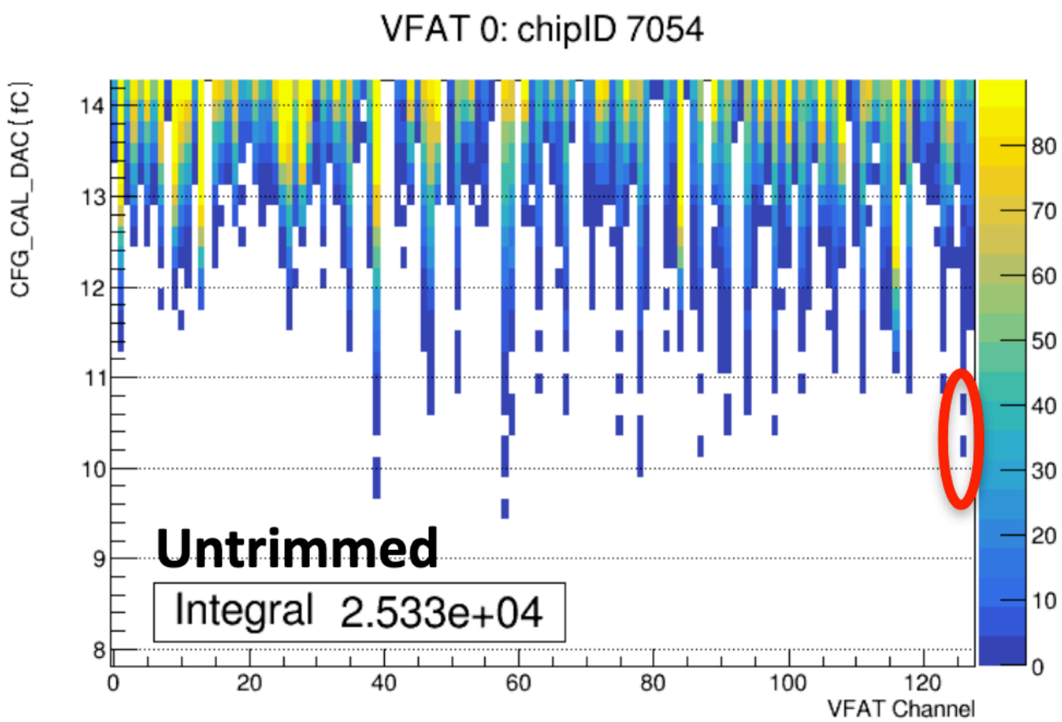
- Target: usage of these updates during full endcap commissioning in 2020

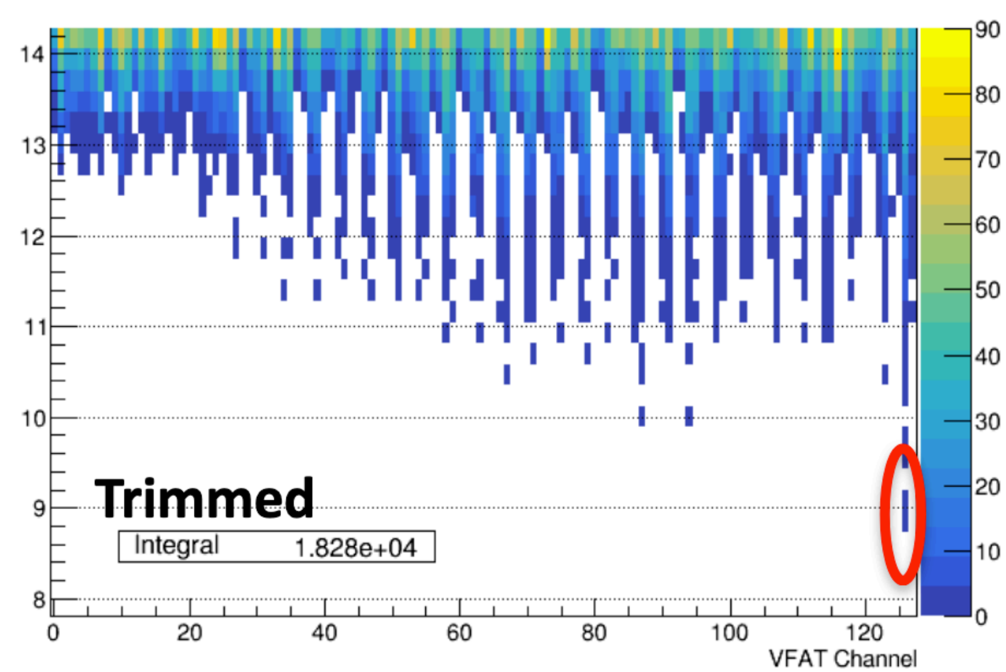
Back up

Iterative trimming (II)

- A hot channel could spoil the batch
- Trimming will push scurves around but not change width
- If wide scurve at higher threshold is pushed to lower threshold by trimming it will act as a hot channel later







Dorney's presentation