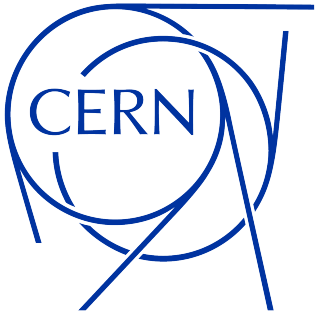


European Organization for Nuclear Research(CERN)

OPC-UA LAr LpGBT status

LAPP

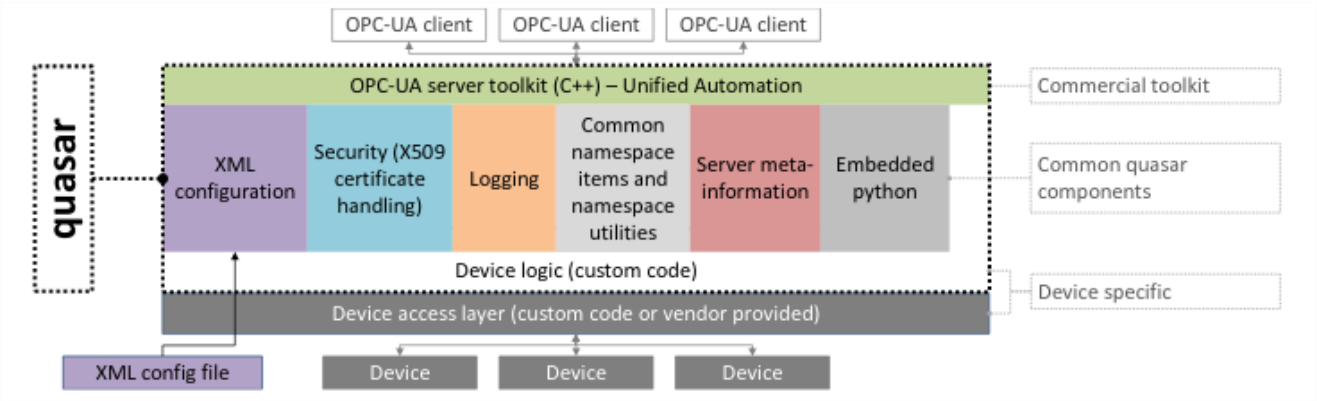
March 20, 2025

Etienne FORTIN, Olivier ARNAEZ

Overview

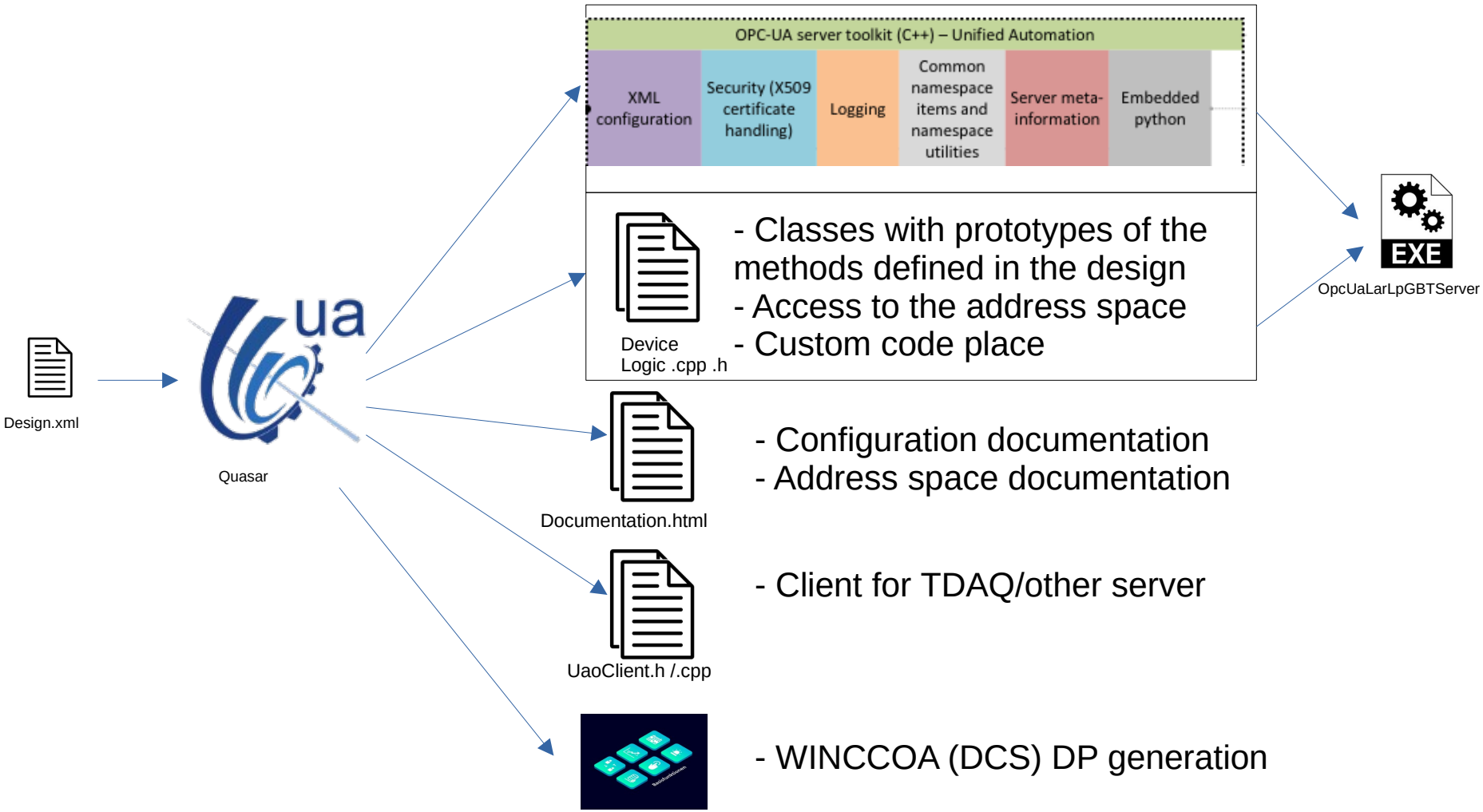
Quasar Framework

Quick OPC-UA Server Generation Framework

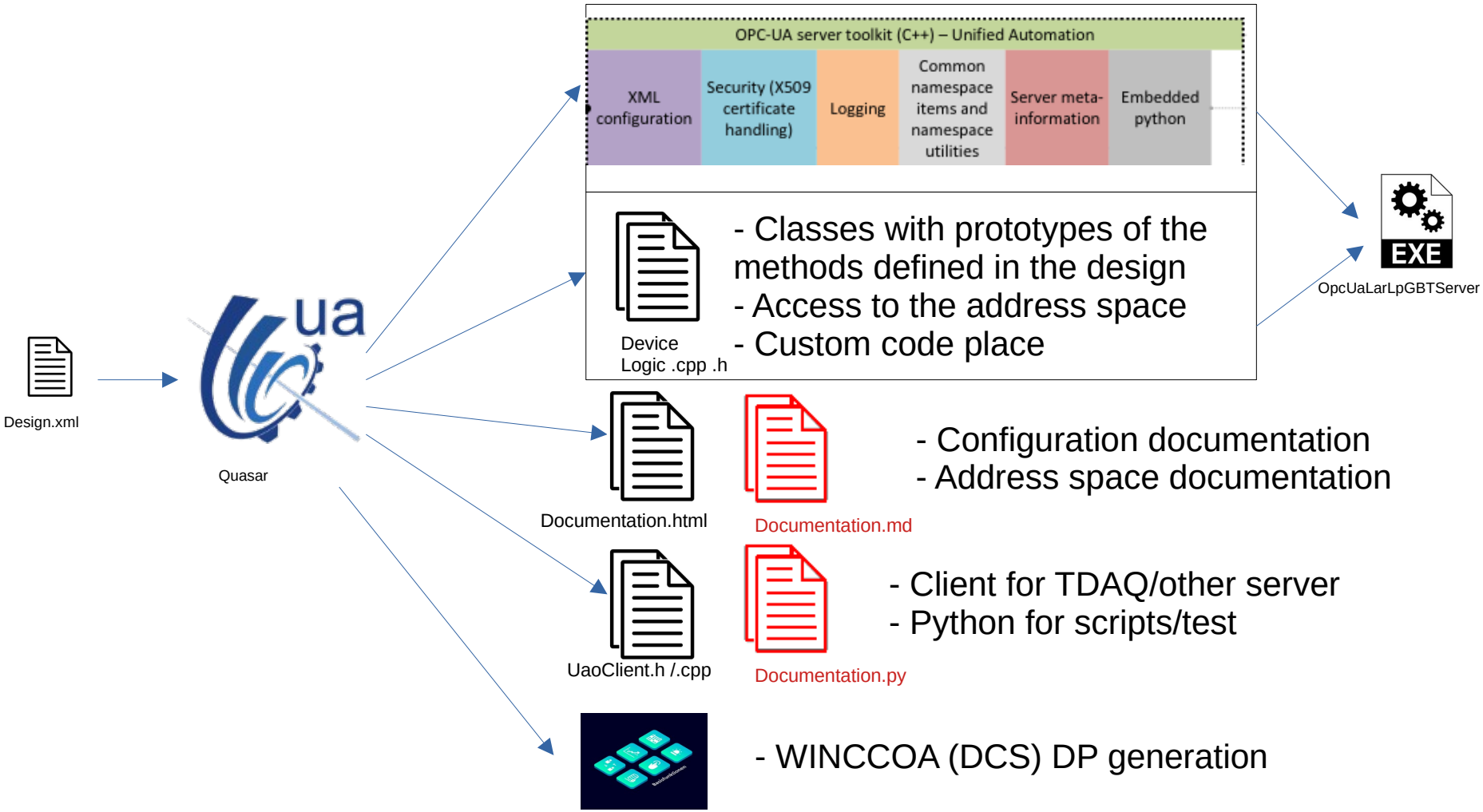


- Ecosystem for the devellopment of OPC-UA
  - in OPC-UA a varaible is always value + timestamp + validity
- Initially devellopped by ATLAS DCS for generation of OPC-UA server
- The devellopers focus on the "Device logic" and "Device Access layer"
  - No devellopment work needed on OPC-UA protocol and server
- Maintened by a collaboration of CERN-JCOP(Joint COntrols Project) , ATLAS central DCS, me...
- Used in ATLAS,LHCb,CMS,BEAM,DUNE ...
- Collaboration agrement with CAEN,ISEG,Wiener.

Quasar generation



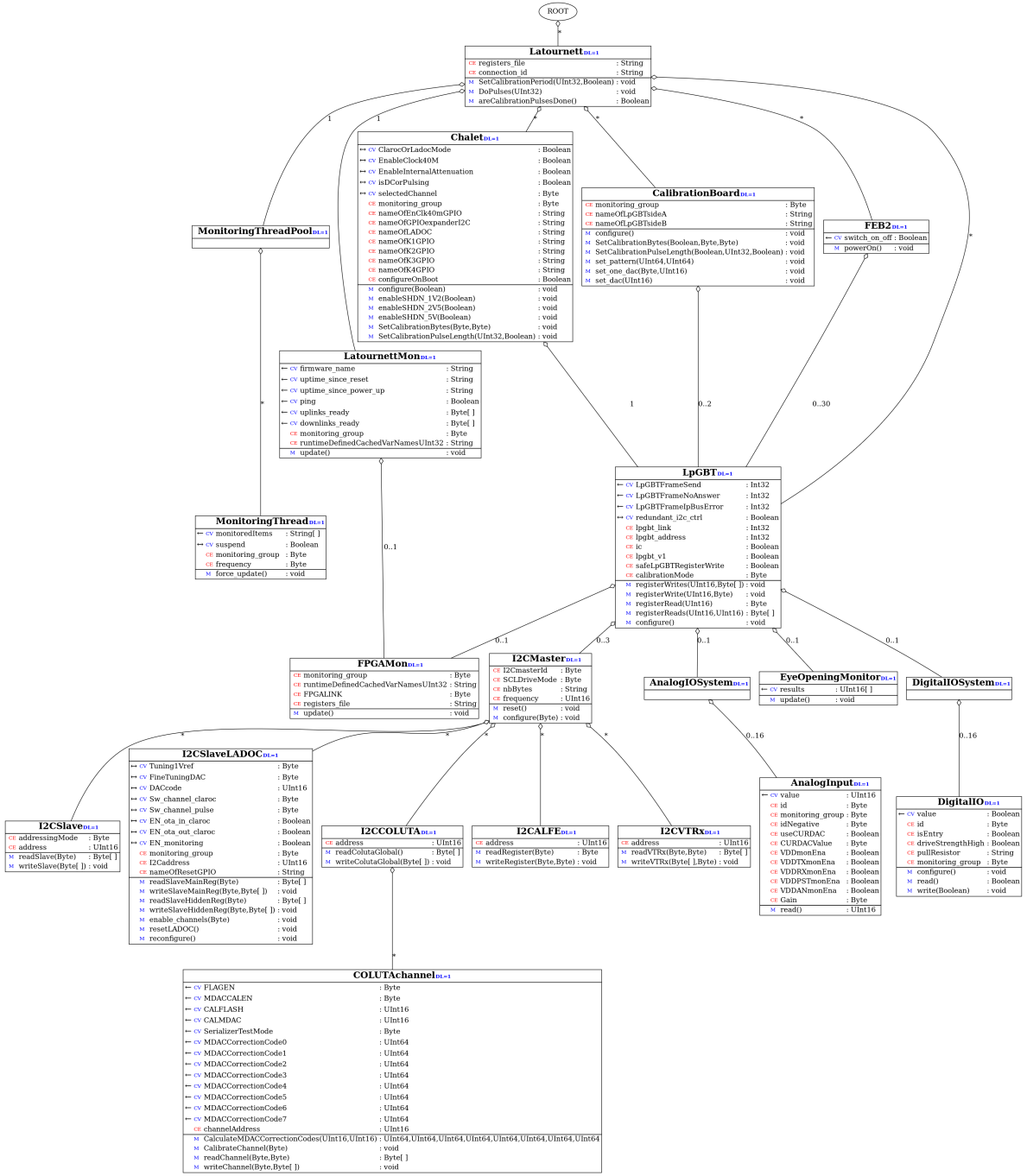
Quasar generation, modification for LAr ph-II



Documentation integrated in quasar v1.7 ...

Specification of the server

- PH-II operation manual : Link to CDS
- A unique server will implement Atomic functions with elementary low level functionalities that can be used to perform higher level actions such as monitoring, configurations, and scans.
- Atomic functions we mean function that perform that need continuous uninterrupted discussion with the hardware e.g. a sequence of I2C commands to write a specific register.
- Clashes between DCS and TDAQ actions will be handled in the server.
- Low level retries will be also handled in the server
- TDAQ will use these functions to implement the needed configurations and scans. The configurations and scans that are the responsibility of DCS will be implemented mainly in the server


ROOT
Latournett DL=1
CE registers_file : String
CE connection_id : String
M SetCalibrationPeriod(UInt32,Boolean) : void
M DoPulses(UInt32) : void
M areCalibrationPulsesDone() : Boolean
Chalet DL=1
CV ClarocOrLadocMode : Boolean
CV EnableClock40M : Boolean
CV EnableInternalAttenuation : Boolean
CV isDCorPulsing : Boolean
CV selectedChannel : Byte
CE monitoring_group : Byte
CE nameOfEnClk40mGPIO : String
CE nameOfGPIOexpanderI2C : String
CE nameOfLADOC : String
CE nameOfK1GPIO : String
CE nameOfK2GPIO : String
CE nameOfK3GPIO : String
CE nameOfK4GPIO : String
CE configureOnBoot : Boolean
M configure(Boolean) : void
M enableSHDN_1V2(Boolean) : void
M enableSHDN_2V5(Boolean) : void
M enableSHDN_5V(Boolean) : void
M SetCalibrationBytes(Byte,Byte) : void
M SetCalibrationPulseLength(UInt32,Boolean) : void
CalibrationBoard DL=1
CE monitoring_group : Byte
CE nameOfLpGBTsideA : String
CE nameOfLpGBTsideB : String
M configure() : void
M SetCalibrationBytes(Boolean,Byte,Byte) : void
M SetCalibrationPulseLength(Boolean,UInt32,Boolean) : void
M set_pattern(UInt64,UInt64) : void
M set_one_dac(Byte,UInt16) : void
M set_dac(UInt16) : void
FEB2 DL=1
CV switch_on_off : Boolean
M powerOn() : void
MonitoringThreadPool DL=1
LatournettMon DL=1
CV firmware_name : String
CV uptime_since_reset : String
CV uptime_since_power_up : String
CV ping : Boolean
CV uplinks_ready : Byte[ ]
CV downlinks_ready : Byte[ ]
CE monitoring_group : Byte
CE runtimeDefinedCachedVarNamesUInt32 : String
M update() : void
MonitoringThread DL=1
CV monitoredItems : String[ ]
CV suspend : Boolean
CE monitoring_group : Byte
CE frequency : Byte
M force_update() : void
LpGBT DL=1
CV LpGBTFrameSend : Int32
CV LpGBTFrameNoAnswer : Int32
CV LpGBTFrameIpBusError : Int32
CV redundant_i2c_ctrl : Boolean
CE lpgbt_link : Int32
CE lpgbt_address : Int32
CE ic : Boolean
CE lpgbt_v1 : Boolean
CE safeLpGBTRegisterWrite : Boolean
CE calibrationMode : Byte
M registerWrites(UInt16,Byte[ ]) : void
M registerWrite(UInt16,Byte) : void
M registerRead(UInt16) : Byte
M registerReads(UInt16,UInt16) : Byte[ ]
M configure() : void
FPGAMon DL=1
CE monitoring_group : Byte
CE runtimeDefinedCachedVarNamesUInt32 : String
CE FPGALINK : Byte
CE registers_file : String
M update() : void
I2CMaster DL=1
CE I2CmasterId : Byte
CE SCLDriveMode : Byte
CE nbBytes : String
CE frequency : UInt16
M reset() : void
M configure(Byte) : void
AnalogIOSystem DL=1
EyeOpeningMonitor DL=1
CV results : UInt16[ ]
M update() : void
DigitalIOSystem DL=1
I2CSlaveLADOC DL=1
CV Tuning1Vref : Byte
CV FineTuningDAC : Byte
CV DACcode : UInt16
CV Sw_channel_claroc : Byte
CV Sw_channel_pulse : Byte
CV EN_ota_in_claroc : Boolean
CV EN_ota_out_claroc : Boolean
CV EN_monitoring : Boolean
CE monitoring_group : Byte
CE I2Caddress : UInt16
CE nameOfResetGPIO : String
M readSlaveMainReg(Byte) : Byte[ ]
M writeSlaveMainReg(Byte,Byte[ ]) : void
M readSlaveHiddenReg(Byte) : Byte[ ]
M writeSlaveHiddenReg(Byte,Byte[ ]) : void
M enable_channels(Byte) : void
M resetLADOC() : void
M reconfigure() : void
I2CSlave DL=1
CE addressingMode : Byte
CE address : UInt16
M readSlave(Byte) : Byte[ ]
M writeSlave(Byte[ ]) : void
I2CCOLUTA DL=1
CE address : UInt16
M readColutaGlobal() : Byte[ ]
M writeColutaGlobal(Byte[ ]) : void
I2CALFE DL=1
CE address : UInt16
M readRegister(Byte) : Byte
M writeRegister(Byte,Byte) : void
I2CVTRx DL=1
CE address : UInt16
M readVTRx(Byte,Byte) : Byte[ ]
M writeVTRx(Byte[ ],Byte) : void
AnalogInput DL=1
CV value : UInt16
CE id : Byte
CE monitoring_group : Byte
CE idNegative : Byte
CE useCURDAC : Boolean
CE CURDACValue : Byte
CE VDDmonEna : Boolean
CE VDDTXmonEna : Boolean
CE VDDRXmonEna : Boolean
CE VDDPSTmonEna : Boolean
CE VDDANmonEna : Boolean
CE Gain : Byte
M read() : UInt16
DigitalIO DL=1
CV value : Boolean
CE id : Byte
CE isEntry : Boolean
CE driveStrengthHigh : Boolean
CE pullResistor : String
CE monitoring_group : Byte
M configure() : void
M read() : Boolean
M write(Boolean) : void
COLUTAchannel DL=1
CV FLAGEN : Byte
CV MDACCALEN : Byte
CV CALFLASH : UInt16
CV CALMDAC : UInt16
CV SerializerTestMode : Byte
CV MDACCorrectionCode0 : UInt64
CV MDACCorrectionCode1 : UInt64
CV MDACCorrectionCode2 : UInt64
CV MDACCorrectionCode3 : UInt64
CV MDACCorrectionCode4 : UInt64
CV MDACCorrectionCode5 : UInt64
CV MDACCorrectionCode6 : UInt64
CV MDACCorrectionCode7 : UInt64
CE channelAddress : UInt16
M CalculateMDACCorrectionCodes(UInt16,UInt16) : UInt64,UInt64,UInt64,UInt64,UInt64,UInt64,UInt64,UInt64
M CalibrateChannel(Byte) : void
M readChannel(Byte,Byte) : Byte[ ]
M writeChannel(Byte,Byte[ ]) : void

LpGBT based classes

- LpGBT,I2CMaster,AnalogInput,EyeOpenningMonitor, etc...
- Give abstraction over the functionality of the LpGBT chips.
- Client develloper don't need to know the details of the LpGBT
- Same LpGBT for calibration and FEB2 -> No specific function

I2C slave classes

- As almost all our I2C chips use indirect addressing:
  - Several I2C frame are needed to address 1 register
  - To keep the **atomic functions** of the specification: specific class in the server
- Provide method to read/write registers

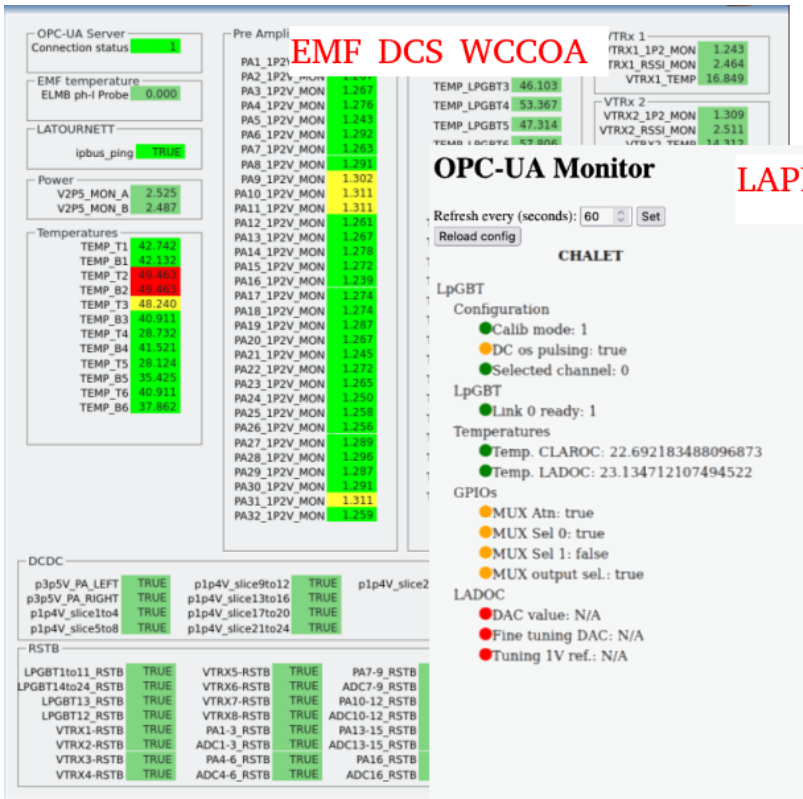
Board level classes

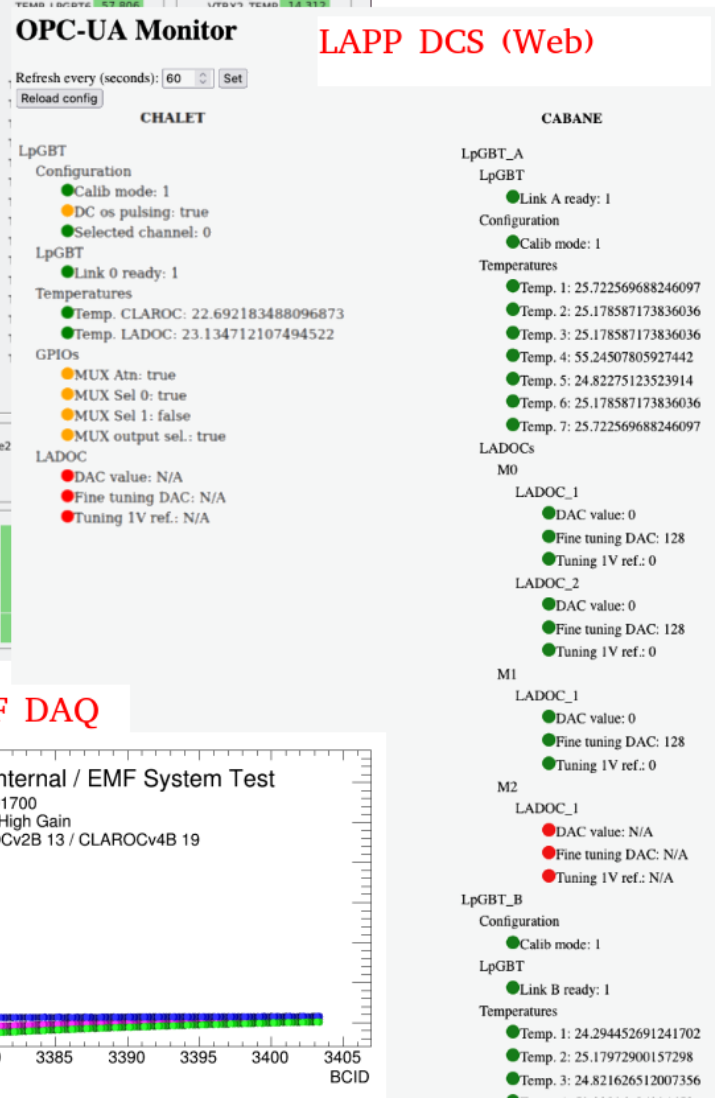
- FEB2: host DCS functionnalities
- CALIB: host DCS functionalities + some LATOURNETT functionalitis for the calibration

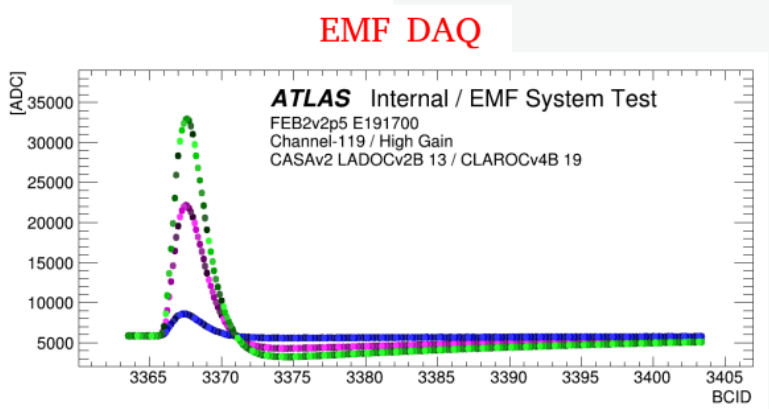
Monitoring

- DCS monitoring : handled by a threadpool on the server
  - Reading on the board and writting to cache variable
- TDAQ monitoring:
  - By method, direct reading of the values
  - Can also use the threadpool

Integration







Status

- On the server:
  - All the mains functionalities are made all boards are configured by the server at EMF everyday.
  - Prototype presented at LAr week in mars 2024, and accepted by steering to go to production.
  - Mising one class for the I2C chip for HEC
  - Several optimisations are possible
  - Support of Real calibration/latournett boards
  - Uniformisation of error code
- On the clients:
  - FEB2:
    - Client using the real functionalities of the server to configure the board
    - Partition and cpp client
  - Calibration:
    - Partition and cpp client
    - DCS Winccoa client
- At we need more HW to test in real condidition EMF:
  - several FEB2
  - calibration board
  - latournett board