Phase II Acquisition & Control Framework - A User's Guide

Fabio Ravera

Phase II Tracker DAQ workshop

8 October 2019

Plan for today

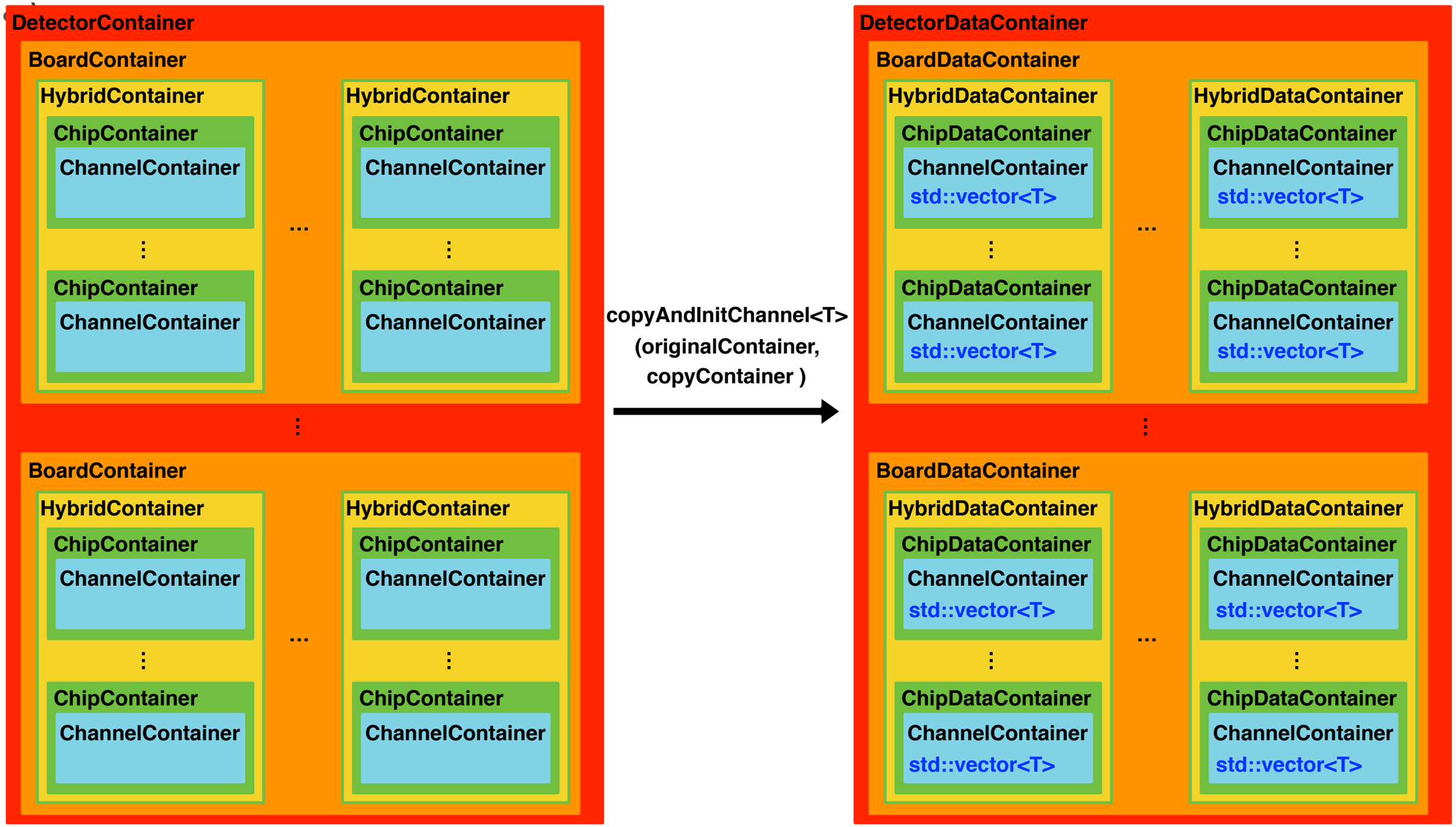
- Introduction on how to allocate memory for the calibration
- A calibration example
- Your exercise

- By the end of the day you should be able to write a calibration procedure

Memory management - Container class

When the configuration file is parsed the detector structure is loaded into memory:

When you need to store memory you use copyAndInit(Channel, Chip, Module, Board):



Container are vectors with more functions, imagine them as vector of vectors -> access them with for loops

Fermilab

Creating the needed memory allocation with Containers

- Step 1: create a DetectorDataContainer object:
  - DetectorDataContainer myContainer;
  - (almost) no memory allocated so far

- Step 2: create the tree structure representing the detector and allocate the memory required:
  - ContainerFactory::copyAndInitChannel<uint32_t>(*fDetectorContainer, myContainer);

Allocate a memory slot for all the channels (strip or pixel) of your detector	As for an std::vector, you write here the data type to allocate for each channel	The detector container from which the HW tree structure is copied

- data type can be whatever you want, but variables not allocated in the stack cannot be automatically streamed. If you foresee to stream data out from the SoC avoid pointers, vectors, strings…
- In Utils you can find some classes you by use to allocate more complex variables: Occupancy, OccupancyAndPh, ThresholdAndNoise, GenericVariableArray, etc

Creating the needed memory allocation with Containers - II

- Allocation of memory can be done for all layers of the tree:
  - ContainerFactory::copyAndInitChip<uint32_t>(*fDetectorContainer, myContainer);
  - ContainerFactory::copyAndInitHybrid<uint32_t>(*fDetectorContainer, myContainer);
  - ContainerFactory::copyAndInitOpticalGroup<uint32_t>(*fDetectorContainer, myContainer);
  - ContainerFactory::copyAndInitBoard<uint32_t>(*fDetectorContainer, myContainer);
  - ContainerFactory::copyAndInitDetector<uint32_t>(*fDetectorContainer, myContainer);

- Initialization can be provided:
  - uint32_t theStartingPoint = 0;
  - ContainerFactory::copyAndInitChip<uint32_t>(*fDetectorContainer, myContainer, theStartingPoint);

- Same container can handles memories for all the levels
  - ContainerFactory::copyAndInitStructure<ChannelType, ChipType, HybridType, OpticalGroupType, BoardType, DetectorType>(*fDetectorContainer, myContainer);
  - ContainerFactory::copyAndInitStructure<TypeForAll>(*fDetectorContainer, myContainer);
    - a TypeForAll variable is allocated for all the tree levels

Access data from Containers - I

- You can loop over containers as for vectors (they inherit from vectors):

```
for(auto board : myContainer) //loop on boards - begin
{
    for(auto opticalGroup : *board) //loop on opticalGroups - begin
    {
        for(auto hybrid: *opticalGroup) //loop on hybrid - begin
        {
            for(auto chip: *hybrid) //loop on chip - begin
            {
                for(auto channel : *chip->getChannelContainer<uint32_t>()) //loop on channel - begin
                {
                    // do something
                } //loop on channel - end
            } //loop on chip - end
        } //loop on hybrid - end
    } //loop on opticalGroups - end
} //loop on boards - end
```

- You can also access elements by index

```
auto chip = myContainer.at(boardIndex)->at(opticalGroupIndex)->at(hybridIndex)->at(chipIndex);
```

- Two containers are two totally parallel trees, same index = same HW

```
size_t chipIndex = chip->getIndex();
```

Access data from Containers - II

- Important to remember: due to technicalities, channels are handled in a different way with respect to all other layers

  - uint32_t &theChannelVariable = chip->getChannel<uint32_t>(row, col)
    - Get memory for Channel (row,col). For strips, simply just provide one argument
    - Note!!! uint32_t is what the channel type is, if you put a wrong type bad things happen

  - uint32_t &theChipVariable = chip->getSummary<uint32_t>()
    - Get memory for a variable associated to the chip
    - Same care in providing the Summary type

- Note: when a container contains data also the lower level from which you are retrieving the information, you should use
  - LevelType &theChipVariable = chip->getSummary<LevelType,LowerLevelType>()
  - Technicality which allows nice features like evaluating averages per chip, module, etc for a variable (e.g. you can automatically get the average chip occupancy)

Access HWDescription from Container

- fDetectorContainer (comes from the SystemController) is the way we store all the HWDescription (Board, OpticalGroup, Hybrid, Chip)

- To get HWDescription you need a static_cast:
  - BeBoard* theBeBoard = static_cast<BeBoard*>(fDetectorContainer->at(boardIndex));

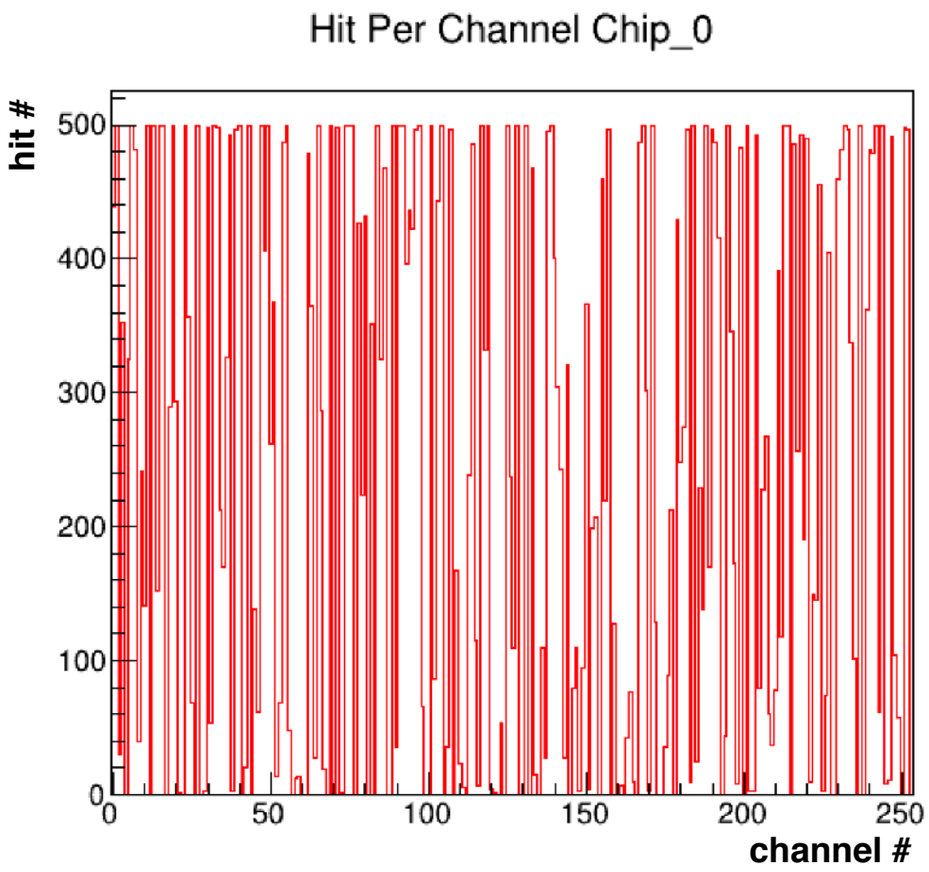
- Same for Chip, just dig down to the Chip Level:
  - Cbc* theCbc = static_cast<Cbc*>(fDetectorContainer->at(boardIndex)
    ->at(opticalGroupIndex)->at(hybridIndex)->at(chipIndex));

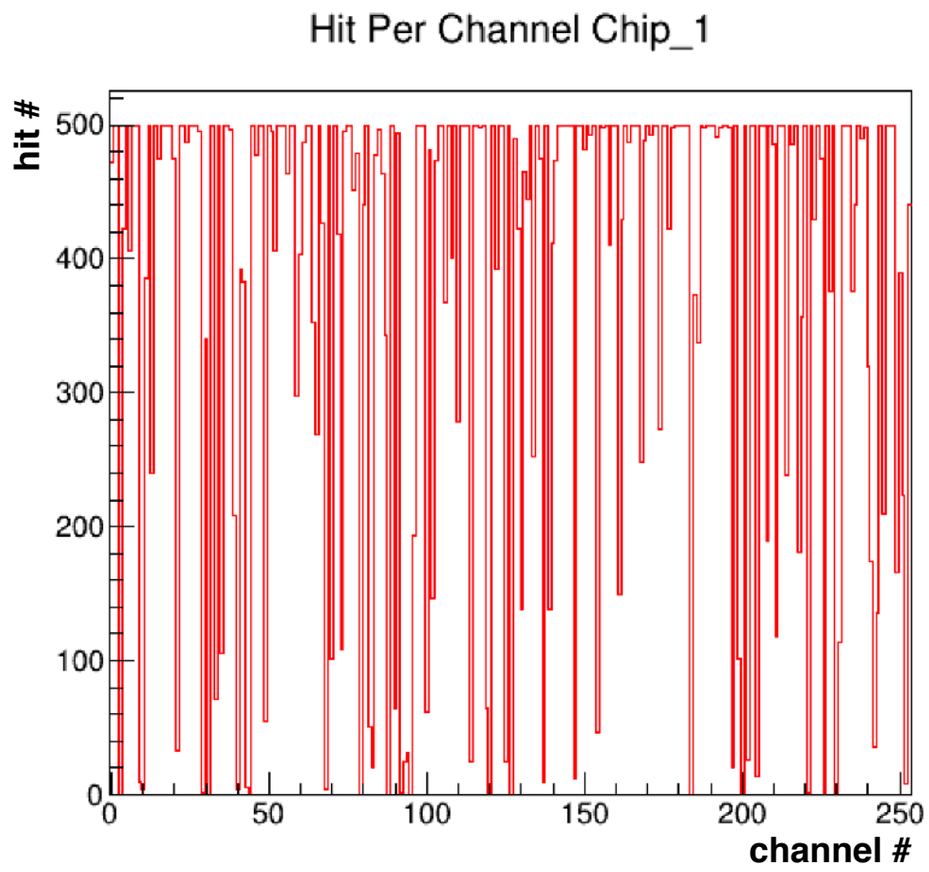
Conventions (not always respected 😤)

- Names must be clear, who reads should immediately understand what you are doing (g++ has no limit of character used for a variable)
- Classes name: UpperCamelCase
- Functions and variables: lowerCamelCase
- Class data members start with and f: e.g. fDetectorContainer
- Strong suggestion: avoid using old style editors like vim or emacs, modern editors comes with tons of helpful features (autocompletion, error check while editing, git integration, integrated debugger like gdb). I would recommend Visual Studio Code that allows also to mount a remote folder.

A calibration example

- What we want to do: measure the number of noise hits per channel:





- What we need:
  - a script to run the calibration
  - a class implementing the code for the calibration
  - a class handling histograms
- Git branch: https://gitlab.cern.ch/cms_tk_ph2/Ph2_ACF/-/tree/daqSchool2021

Script

- src/calibration_example.cc
- Script that allows you to run the calibration
- This script works only without the SoC, do not worry to use Root utilities here
- It parses command line commands, create the base objects to run the calibration, run the calibration, save results and clean up the memory

- How to compile:
  - cd build
  - cmake ..
  - make -j
  - cd ..

- How to run the example
  - calibration_example -f <xml file for your setup> [-withCIC]

Include classes, specify namespaces and init easy logger

```
1   #include <cstring>
2   #include "tools/BackEndAlignment.h"
3   #include "tools/CicFEAlignment.h"
4   #include "../tools/CalibrationExample.h"
5   #include "../Utils/argvparser.h"
6   #include "TROOT.h"
7   #include "TApplication.h"
8   #include "../Utils/Timer.h"
9
10  using namespace Ph2_HwDescription;
11  using namespace Ph2_HwInterface;
12  using namespace Ph2_System;
13  using namespace CommandLineProcessing;
14
15  INITIALIZE_EASYLOGGINGPP
16
17  int main ( int argc, char* argv[] )
18  {
19      //configure the logger
20      el::Configurations conf ("settings/logger.conf");
21      el::Loggers::reconfigureAllLoggers (conf);
```

Include headers:
CalibrationExample is the one we are going to run

Specify namespaces will be used

Init easy logging macros

provide to easy logging levels of printing (INFO, ERROR, DEBUG, etc)

Parsing command line arguments

```
23    ArgvParser cmd;
24
25    // init
26    cmd.setIntroductoryDescription ( "CMS Ph2_ACF calibration example" );
27    // error codes
28    cmd.addErrorCode ( 0, "Success" );
29    cmd.addErrorCode ( 1, "Error" );
30    // options
31    cmd.setHelpOption ( "h", "help", "Print this help page" );
32
33    cmd.defineOption ( "file", "Hw Description File . Default value: settings/Calibration8CBC.xml", ArgvPars
34    cmd.defineOptionAlternative ( "file", "f" );
35
36    cmd.defineOption ( "output", "Output Directory . Default value: Results", ArgvParser::OptionRequiresVal
37    cmd.defineOptionAlternative ( "output", "o" );
38
39    cmd.defineOption ( "batch", "Run the application in batch mode", ArgvParser::NoOptionAttribute );
40    cmd.defineOptionAlternative ( "batch", "b" );
41
42    cmd.defineOption("withCIC", "With CIC. Default : false", ArgvParser::NoOptionAttribute);
43      You, 13 minutes ago • Added backend alignment in exercises
44    int result = cmd.parse ( argc, argv );
45
46    if ( result != ArgvParser::NoParserError )
47    {
48        LOG (INFO) << cmd.parseErrorDescription ( result );
49        exit ( 1 );
50    }
```

Set list of options
write also description of the option that will be printed with the -h argument

Parse command line arguments

Initialize and Configure

```
52  // now query the parsing results
53  std::string cHWFile    = ( cmd.foundOption ( "file" ) ) ? cmd.optionValue ( "file" ) : "setti
54  std::string cDirectory = ( cmd.foundOption ( "output" ) ) ? cmd.optionValue ( "output" ) : "R
55  bool        cWithCIC   = (cmd.foundOption("withCIC"));
56
57  cDirectory += "CalibrationExample";
58  bool batchMode = ( cmd.foundOption ( "batch" ) ) ? true : false;
59
60  TApplication cApp ( "Root Application", &argc, argv );
61
62  if ( batchMode ) gROOT->SetBatch ( true );
63  else TQObject::Connect ( "TCanvas", "Closed()", "TApplication", &cApp, "Terminate()" );
64
65  //create a generic Tool Object, I can then construct all other tools from that using the Inhe
66  //this tool stays on the stack and lives until main finishes - all other tools will update th
67  Tool cTool;
68  std::stringstream outp;
69  cTool.InitializeHw ( cHWFile, outp );
70  cTool.InitializeSettings ( cHWFile, outp );
71  LOG (INFO) << outp.str();
72  outp.str ("");
73  cTool.ConfigureHw ();
74  cTool.CreateResultDirectory ( cDirectory );
75  cTool.InitResultFile ( "CalibrationResults" );
```

Use info retrieved by the command line parser

Start Root TApplication to keep canvases until an interrupt is issued

Create a Tool object and use the Initialize functions to parse the xml file to retrieve HW and Setting configuration

Easy logging print if level is INFO

Configure the HW (upload registers to boards and chips) and create a folder for results

Fermilab

Set up communication

```
77      Timer t;
78      t.start();
79
80      // align back-end
81      BackEndAlignment cBackEndAligner;
82      cBackEndAligner.Inherit(&cTool);
83      cBackEndAligner.Start(0);
84      cBackEndAligner.waitForRunToBeCompleted();
85      // reset all chip and board registers
86      // to what they were before this tool was called
87      cBackEndAligner.Reset();
88
89      // if CIC is enabled then align CIC first
90      if(cWithCIC)
91      {
92          CicFEAlignment cCicAligner;
93          cCicAligner.Inherit(&cTool);
94          cCicAligner.Start(0);
95          cCicAligner.waitForRunToBeCompleted();
96          // reset all chip and board registers
97          // to what they were before this tool was called
98          cCicAligner.Reset();
99          cCicAligner.dumpConfigFiles();
100     }
```

Align Back-end board
OT module specific

Align CIC
OT module specific

Run calibration, save and clean

```
102     // now create a calibration object
103  1  CalibrationExample theCalibrationExample;
104  2  theCalibrationExample.Inherit (&cTool);
105  3  theCalibrationExample.Initialise ();
106  4  theCalibrationExample.runCalibrationExample();
107  5  theCalibrationExample.writeObjects();
108
109     //Tool old style command (some of them will vanish/merged)
110     cTool.dumpConfigFiles();
111     cTool.resetPointers();
112     cTool.SaveResults();
113     cTool.WriteRootFile();
114     cTool.CloseResultFile();
115     cTool.Destroy();
116     t.stop();
117     t.show ( "Time to Run Calibration example" );
118
119     if ( !batchMode ) cApp.Run();
120     return 0;
121  }
```

1. Create a CalibrationExample object
2. Pass to if the Tool (it will “take ownership” of the HW) Inherit comes from Tool (CalibrationExample inherit from Tool!!!), you do not need to implement it
3. Initialize calibration specific data members
4. run the calibration
5. conclude calibration

Save and Write results and clean up the memory (For the moment, just copy and paste, we are working on cleaning this part of the code)

Keep application running not to close the canvases

Fermilab

Calibration code

- tools/CalibrationExample.h
- tools/CalibrationExample.cc
- Implements the code of the calibration steps.

- We always keep the possibility to use Root directly in the calibration (used to debug network streaming utilities)
  - Handled by pre-compiler instructions:

```
#ifdef __USE_ROOT__
    // ROOT-depended instructions
#endif
```

Calibration Header - includes

calibration_example.cc | CalibrationExample.h

tools > CalibrationExample.h > ...

```
1   /*!
2   *
3   * \file CalibrationExample.h
4   * \brief Calibration example -> use it as a template
5   * \author Fabio Ravera
6   * \date 25 / 07 / 19
7   *
8   * \Support : fabio.ravera@cern.ch
9   *
10  */
11
12  #ifndef CalibrationExample_h__
13  #define CalibrationExample_h__
14
15  #include "Tool.h"
16  #include <map>
17  #ifdef __USE_ROOT__
18    //Calibration is not running on the SoC: I need to instantiate the DQM histrgrammer here
19    #include "../DQMUtils/DQMHistogramCalibrationExample.h"
20  #endif
21
22  using namespace Ph2_HwDescription;
23  using namespace Ph2_HwInterface;
24  using namespace Ph2_System;
25
```

Include headers:
DQMHistogramCalibrationExample
is included only if __USE_ROOT__ is used
(it will for today)

Specify namespaces will be used

Calibration Header - class declaration

CalibrationExample.h

tools > CalibrationExample.h > ...

```
25
26  class CalibrationExample : public Tool
27  {
28
29    public:
30      CalibrationExample();
31      ~CalibrationExample();
32
33      void Initialise           (void);
34      void runCalibrationExample(void);
35      void writeObjects         (void);
36
37      //State machine
38      void Start                (int currentRun) override;
39      void Stop                 (void) override;
40
41    private:
42      uint32_t fEventsPerPoint;
43
44      #ifdef __USE_ROOT__
45        //Calibration is not running on the SoC: Histogrammer is handeld by the calibration itself
46        DQMHistogramCalibrationExample fDQMHistogramCalibrationExample;
47      #endif
48  };
49
50  #endif
51
```

your calibration must inherit from Tool

declare function prototypes

State machine commands for running the calibration on the System on chip

data members
Histograms handled by an external class (loaded only if not running on the SoC)

Calibration Implementation - donstructor, destructor, initialize

CalibrationExample.h | CalibrationExample.cc

tools > CalibrationExample.cc > ...

```
1  #include "../tools/CalibrationExample.h"
2  #include "../Utils/Container.h"
3  #include "../Utils/ContainerFactory.h"
4  #include "../Utils/ContainerStream.h"
5  #include <math.h>
6
7  CalibrationExample::CalibrationExample() :
8      Tool(),
9      fEventsPerPoint(0)
10 {
11 }
12
13 CalibrationExample::~CalibrationExample()
14 {
15 }
16
17 void CalibrationExample::Initialise (void)
18 {
19
20     auto cSetting = fSettingsMap.find ( "Nevents" );
21     fEventsPerPoint = ( cSetting != std::end ( fSettingsMap ) ) ? cSetting->second : 10;
22
23     LOG (INFO) << "Parsed settings:" ;
24     LOG (INFO) << " Nevents = " << fEventsPerPoint ;
25
26     #ifdef __USE_ROOT__  // to disable and anable ROOT by command
27         //Calibration is not running on the SoC: plots are booked during initialization
28         fDQMHistogramCalibrationExample.book(fResultFile, *fDetectorContainer, fSettingsMap);
29     #endif
30
31 }
```

Include headers

Constructor

Destructor

Setting from the XML are converted into a map by the FileParser, retrieving Nevents

Book histograms

output file **the Detector** **settings**

Calibration Implementation - calibration code

```
33  void CalibrationExample::runCalibrationExample(void)
34  {
35      LOG (INFO) << "Taking Data with " << fEventsPerPoint << " triggers!" ;
36
37      DetectorDataContainer        theHitContainer;
38      ContainerFactory::copyAndInitChannel<uint32_t>(*fDetectorContainer, theHitContainer);
39
40      //getting n events and filling the container:
41      for(auto board : theHitContainer) //for on boards - begin
42      {
43          BeBoard* theBeBoard = fDetectorContainer->at(board->getIndex());
44          //Send N triggers (as it was in the past)
45          ReadNEvents ( theBeBoard, fEventsPerPoint );
46          //Get the event vector (as it was in the past)
47
48          const std::vector<Event*> &eventVector = GetEvents ();
49
50          for ( auto event : eventVector ) //for on events - begin
51          {
52              for(auto opticalGroup: *board) // for on opticalGroups - begin
53              {
54                  for(auto hybrid: *opticalGroup) // for on hybrid - begin
55                  {
56                      for(auto chip: *hybrid) // for on chip - begin
57                      {
```

Create a Container to store the hits

Allocate an uint32_t for each channel

Loop on all boards

This is a std::vector !!!

Retrieve corresponding board from the Detector and use it to send and read N triggers (ReadNEvents comes for free)

Retrieve the vector of events

Loop on all events

Loop on all optical groups

Loop on all hybrids

Loop on all chips

```
56                  for(auto chip: *hybrid) // for on chip - begin
57                  {
58                      unsigned int channelNumber = 0;
59                      for(auto &channel : *chip->getChannelContainer<uint32_t>()) // for on channel - begin
60                      {
61                          //retreive data in the old way and add to the current number of hits of the correspor
62                          channel += event->DataBit ( hybrid->getId(), chip->getId(), channelNumber++);
63                      } // for on channel - end
64                  } // for on chip - end
65              } // for on hybrid - end
66          } // for on opticalGroups - end
67      } // for on events - end
68  } // for on board - end      You, 3 hours ago • Updated OT exercise
69
70  #ifdef __USE_ROOT__
71      //Calibration is not running on the SoC: plotting directly the data, no shipping is done
72      fDQMHistogramCalibrationExample.fillCalibrationExamplePlots(theHitContainer);
73  #else
74      //Calibration is running on the SoC: shipping the data!!!
75      //I prepare a stream of an uint32_t container, prepareChannelContainerStreamer adds in the stream also th
76      // that is used when multiple calibrations are concatenated
77      auto theHitStream = prepareChannelContainerStreamer<uint32_t>();
78      // if the streamer was enabled (the supervisor script enable it) data are streamed
79      if(fStreamerEnabled)
80      {
81          // Disclamer: final MW will not do a for loop on board since each instance will hanlde 1 board only
82          for(auto board : theHitContainer)  theHitStream.streamAndSendBoard(board, fNetworkStreamer);
83      }
84  #endif
85  }
```

loops form previous slide

Loop on all channels

This is a std::vector<uint32_t> !!!

Event function that checks if channel from module and chip was hit

If not in the SoC, fill directly histograms

otherwise ship the through the network

Fermilab

Write out objects and SM commands

```
83
84  void CalibrationExample::writeObjects()
85  {
86      #ifdef __USE_ROOT__
87          //Calibration is not running on the SoC: processing the histograms
88          fDQMHistogramCalibrationExample.process();
89      #endif
90  }
91
92  //For system on chip compatibility
93  void CalibrationExample::Start(int currentRun)
94  {
95      LOG (INFO) << "Starting calibration example measurement.";
96      Initialise ( );
97      runCalibrationExample();
98      LOG (INFO) << "Done with calibration example.";
99  }
100
101 //For system on chip compatibility
102 void CalibrationExample::Stop(void)
103 {
104     LOG (INFO) << "Stopping calibration example measurement.";
105     writeObjects();
106     dumpConfigFiles();
107     SaveResults();
108     CloseResultFile();
109     Destroy();
110     LOG (INFO) << "Calibration example stopped.";
111 }
```

If not on the SoC, process histograms (plot them, SetLineColor, etc)

Call start and stop functions (as you call them in the script)
Not needed if not in the SoC

DQM histogram class - I

- DQMUtils/DQMHistogramCalibrationExample.h
- DQMUtils/DQMHistogramCalibrationExample.cc
- DQM class books the histograms:
  - DetectorDataContainer fDetectorHitHistograms;
    - Container for histograms (as before for storing data!!)
  - HistContainer<TH1F> theTH1FPedestalContainer("HitPerChannel", "Hit Per Channel", 254, -0.5, 253.5);
    - As you would create a TH1F in root, just wrapped into a class. It will be used as the histogram template to create all the needed histograms
  - RootContainerFactory::bookChipHistograms<HistContainer<TH1F>>(theOutputFile, theDetectorStructure, fDetectorHitHistograms, theTH1FPedestalContainer);
    - As ContainerFactory, but you have to pass to it the output file and the histogram template
    - theDetectorStructure = this is the detector structure as fDetectorContainer
    - fDetectorHitHistograms = container of the histograms you are allocating
    - as for ContainerFactory, you can allocate histograms for Channel, Chip, Hybrid, OpticalGroup, Board and Detector
    - Histograms are automatically saved and destroyed using this tool!!

DQM histogram class - II

- Plot filling has to be done by a function that receives
  - ONE AND ONLY ONE container
    - the shipping though the network you can ship one at a time
  - In case need, other types contiguous in memory (easier to be shipped through network)
    - uint23_t, array, etc are fine
    - vectors, string, maps, pointers are not!!
    - example: need to scan over the threshold and measure occupancy
      - fillThresholdOccupancy(uint16_t threshold, DetectorDataContainer theOccupancy)

- When infos are sent through the network, fill class handles the data decoding and calls the previous function -> write well the previous one!!!

DQM Header - includes

CalibrationExample.cc | DQMHistogramCalibrationExample.h

DQMUtils > DQMHistogramCalibrationExample.h > ...

```
1   /*!
2           \file                DQMHistogramCalibrationExample.h
3           \brief               DQM class for Calibration example -> use it as a templare
4           \author              Fabio Ravera
5           \date                25/7/19
6           Support :            mail to : fabio.ravera@cern.ch
7   */
8
9   #ifndef __DQMHISTOGRAMCALIBRATIONEXAMPLE_H__
10  #define __DQMHISTOGRAMCALIBRATIONEXAMPLE_H__
11  #include "../DQMUtils/DQMHistogramBase.h"
12  #include "../Utils/Container.h"
13  #include "../Utils/DataContainer.h"
14
15  class TFile;
16
17  /*!
18   * \class DQMHistogramCalibrationExample
19   * \brief Class for CalibrationExample monitoring histograms
20   */
21  class DQMHistogramCalibrationExample : public DQMHistogramBase
22  {
23
24    public:
25      /*!
26       * constructor
27       */
28      DQMHistogramCalibrationExample ();
29
30      /*!
31       * destructor
32       */
33      ~DQMHistogramCalibrationExample();
34
```

Include headers / declare classes

Class must inherit from DQMHistogramBase

DQM Header - member function

```
35      /*!
36       * \brief Book histograms
37       * \param theOutputFile : where histograms will be saved
38       * \param theDetectorStructure : Detector container as obtained after file parsing, used to create histograms for all board/chip/modu
39       * \param pSettingsMap : setting as for Tool setting map in case coe informations are needed (i.e. FitSCurve)
40       */
41      void book(TFile *theOutputFile, const DetectorContainer &theDetectorStructure, const Ph2_System::SettingsMap& pSettingsMap) override;
42
43      /*!
44       * \brief fill : fill histograms from TCP stream, need to be overwritten to avoid compilation errors, but it is not needed if you do
45       * \param dataBuffer : vector of char with the TCP datastream
46       */
47      bool fill (std::vector<char>& dataBuffer) override;
48
49      /*!
50       * \brief process : do something with the histogram like colors, fit, drawing canvases, etc
51       */
52      void process () override;
53
54      /*!
55       * \brief Reset histogram
56       */
57      void reset(void) override;
58
59      /*!
60       * \brief fillCalibrationExamplePlots
61       * \param theHitContainer : Container with the hits you want to plot
62       */
63      void fillCalibrationExamplePlots(DetectorDataContainer &theHitContainer);
64
65  private:
66
67      DetectorDataContainer fDetectorHitHistograms;
68      DetectorDataContainer fDetectorData;
69  };
70  #endif
```

Functions you are forced to implement otherwise it will not compile

Function receiving container and filling histograms

Container for histograms

Container for decoding data received from the network

DQM Implementation - includes, constructor, destructor

CalibrationExample.cc | DQMHistogramCalibrationExample.cc

DQMUtils > DQMHistogramCalibrationExample.cc > ...

```
1   /*!
2           \file               DQMHistogramCalibrationExample.cc
3           \brief              DQM class for Calibration example -> use it as a templare
4           \author             Fabio Ravera
5           \date               25/7/19
6           Support :           mail to : fabio.ravera@cern.ch
7   */
8
9   #include "../DQMUtils/DQMHistogramCalibrationExample.h"
10  #include "../Utils/Container.h"
11  #include "../Utils/ContainerFactory.h"
12  #include "../RootUtils/RootContainerFactory.h"
13  #include "../Utils/ContainerStream.h"
14  #include "../RootUtils/HistContainer.h"
15  #include "TCanvas.h"
16  #include "TFile.h"
17
18  //========================================================================================================================
19  DQMHistogramCalibrationExample::DQMHistogramCalibrationExample ()
20  {
21  }
22
23  //========================================================================================================================
24  DQMHistogramCalibrationExample::~DQMHistogramCalibrationExample ()
25  {
26
27  }
28
```

Include headers / declare classes

Constructor

Destructor

Fermilab

DQM Implementation - Book histograms

```
28
29  //========================================================================================================================
30  void DQMHistogramCalibrationExample::book(TFile *theOutputFile, const DetectorContainer &theDetectorStructure, std::map<std::string,
31  {
32      // SoC utilities only - BEGIN
33      // THIS PART IT IS JUST TO SHOW HOW DATA ARE DECODED FROM THE TCP STREAM WHEN WE WILL GO ON THE SOC
34      // IF YOU DO NOT WANT TO GO INTO THE SOC WITH YOUR CALIBRATION YOU DO NOT NEED THE FOLLOWING COMMENTED LINES
35      // make fDetectorData ready to receive the information fromm the stream
36      ContainerFactory::copyStructure(theDetectorStructure, fDetectorData);
37      // SoC utilities only - END
38
39      // creating the histograms fo all the chips:
40      // create the HistContainer<TH1F> as you would create a TH1F (it implements some feature needed to avoid memory leaks in copying
41      HistContainer<TH1F> theTH1FPedestalContainer("HitPerChannel", "Hit Per Channel", 254, -0.5, 253.5);
42      // create Histograms for all the chips, they will be automatically accosiated to the output file, no need to save them, change t
43      RootContainerFactory::bookChipHistograms<HistContainer<TH1F>>(theOutputFile, theDetectorStructure,
44          fDetectorHitHistograms, theTH1FPedestalContainer);
45  }
46
```

Only copy the tree structure of the detector, used for decoding network streams

- Reminder how the function is called in the Calibration code:

```
25
26  #ifdef __USE_ROOT__  // to disable and anable ROOT by command
27      //Calibration is not running on the SoC: plots are booked during initialization
28      fDQMHistogramCalibrationExample.book(fResultFile, *fDetectorContainer, fSettingsMap);
29  #endif
30
```

output file **the Detector** **settings**

Book histograms

DQM Implementation - Fill histograms

CalibrationExample.cc | DQMHistogramCalibrationExample.cc

DQMUtils > DQMHistogramCalibrationExample.cc > ...

```
46
47  //========================================================================================================================
48  void DQMHistogramCalibrationExample::fillCalibrationExamplePlots(DetectorDataContainer &theHitContainer)
49  {
50      for(auto board : theHitContainer) //for on boards - begin
51      {
52          size_t boardIndex = board->getIndex();
53          for(auto module: *board) //for on module - begin
54          {
55              size_t moduleIndex = module->getIndex();
56              for(auto chip: *module) //for on chip - begin
57              {
58                  size_t chipIndex = chip->getIndex();
59                  // Retreive the corresponging chip histogram:
60                  TH1F *chipHitHistogram = fDetectorHitHistograms.at(boardIndex)->at(moduleIndex)->at(chipIndex)
61                      ->getSummary<HistContainer<TH1F>>().fTheHistogram;
62                  uint channelBin=1;
63                  // Check if the chip data are there (it is needed in the case of the SoC when data may be sent chip by chip and not
64                  if(chip->getChannelContainer<uint32_t>() == nullptr ) continue;
65                  // Get channel data and fill the histogram
66                  for(auto channel : *chip->getChannelContainer<uint32_t>()) //for on channel - begin
67                  {
68                      chipHitHistogram->SetBinContent(channelBin++,channel);
69                  } //for on channel - end
70              } //for on chip - end
71          } //for on module - end
72      } //for on boards - end
73  }
74
```

Loop on all boards

Loop on all modules

Loop on all chips

Retrieve the corresponding chip histogram

Only for the SoC, data may be partial

Loop on all channels

Fill the plot

DQM Implementation - Plot histograms

CalibrationExample.cc | DQMHistogramCalibrationExample.cc

DQMUtils > DQMHistogramCalibrationExample.cc > ...

```
75  //=============================================================================================================
76  void DQMHistogramCalibrationExample::process()
77  {
78      // This step it is not necessary, unless you want to format / draw histograms,
79      // otherwise they will be automatically saved
80      for(auto board : fDetectorHitHistograms) //for on boards - begin
81      {
82          size_t boardIndex = board->getIndex();
83          for(auto module: *board) //for on module - begin
84          {
85              size_t moduleIndex = module->getIndex();
86
87              //Create a canvas do draw the plots
88              TCanvas *cValidation = new TCanvas(("Hits_module_" + std::to_string(module->getId())).data(),("Hits module " + std::to_s
89              cValidation->Divide(module->size());
90
91              for(auto chip: *module)  //for on chip - begin
92              {
93                  size_t chipIndex = chip->getIndex();
94                  cValidation->cd(chipIndex+1);
95                  // Retreive the corresponging chip histogram:
96                  TH1F *chipHitHistogram = fDetectorHitHistograms.at(boardIndex)->at(moduleIndex)->at(chipIndex)
97                      ->getSummary<HistContainer<TH1F>>().fTheHistogram;
98
99                  //Format the histogram (here you are outside from the SoC so you can use all the ROOT functions you need)
100                 chipHitHistogram->SetStats(false);
101                 chipHitHistogram->SetLineColor(kRed);
102                 chipHitHistogram->DrawCopy();
103             } //for on chip - end
104         } //for on module - end
105     } //for on boards - end
106 }
107
```

Loop on all boards

Loop on all modules

Create a canvas

Loop on all chips

Retrieve the corresponding chip histogram

Set histogram properties and plot

DQM Implementation - Fill from network

CalibrationExample.cc | DQMHistogramCalibrationExample.cc

DQMUtils > DQMHistogramCalibrationExample.cc > ...

```
107
108  //=============================================================================================================
109  void DQMHistogramCalibrationExample::reset(void)
110  {
111      // Clear histograms if needed
112  }
113
114  //=============================================================================================================
115  bool DQMHistogramCalibrationExample::fill(std::vector<char>& dataBuffer)
116  {
117      // SoC utilities only - BEGIN
118      // THIS PART IT IS JUST TO SHOW HOW DATA ARE DECODED FROM THE TCP STREAM WHEN WE WILL GO ON THE SOC
119      // IF YOU DO NOT WANT TO GO INTO THE SOC WITH YOUR CALIBRATION YOU DO NOT NEED THE FOLLOWING COMMENTED LINES
120
121      //I'm expecting to receive a data stream from an uint32_t contained from calibration "CalibrationExample"
122      ChannelContainerStream<uint32_t>  theHitStreamer("CalibrationExample");
123
124      // Try to see if the char buffer matched what I'm expection (container of uint32_t from CalibrationExample procedure)
125      if(theHitStreamer.attachBuffer(&dataBuffer))
126      {
127          //It matched! Decoding chip data
128          theHitStreamer.decodeChipData(fDetectorData);
129          //Filling the histograms
130          fillCalibrationExamplePlots(fDetectorData);
131          //Cleaning the data container to be ready for the next TCP string
132          fDetectorData.cleanDataStored();
133          return true;
134      }
135      // the stream does not match, the expected (DQM interface will try to check if other DQM istogrammers are looking for this strea
136      return false;
137      // SoC utilities only - END
138  }
139
```

Clean up histograms here

Fill from the network, not needed today! (just put a return false)

Waiting to receive a stream with uint32_t for each channel from a calibration called CalibrationExample

If stream corresponds to the expected data format

Decode data and make the fDetectorData container pointing to them

Same function to fill the histograms!! -> write it well

Data where read, cleaning up the container

It was my data stream!

It was someone else data stream!

What about RD53?

- Switch to branch daqWorkshopIT

- src/RD53calibration_example.cc
- tools/RD53CalibrationExample.h
- tools/RD53CalibrationExample.cc
- DQMUtils/RD53DQMHistogramCalibrationExample.h
- DQMUtils/RD53DQMHistogramCalibrationExample.cc

- (Tons of fantasy)
- We do exactly the same measurement for RD53
- Small changes are needed

- How to run the example
  - RD53calibration_example.cc -f <xml file for your setup>

Differences for the RD53 code - RD52CalibrationExample

```
20      auto cSetting = fSettingsMap.find ( "nEvents" );
21      fEventsPerPoint = ( cSetting != std::end ( fSettingsMap ) ) ? cSetting->second : 10;
22      fRowStart     = fSettingsMap["ROWstart"];
23      fRowStop      = fSettingsMap["ROWstop"];
24      fColStart     = fSettingsMap["COLstart"];
25      fColStop      = fSettingsMap["COLstop"];
26
27      LOG (INFO) << "Parsed settings:" ;
28      LOG (INFO) << " nEvents = " << fEventsPerPoint ;
29
```

Get some more informations from the XML file

```
47          for(auto event: eventVector) // for on events - begin
48          {
49              for(auto opticalGroup: *board) // for on opticalGroup - begin
50              {
51                  for(auto hybrid: *opticalGroup) // for on hybrid - begin
52                  {
53                      for(auto chip: *hybrid) // for on chip - begin
54                      {
55                        auto RD53_event = static_cast<RD53Event*>(event);
56                        if(RD53_event->isHittedChip(hybrid->getId(), chip->getId(), chipIndx) == true)
57                          {
58                              for(const auto& hit: RD53_event->chip_frames_events[chipIndx].second.hit_data)
59                              {
60                                  chip->getChannel<uint32_t>(hit.row, hit.col)++;
61                              }       You, seconds ago • Uncommitted changes
62                          }
63                      } // for on chip - end
64                  }     // for on hybrid- end
65              }         // for on opticalGroup - end
66          }             // for on events - end
```

Check if chip has hits

Loop over all hits and increase the channel count

Differences for the RD53 code - RD53DQMHistogramCalibrationExample

```
35      fRowStart = this->findValueInSettings(pSettingsMap, "ROWstart");
36      fRowStop  = this->findValueInSettings(pSettingsMap, "ROWstop");
37      fColStart = this->findValueInSettings(pSettingsMap, "COLstart");
38      fColStop  = this->findValueInSettings(pSettingsMap, "COLstop");
39
40      // creating the histograms fo all the chips:
41      // create the HistContainer<TH2F> as you would create a TH2F (it implements some feature needed to avoid memor
42      HistContainer<TH2F> theTH2FPedestalContainer("HitPerChannel", "Hit Per Channel",
43          fColStop - fColStart, fColStart, fColStop, fRowStop - fRowStart, fRowStart, fRowStop);
44      // create Histograms for all the chips, they will be automatically accosiated to the output file, no need to s
45      RootContainerFactory::bookChipHistograms<HistContainer<TH2F>>(theOutputFile,
46          theDetectorStructure, fDetectorHitHistograms, theTH2FPedestalContainer);      You, seconds ago • Uncommitt
```

Create a TH2F instead of a TH1F

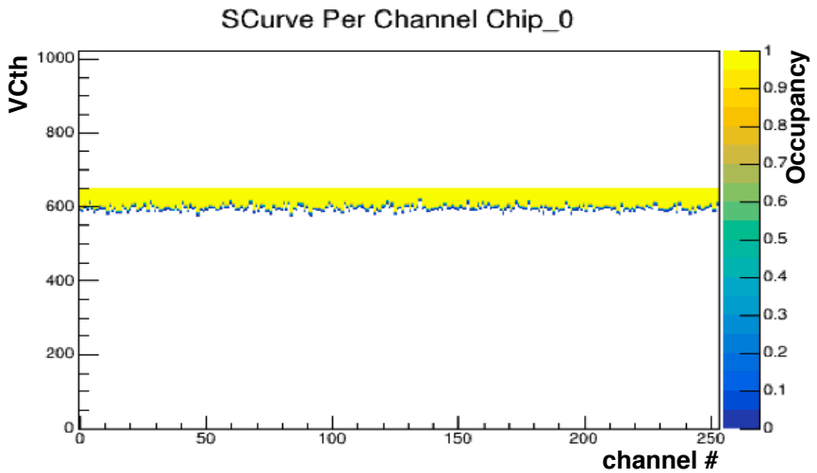
```
63                  TH2F* chipHitHistogram = fDetectorHitHistograms.at(boardIndex)->at(opticalGroupIndex)->at(hybridIndex)
64                      ->at(chipIndex)->getSummary<HistContainer<TH2F>>().fTheHistogram;      You, seconds ago • Uncommitted
65                  // Check if the chip data are there (it is needed in the case of the SoC when data may be sent chip by chi
66                  if(chip->getChannelContainer<uint32_t>() == nullptr) continue;
67
68                  // Get channel data and fill the histogram
69                  for(auto row = fRowStart; row <= fRowStop; row++)
70                  {
71                      for(auto col = fColStart; col <= fColStop; col++)
72                      {
73                          chipHitHistogram->Fill(col, row, chip->getChannel<uint32_t>(row, col));
74                      }
75                  }
```

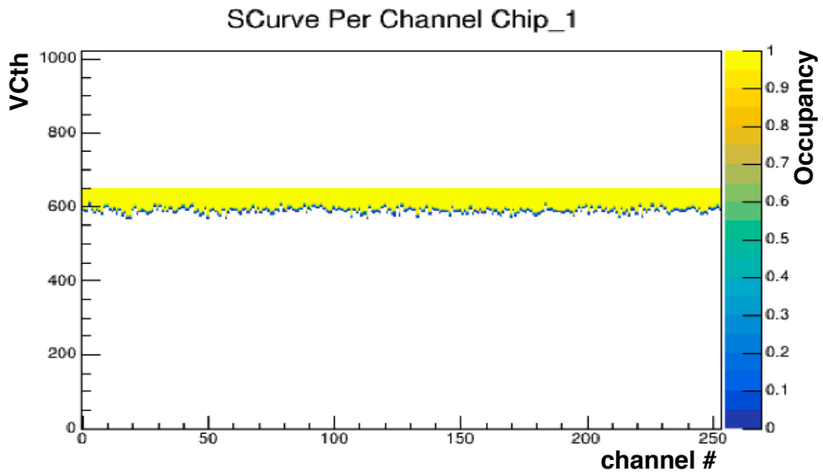
retrieve number of hits from the container and fill the histogram

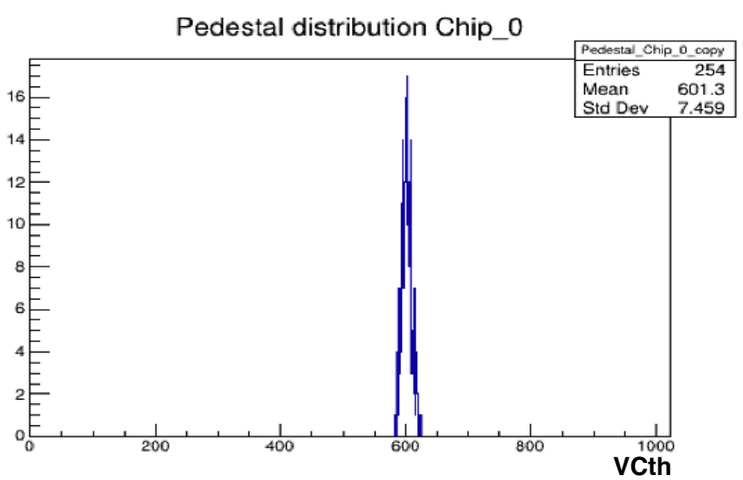
Fermilab

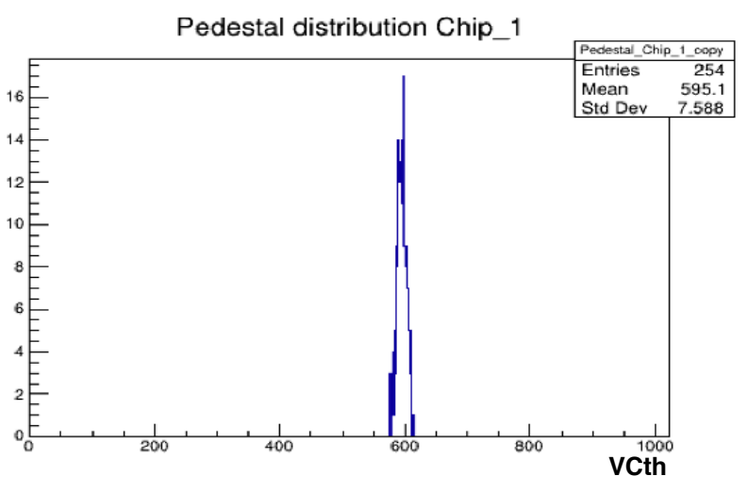
Your turn!! - OT

- Outer tracker group:
  - measure occupancy for threshold going from 550 to 650 (S-curve)
    - Tip: Change threshold using
      - fReadoutChipInterface->WriteChipReg (ReadoutChip*, “VCth”, dacValue);
  - plot a 2D histogram (x = channel #, y = threshold, z = occupancy)
  - estimate pedestal using root (use a simple assumption: bin closer to 50%)
    - If you finish in time, try to do the same pedestal estimation without root









OT skeleton

- Switch to daqWorkshop branch
- script:
  - src/run_simple_scurve.cc
- calibration class:
  - tools/SimpleSCurve.h
  - tools/SimpleSCurve.cc
- DQM class:
  - DQMUtils/DQMHistogramSimpleSCurve.h
  - DQMUtils/DQMHistogramSimpleSCurve.cc

Your turn!! - IT

- Same Idea, but stuff are a bit more complicated here:
  - Buffers are limited, you cannot simply lower the threshold to look for a pedestal
    - You will inject and change the threshold to measure the Vthr at which you see an hit
  - Injection has to be done in patterns of pixels
- We provide the tools for doing that!
  - In Initialise function you must enable the utility for handling pattern injection:

```
36
37      // ###################################
38      // # Custom channel group handler #
39      // ###################################
40      ChannelGroup<RD53::nRows,RD53::nCols> customChannelGroup;
41      customChannelGroup.disableAllChannels();
42
43      for (auto row = fRowStart; row <= fRowStop; row++)
44          for (auto col = fColStart; col <= fColStop; col++)
45          customChannelGroup.enableChannel(row,col);
46
47      fChannelGroupHandler = new RD53ChannelGroupHandler(customChannelGroup, RD53GroupType::AllGroups);
48      fChannelGroupHandler->setCustomChannelGroup(customChannelGroup);
49
```

For time reason you are going to inject on a chance subset:
Create a list of enable channels

Create the Channel handler (fChannelGroupHandler is a Tool data member) and pass to it the list of enabled channels

Your turn!! - IT

- In this case you will need to create a container in which all layers have a OccupancyAndPh object and pass it to Tool:

```
62      // Create a container to store the occupancy and allocate a float for every channel
63      DetectorDataContainer  theOccupancyContainer;
64      ContainerFactory::copyAndInitStructure<OccupancyAndPh>(*fDetectorContainer, theOccupancyContainer);
65      fDetectorDataContainer = &theOccupancyContainer;
66
```

fDetectorDataContainer is a Tool data member

- Enable injection, enable masking and take data:

```
83
84      SetTestPulse(true);
85      fMaskChannelsFromOtherGroups = true;
86      measureData(fEventsPerPoint);
87
```

- measureData is a Tool function that will scan all the enabled pixels with a predefined pattern of injection and will update the fDetectorDataContainer which is pointing to theOccupancyContainer (it works for 2S too… do not cheat!)
- To get the occupancy from a OccupancyAndPh object do foo.fOccupancy

IT skeleton

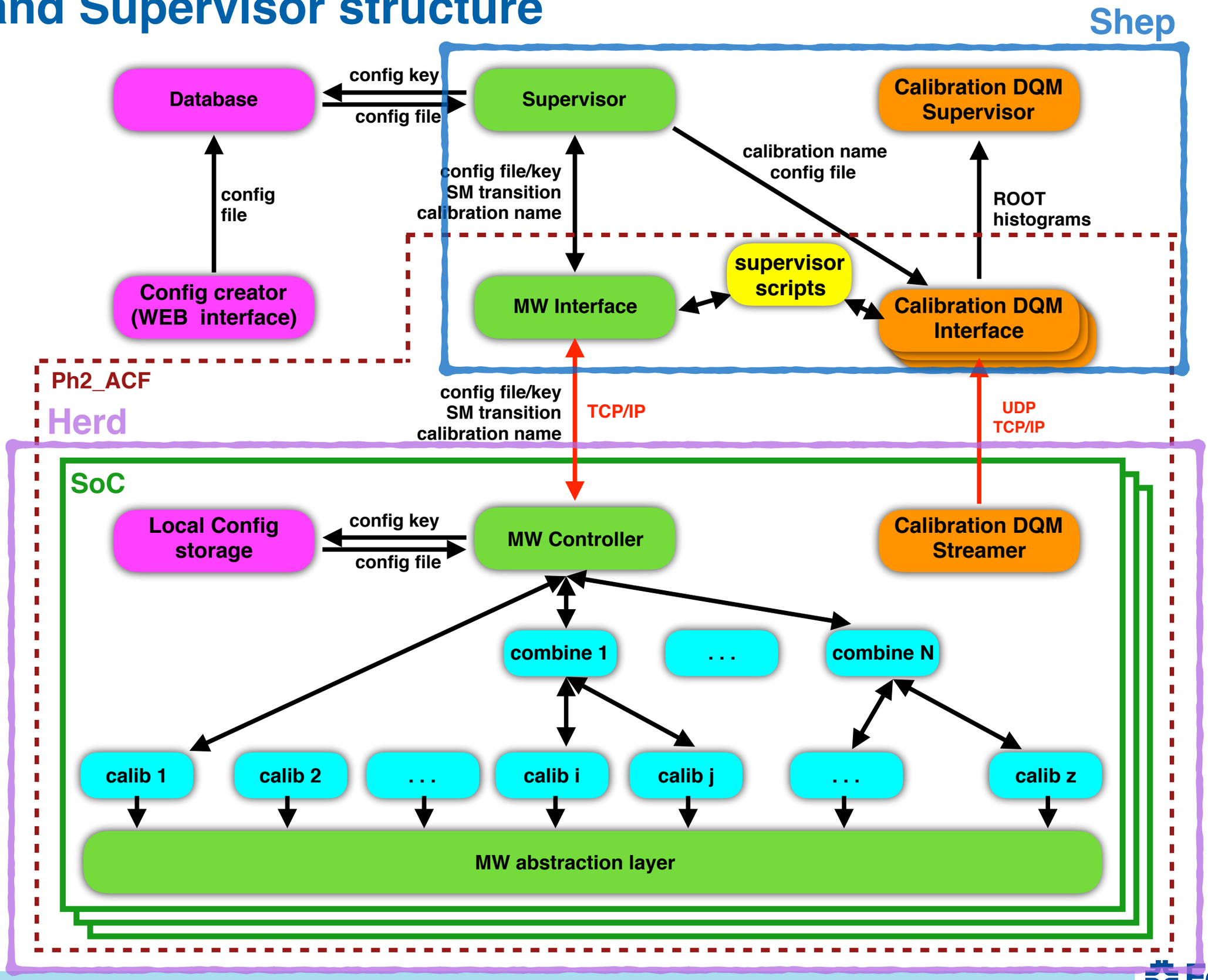
- Switch to daqWorkshopIT branch
- script:
  - src/run_RD53simple_scurve.cc
- calibration class:
  - tools/RD53SimpleSCurve.h
  - tools/RD53SimpleSCurve.cc
- DQM class:
  - DQMUtils/RD53DQMHistogramSimpleSCurve.h
  - DQMUtils/RD53DQMHistogramSimpleSCurve.cc

GOOD LUCK!

Casting in C++ and usage in the Ph2_ACF

- Casting in C++ allows you to change the variable time
- Casting we are using in the code are needed to move up in the inheritance layers, and two are mainly used:
  - static_cast<ChildClass*>(theVariable):
    - ChildClass must inherit from the theVariable type
    - cast done at compiler time, but not checks done (if variableName was not pointing to a ChildClass object it will cause undefined behavior)
    - Preferred way since it does not cause a run time overhead, but be careful
  - static_cast<ChildClass*>(theVariable):
    - ChildClass must inherit from the theVariable type
    - cast done at run time, it checks if theVariable is from ChildClass and if not returns a null pointer
    - Safer but it has run time overhead, use it only if you do not know a priori was object it is pointed by theVariable

MW and Supervisor structure



Fermilab