Apollo User Manual

S. Di Dimizio [†], A. Giachero [‡], E. Guardincerri [§], M. Pallavicini [†]

January 15, 2007

[†]affiliation 2
[‡]affiliation 3
[§]affiliation 4

Contents

Introduction

To be done

Chapter 1

Running Apollo

1.1 Intoduction

To be done

1.2 Running Mode

To be done

1.2.1 Data taking modes of Diana within Apollo

Diana is used to perform several online operations with different set of modules and configuration files. The following is a preliminary list of "modes" with their function and use.

1.2.1.1 Mode 0 - Normal

All trigger used defined; heater may be on of or off according to user needs; online monitor available; raw data output enabled; single channel DAQ according to user selection. This is the normal physics running mode.

1.2.1.2 Mode 1 - LoadCurve_1

Random trigger `WYQApolloRandomTrigger`; no heater; compute channel voltage baseline after stop using Diana module `MDaqComputeBaseline`; data saved in a temporary files.

1.2.1.3 Mode 2 - LoadCurve_2

Heater trigger `QApolloHeaterTrigger`; heater on; compute peak-base amplitude spectrum for each channel in module `MDaqHeaterIVAnalysis`; data saved in a temporary files.

1.2.1.4 Mode 3 - LoadCurve_3

Heater trigger `QApolloHeaterTrigger`; heater on; compute peak-base amplitude spectrum for each channel using Optimum Filter. The very same modules that are used in data analysis will be used to compute the average pulse and to compute the optimum filter. The final module `MDaqFinalIVAnalysis` will compute the best working point for each channel and save into database.

Chapter 2

Installation of Apollo

2.1 Introduction

To be done

Chapter 3

Processes running in Apollo

3.1 Introduction

The following processes are needed by Apollo: DataReader, DataSender, DataReceiver, Builder, DaqServer, SlowServer, MsgServer and Gui. In case all proceesses run on the same machine (like for example in Hall C) the DataSender and DataReceiver processes do not exist.

3.2 Processes definition

The follwing is a brief description of these processes and their functions:

- **DataReader**: it is the process that continuosly reads data from all channels defined, and apply triggering and possibly filtering algorythms to the data. At this level the trigger algotythms just flag the events, without creating the pulses. All data are then stored into a shared memory segment (one for each channel). There is one DataReader process for each DAQ crate. In Hall C (and possibly also in Hall A in Cuoricino), there is only one DAQ crate. In Cuore the number will be of the order of 10 or less.

- **DataSender**: a processes that simply take date from the DataReader shared memory and send it to the Builder machine via network (TCP/IP connection). A DataReceiver is there available to collect this data and fill a shared memory segment that is identycal to the one filled by DataReader. In this way the couple DataSender - DataReceiver can be just removed when everything is installed in the same machine.

Chapter 4

Data Base for Apollo

4.1 Introduction

To be done

Chapter 5

Graphical User Interface

5.1 Introduction

This section describes the main features of the Apollo Graphical User Interface (GUI).

5.2 List of GUI windows

The Apollo GUI is made of the following windows:

- **Main Apollo Window:** This is the main entrance window of the whole GUI. All other windows can be open from here. This window handles directly the following functions:
  - **Login:** User authentication. Without login no action is allowed. Users that have a valid login password can be:
    - GUEST: only viewers of the status of the system. A GUEST can look at histograms or data but cannot change the status of the system in any way
    - OPERATOR: can start or stop run, and not much else
    - EXPERT: can perform settings, change hardware parameters, create new profiles and such
    - ADMINISTRATOR: can do everything

    See **??** section for more details.
  - **System Setup:** Setting of all Apollo Environment variables (i.e. Data Base location, computing system variables and similar). See section **??** for more details.

- **Profile Manager Window:** This window allow to create, modify and view a data base profile. A new profile can be created starting from the existing status of the system or copying from an already existing profile. Only EXPERT or ADMINISTRATORS can use this window.

- **Slow Control Window:** This window allow to set/view hardware parameters for Front End, Bessel, Pulser and DAQ boards parameters. This window cannot be used during data taking, unless a special test run is in progress. Only EXPERT or ADMINISTRATORS can use this window. A very low level interface is also available to address physical registers (by-passing logical channel address system). This very low level interface can be used by ADMINISTRATORS only. See **??** for details.

- **Run Controller Window:** Handles start and stop of a given run, allowing trigger setup, profile selection, and options concerning data file saving, format and so on.

- **Run analyzer:** This is the main window of the data analyzer system. From this window a set of different windows can be open to view data during data taking for data quality assurance. The list of sub-windows include:

  - **Single channel analyzer:** Displays infos and histograms of data of a single channel. Histograms are update real-time during data flow. There are no limits on the number of histograms available, although up to 4 or 8 (depending on monitor size) can be displayed at the same time. The list of histograms include:

    1. Baseline at full resolution (last 5 seconds)
    2. Medium resolution baseline (last 10 minutes)
    3. Low resolution baseline (from the beginning of run)
    4. Sigma baseline after pulse removal (3 histograms as for baseline)
    5. Triggering pulses in real time (all pulses)
    6. High energy triggering pulses (same as above with high energy threshold)
    7. Channel energy spectrum (simple energy calculation as peak-base)
    8. Channel energy spectrum with pulser triggers only
    9. Channel energy spectrum with particle triggers only (everything but pulser)

10. FFT spectrum of noise after pulse removal
11. FFT spectrum including pulses

– **Global analyzer:** Displays infos and histograms of data of the whole collection of channels.

Chapter 6

Trigger

6.1 Introduction

We define a *trigger* an algorythm designed to search, identify and tag any interesting feature (e.g. a pulse) in the data flow of a single DAQ channel. Apollo supports the existence of several trigger algorythms running at the same time. The number of algorythms is in principle unlimited, although in the current implementation no more than 4 can be run at the same time in a given run. The trigger parameters can be tuned channel by channel.

The trigger algorythms are executed in DataReader .

6.2 Setup

The table *trigger* in *qdaq* database contains the trigger setup informations for each run. There are 14 fields:

1. **run:** the run number this record refers to. There must be a record for each valid run.

2. **lg:** the logical channel number this record refers to. A record with lg=0 is mandatory for each run. This record will act as default trigger setup for all channels that are NOT explicitely listed in the following records.If lg is different from 0, the trigger setup of the corresponing channel is overwritten. In this way we can have all channels with the same trigger parameters by writing the record with lg=0 only, or we can tune channel by channel adding more records.

3. **trg1_enable** Enable flag for trigger 1 for this channel (or the default value for lg=0)

4. **trg2_enable** Enable flag for trigger 2 for this channel (or the default value for lg=0)

5. **trg3_enable** Enable flag for trigger 3 for this channel (or the default value for lg=0)

6. **trg4_enable** Enable flag for trigger 4 for this channel (or the default value for lg=0)

7. **trg1_name** The name of the trigger 1 active in this run.

8. **trg2_name** The name of the trigger 2 active in this run.

9. **trg3_name** The name of the trigger 3 active in this run.

10. **trg4_name** The name of the trigger 4 active in this run.

11. **trg1_pars**A string describing the parameters for trigger 1 algorythm for this channel, and this run. The syntax of this string must be key1=val1;key2=val2 etc. The number of parameters is unlimited.

12. **trg2_pars**A string describing the parameters for trigger 2 algorythm for this channel, and this run. The syntax of this string must be key1=val1;key2=val2 etc. The number of parameters is unlimited.

13. **trg3_pars** A string describing the parameters for trigger 3 algorythm for this channel, and this run. The syntax of this string must be key1=val1;key2=val2 etc. The number of parameters is unlimited.

14. **trg4_pars** A string describing the parameters for trigger 4 algorythm for this channel, and this run. The syntax of this string must be key1=val1;key2=val2 etc. The number of parameters is unlimited.

Chapter 7

Load Curves

7.1 Introduction

7.2 Description of the procedure

7.2.1 Step 1: Gain and offset

1. Channel gain set to a reasonable low value;

2. All bias set to zero nominal;

3. Start data taking Mode 1;

4. Increment bias with a rough step, go back to step 3 until end of bias scan;

5. Increment gain until maximum gain compatible with the whole I-V curve is found. All temporary files produced at step 3 are analyzed by module `MDaqFindBestGain` and save in database.

At the end of this step for each channel define in profile a proper gain and offset value is saved in the database.

7.2.2 Step 2: I-V curve

Similar to Step 1 with fixed gain and much finer bias step. At the end of this step the whole I-V curve is saved by module `MDaqBuildIVCurve` in database.

7.2.3 Step 3: Analisys of I-V curve

Analysis of I-V curve and calculation of inversion point (if possible), work point 1 (WP1), which is the point from which the system will scan using heater pulses; around WP1 an interval IWP1 is also given (which is the interval around WP1 that will be scan). This operation is done by `MDaqIVCurveAnalyzer`. At the end all I-V curves saved into database.

7.2.4 Step 4: Heater scan around WP1

1. Channel gain ad offset set to value deteminated in step 1
2. All bias set to the beginning of IWP1 interval.
3. Start data taking Mode 2;
4. Increment bias with proper step within interval IWP1;

7.2.5 Step 5: Analisys of heater curve 1

Analisys of temporary files produced on step 4; determine WP2 (point of maximum amplitude) and save into temporary files.

7.2.6 Step 6: Measure average heater pulse and noise in heater curve 1

Scan as Step 4 but using run Mode 3.

7.2.7 Step 7: Getting WP using optimum filter

Analisys of temporary files produced on step 6; determine WP (best working point) and save into database.