Table of Contents

TB2009DataAnalysis

Introduction

Muon Atlas MicroMegas Analysis (**MAMMA**) is an analysis framework that is being developed with a goal to provide a coherent framework to analyze the test beam data collected for the ATLAS Muon upgrade project (more information here: MuonMicromegas). There are several steps involved currently to convert raw data collected using the Bonn Atlas Telescope (BAT) and Micromegas (MM) data acquisition systems into rootuples which can be analyzed using ROOT. The scope of the analysis framework includes easier conversion of raw data.

People Involved in Analysis/Software

- List of people involved and willing to contribute to the analysis (please add yourself if you are interested)

- W. Park (U. South Carolina)
- K. Nikolopolous (Athens/BNL)
- V. Kaushik (U. Arizona)

Communication

For the analysis activities we have a hypernews that one can subscribe to. All discussions (and nuts-and-bolts of analysis) should be addressed in this forum
hn-atlas-muon-micromegas-testbeam-analysis@cernSPAMNOT.ch

Data Preparation

This stage involves converting BAT data and the raw MM data into rootuples. The run information from the AtlasMMLogbook is read and stored for use in analysis by the `RunInfo` class. All the relevant information is contained in this class. In addition, the temperature, relative humidity and the pressure information (only for MM) is also read from the input and stored in the ntuples. The BAT data consists of strip information for the four modules (labeled 1,2,3,6) along with the charge on each strip, the strip ID and the run and event numbers. Every BAT run number has a corresponding (and different) MM run number which is stored in the `RunInfo` object. The MM raw data also consists of charge read out over several time-samples for each strip, the run and event numbers along with the auxiliary information about the gas mixture used, the strip pitch and width and inclination of the chamber with respect to normal and a comments field. There are several issues with the raw data itself, which needs to be addressed before analysis can be performed which is also the scope of data-preparation stage. These are listed below:

- BAT internal alignment
- Mapping
- Synchronization of BAT events with MM events

Data Analysis Tasks

- Convert all available BAT and MM runs into analysis ntuples
  - ♦ Strip data (charge measurement for time samples) for BAT / MM
  - ♦ Auxiliary data (event information, pressure, humidity, gas mixture etc.)
- Alignment Studies
  - ♦ Internal BAT Alignment
  - ♦ External Alignment
- Mapping
- Synchronization of BAT/MM events
- Data Quality Monitoring
- Reconstruction
  - ♦ Fitting time samples and extracting amplitude/time for each strip.
  - ♦ Clusterization - grouping strips into a cluster. Charge and position measurement
  - ♦ Extrapolation of tracks from BAT to MM
  - ♦ Matching hits/ finding track segments in MM
  - ♦ Determine position resolution
- μ-TPC Studies

Software Tasks

- Documentation
  - ♦ Doxygen documentation of code
  - ♦ Description of available information in the ntuples
- Data-taking/Monitoring
  - ♦ Reading configuration from GUI
  - ♦ improving the look/feel of existing monitoring GUI

Core Analysis Package

The MAMMA package is available for download at this SVN link.

Checking out, installing and running the package

The requirements are as follows:

- You need an AFS user account and be able to access to one of the lxplus@cernSPAMNOT.ch nodes
- Python (v 2.5 or higher)
- ROOT (v 5.20 or higher)
    - To setup ROOT environment one can do the following in bash shell: (or use equivalent setenv commands in c-shell):

```
export
ROOTSYS=/afs/cern.ch/sw/lcg/app/releases/ROOT/5.22.00b/slc4_ia32_gcc
export PATH=$ROOTSYS/bin/:$PATH
export LD_LIBRARY_PATH=${ROOTSYS}/lib:$LD_LIBRARY_PATH
export PYTHONPATH=${ROOTSYS}/lib:$PYTHONPATH
```

Once you log into the lxplus node with your username and password you can request for a scratch area (usually 100 MB is minimum). We assume that you have at least 100 MB in your user area for this purpose.

Check out from SVN

```
cd mydir
export SVNGRP=svn+ssh://svn.cern.ch/reps/atlasgrp
svn co $SVNGRP/Institutes/Arizona/micromegas/trunk micromegas
```

At this stage you will see the package being checked out and output looks something like this..
12:46]tb2009> svn co $SVNGRP/mumegas
A micromegas/input
A micromegas/mmcore/python/mamma.py
...
...
A micromegas/Makefile
Checked out revision 3436.

Setup environment to compile

```
cd micromegas
source setup.(c)sh
export PATH=$PYTHONDIR/bin:$PATH
python
```

At this stage the output looks like this..
1:21]micromegas> python
Python 2.5 (r25:51908, Oct 18 2007, 16:26:11)
[GCC 3.4.6 20060404 (Red Hat 3.4.6-8)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>

Compile the package

```
(g)make
```

At this stage the output looks like this..
12:46]micromegas> gmake
mamma --> Making RunInfo.d
mamma --> Making TdcDecoder.d
mamma --> Making NTupleHandler.d
mamma --> Making mplib.d
mamma --> Making mp.d
...
...
mamma --> Linking /afs/cern.ch/user/v/venkat/scratch0/tb2009/mumegas/bin/mumegas.exe
mamma --> Making Ana.d
mamma --> Compiling src/Ana.cxx
mamma --> Generating dictionary src/mumuser_Dict.cxx
mamma --> Compiling src/mumuser_Dict.cxx
mamma --> Making shared library: /afs/cern.ch/user/v/venkat/scratch0/tb2009/mumegas/lib/libmumuser.so
mamma --> All OK

+++Start application program `mamma -m` [This is to start the monitoring GUI]
`mamma -h` [This will display available switches]

The output looks like this
12:47]micromegas> mamma
+---+
Muon ATLAS MicroMegas Analysis
Usage: mamma [opts] args
Help : mamma --help or -h
+---+

Examples

Example 1: Convert Raw BAT/MM data for a single run into rootuple (single-file mode)

To run this example, you need two input files (BAT and MM raw data), the logbook and weather information. All the input parameters are specified using a configuration file which is provided in `input/test.config`.

```
mamma -c input/test.config
```

The output looks like this
--> mumegas.exe : Switching to batch mode
--> mumegas.exe : Reading configuration from : input/test.config
--> mumegas.exe : Dumping configuration
---
mmega.BATFile : /afs/cern.ch/user/v/venkat/scratch0/tb2009/run2711.bdt
mmega.MUMFile : /afs/cern.ch/user/v/venkat/scratch0/tb2009/run3448.raw
mmega.ROOTFile : input/TB09_MM3448_BAT2711.root
mmega.RunInfo : input/mm_run_book.csv
mmega.WeatherInfo : input/mm_temp.txt

rootFile = bdtFullPath = /afs/cern.ch/user/v/venkat/scratch0/tb2009/run2711.bdt
End of Run
Events counted: 10041
10041 events processed in total. Done.
PHOS MONITOR RECEIVED START OF RUN 3448.
PHOS MONITOR RECEIVED START OF RUN 3448.
PHOS MONITOR RECEIVED END OF RUN 3448.
Mamma: MONITOR RUN 3448
10044 events processed.
10045 events processed.
DONE - now exiting

Example 2: Convert Raw BAT/MM data for all runs from logbook (batch mode)

This job is performed using a different configuration where the path to input files are specified and the program loops over all the runs (which are read and stored from logbook) and creates the output rootuples in the destination specified. The runs are labeled as `TB_MM001234_BAT005678_sync00XX.root`. This job option is not for users interested in analysis, but for those involved in data-preparation.

Example 3: How to access RunInfo from standalone ROOT

- Setup MAMMA the usual way. Then you may access the RunInfo in the following way:

```
root [0] gSystem->Load("libHist.so");
root [2] gSystem->Load("libmmutil.so");
root [3] gSystem->Load("libmmcore.so");
root [4] TFile *_file0 = TFile::Open("SimpleFastReco_refit_before_4744.root")
root [5] .ls
TFile**         SimpleFastReco_refit_before_4744.root
 TFile*         SimpleFastReco_refit_before_4744.root
  KEY: TTree    ntp;1    ntp: combined mm and bat ntuple
  KEY: TTree    RunInfo4744MM;1    RunInfo for MM Run:  4744 and BAT Run  4017
root [6] RunInfo4744MM->Scan()
************************************************************************************************
*    Row   * runinfo.m * runinfo.b * runinfo.m * runinfo.b * runinfo.s * runinfo.e * runinfo.s *
************************************************************************************************
*        0 *      4744 *      4017 *     10053 *     10053 *      -999 *     ALTRO *       R13 *
************************************************************************************************
(Long64_t)1
root [7] .q
==========
```

Core Analysis Output

The analysis output is a ROOT ntuple containing the combined information (BAT and micromegas) along with run information for each run. The contents of the ntuple are tabulated below. It is assumed that most data analysis tasks listed above will be performed on this combined ntuple. A user analysis framework (similar to the core framework) has been developed and is available to use. As the analysis tasks proceed and mature, it is envisaged that they will be included into the core framework. The user analysis part is described in the next section.

Sl.	Datatype	Variable	Comment
1	Int_t	bat_run	BAT run number
2	vector < int >	bat_id1x	strip Index in layer 1
3	vector < int >	bat_q1x	charge of the strip in layer 1
4	vector < int >	bat_id2x	
5	vector < int >	bat_q2x	
6	vector < int >	bat_id3x	
7	vector < int >	bat_q3x	
8	vector < int >	bat_id6x	
9	vector < int >	bat_q6x	
10	vector < int >	bat_id1y	
11	vector < int >	bat_q1y	
12	vector < int >	bat_id2y	
13	vector < int >	bat_q2y	
14	vector < int >	bat_id3y	
15	vector < int >	bat_q3y	
16	vector < int >	bat_id6y	
17	vector < int >	bat_q6y	
18	Int_t	mm_run	MM run number
19	Int_t	mm_evt	No. of MM events in this run
20	Float_t	mm_time	Amplitude from gamma_2 fit for time channel from HIGHGAIN
21	Float_t	mm_timex	sampling time at maximum amplitude from gamm_2 fit for time channel from HIGHGAIN
22	Float_t	mm_time2	Amplitude from gamma_2 fit for time channel from LOWGAIN
23	Float_t	mm_timex2	sampling time at maximum mplitude from gamm_2 fit for time channel from LOWGAIN
24	Int_t	mm_qmaxStrId	Strip Id (after mapping) with highest charge in the event
25	Float_t	mm_maxStrQ	Amplitude of the strip Id with highest charge
26	Float_t	mm_maxStrT	Time of the strip Id with highest charge

28	vector < int >	mm_higain	1 : high gain 0: lowgain
29	vector < int >	mm_addr	channel address ranging from 1-128
30	vector < int >	mm_strid	Strip Id for channel after mapping
31	vector < int >	mm_qmaxsample	maximum sample charge before fit
32	vector < int >	mm_tqmaxsample	sampling time at maximum for before fit
33	vector < float >	mm_ped	pedestal
34	vector < float >	mm_noise	noise (RMS)

35	vector < float >	mm_qmaxfit	Amplitude from gamma_2 fit
36	vector < float >	mm_tqmaxfit	ampling time at maximum amplitude
37	vector < float >	mm_tau	tau parameter of gamma_2 function (width at halfmax)
38	vector < float >	mm_thalf	half-rising time

39	Int_t	mm_nclu	No. of clusters per event
40	vector < int >	mm_clsz	cluster size
41	vector < int >	mm_clstr0	strip Index of the beginning for a cluster
42	vector < int >	mm_clpkid	cluster peak id for strip
43	vector < float >	mm_clmnpos	cluster position from stripId mean
44	vector < float >	mm_clcogpos	cluster str wgt pos (mean position)
45	vector < float >	mm_clchgsum	cluster charge sum
46	vector < float >	mm_cltime	time from weighted average of strip time
47	vector < float >	mm_clpkchg	cluster peak charge

User Analysis

It is assumed that users are aware of setting up their environment to run ROOT (5.22 or higher), python (2.5 or higher) and a compatible gcc (g++) compiler. User analysis involves analyzing the output of the core analysis (ROOT tuple). The following operating systems / architectures are supported currently. If one wants to use others, they have to try it by themselves.

Operating Sytem	architecture	compiler
SLC 4, 5	i386, i686 (32, 64 bit)	gcc34, gcc43

Setup the environment to checkout and compile user packages

Download and save the configuration script dpdenv.slc5.i686.gcc43.sh to a directory of your choice. Everything will be installed under this base directory (called *your work area*). Download this script here

```
cd your_work_area
source dpdenv.slc5.i686.gcc43.sh
```

Check out

- Check out core and user packages

  ```
  svn co -q
  svn+ssh://venkat@svn.cern.ch/reps/atlasgrp/Institutes/Arizona/DPDAnalysis/
  core
  svn co -q
  svn+ssh://venkat@svn.cern.ch/reps/atlasgrp/Institutes/Arizona/DPDAnalysis/
  mmuser
  ```

  - Note 1: Replace venkat with your AFS user name
  - Note 2: You might have to authenticate yourself before you check out the packages using `kinit`

- Set up environment `source ./dpdroot_setup.sh` If file does not exists. Download/copy the file to the base directory.
- Check if `Makefile.mmuser` exists in the base directory. If not then do: `cp mmuser/dpd.root/Makefile ./Makefile.mmuser`
- Compile the package

  ```
  source ./setup.sh
  make -f Makefile.mmuser
  ```

Package Components

Run with the option --help for more information. Both files are in the path if environment is set up properly.
`bin/checkDPD root_file` prints out characteristics (variable names and their types) of the content of the root_file
`bin/runDPD -f list_file -c config_file` run analysis on files specified in the text file list_file. Default configuration file is mmuser/config/mmuser.config (use: -c mmuser/config/mmuser.config)

User Modules

Definition

A user module is a new class here called CSample defined in the following files (referenced from the base directory of the work area):
`./mmuser/src/sample.cxx` implementation file (Download)
`./mmuser/mmuser/sample.h` header file (Download)

The following class declaration defines minimum requirements for a module:

```
class CSample : public DPDAnalysisEvent {
  public :
    CSample(const std::string& name);
    CSample();
    ~ CSample() {;}
    void LinkData(DPDInput *data = 0);
    int ProcessEvent(DPDInput *data=0, Long64_t entry = 1, bool isLast = false);
    void Begin();
    void End();
  private:
    std::string m_name;
    Int_t           mm_run;
    ClassDef(CSample, 1)
};
```

where:
`CSample(const std::string& name);` //constructor to be called from text.cxx with a meaningful name
`void Begin();` // put all your initialization in this function called ad the beginning of the analysis
`void LinkData(DPDInput *data = 0);` // link root file data with your member variables here
`int ProcessEvent(DPDInput *data=0, Long64_t entry = 1, bool isLast = false);` // called for each event, do your processing here
`void End();` // called at the end of the analysis (i.e. save file here)

To pass message to the console use the m_log member (inherited from DPDAnalysisEvent):
`m_log << INFO  << "Begin user analysis. " << Endl;`

Invocation by the analysis package

The module is added to the analysis package by modifying two files:

`./mmuser/exec/test.cxx` where analysis module(s) are pushed back into a vector. The order in which they are pushed back is the order in which they are executed. Search for the declaration of
`std::vector<DPDAnalysis*> my;`
Create Sample class instance and add it to the list (change q2 to the next available number):

```
 CSample q2("samplemodule");
  my.push_back(&q2);
```

The second file `./mmuser/mmuser/mmuser_LinkDef.h` is used to inform ROOT of your class CSample. Add a line like the one below:

```
// user analysis
#pragma link C++ class MMBDT+;
#pragma link C++ class CSample+;
```

Finally recompile the analysis package.

Use your modules

To actually use the module it needs to be specified in the used configuration file:
Here we change the default: `./mmuser/config/mmuser.config`
Add module **name** as pushed back into the vector to the values list of the azdpd.Analysis key:

```
## name(s) of analysis modules to
## to run over tree(s) on event-by-event
## basis. use comma separated list
azdpd.Analysis: MMBDT , samplemodule
```

In the same file you specify names of the trees to be linked to the analysis:

```
## name(s) of the tree(s) to link
## use comma separated list
azdpd.TreeNames: ntp
```

Major updates:
-- VenkateshKaushik - 04 Aug 2009

Responsible: VenkateshKaushik
Last reviewed by: **Never reviewed**

- dpdenv.slc5.i686.gcc43.sh.txt: script to setup environment to run user analysis on lxplus machines (SLC5, gcc43)

This topic: Atlas > TB2009DataAnalysis
Topic revision: r9 - 05-Oct-2010 - MarcinByszewski