DAQ browser for MATLAB

Davit Kalantaryan
PPS
Zeuthen, 13.11.2014

HELMHOLTZ | ASSOCIATION

DESY

Content

- Introduction (reasons of creating new functions)
- C interface functions
- MATLAB functions
- Outlook
- Demo

Introduction

New C functions were created to read data from root files. Using MATLAB MEX interface these functions are available in MATLAB. Using these functions root files can be read directly from D-cache.

Current approach to handle data in root files from D-cache is to run a stand alone application from MATLAB, that stores data for predefined branches into file, then later on MATLAB loads this file to its workspace.

The advantages to have these functions in MATLAB are the following

a) **These functions match different branches by event number. The functions are able to get all data (those exist in root file). For example for the case of bad timing we often see frozen spectrums, but in reality all data are in root file, just time stamp for same event numbers differ for different branches.**
b) For current approach adding new branches or removing old branches can be done by changing stand alone application. In the case of these functions adding or removing branches means just to call corresponding function by other arguments.
c) New function directly and very quickly reads data from root file to MATLAB workspace.
d) In the case of using new approach we do not need AFS space for intermediate data storage. These intermediate files further on should be deleted by some person.

Platform where all of these functions are valid is SL5 (as for our all root dependent applications). Works should be done to have all our ROOT staff available in SL6 and may be even in the WINDOWS.

C interface functions

```
int ReadDataFromDCacheByEv2(
                int bufferItemsCount, struct19* bufferForData,
                int (*fpErrFunc)(const char*,...),
                int fromEvent, int toEvent,
                int numberOfBranches, const BranchItem* branchItems,
                void* notFoundedBranchesPtr = NULL,
                char* bufferForNames = NULL,
                int lenForBufForNames = 0,
                void* pRootFiles = NULL);

int ReadDataFromDCache(
                int bufferItemsCount, struct19* bufferForData,
                int (*fpErrFunc)(const char*,...),
                int fromTime, int toTime,
                int numberOfBranches, const BranchItem* a_pBranchItems);
```

Implementation of these functions are in the file browsing_funcs.cpp and declarations are in the file browsing_funcs.h. Sources can be used for C++ project directly. For C project libbrowsing_funcs.so with browsing_funcs.h header file should be used. All of these can be found in /doocs/develop/kalantar/public2/root_browsing_funcs AFS folder

MATLAB function for reading root files

From MATLAB these 2 C functions can be called from single MEX function. The name of this MEX function is **daq_browser**. Following are the input arguments of this function

1. verbosity: This argument shows verbosity, if this is 0, then all messages from ROOT library will not be outputted in MATLAB otherwise messages from ROOT library will be directed to MATLAB's main window.
2. option: this argument shows which C function will be called. If 0, then begin and end will be the starting event number and final event number otherwise begin is starting time in seconds and end is final time in seconds.
3. begin: depending on option argument shows either first interested event number, or first interested time in epoch seconds.
4. end: depending again on option argument shows final event number, or final epoch time in seconds
5. varargin: after 4-th argument variable number of string arguments can be provided. Each string argument shows branch name of interested branch.

First and more important return of this MEX function to MATLAB workspace is [NBxNE] array of structures. Where NB is Number of Branches and NE is Number of Event numbers. Structure consists of following fields

a) type
b) time
c) event
d) isAvailable
e) data

List of all MATLAB functions

- **daq_browser** (mex function) core function
- **createepoch** (mex function) converts LILI date string or 1x6 vector [year,month,day,hours,min,seconds] to epoch seconds
- **getmatlabtime** (mex function) converts epoch seconds to date string
- **testDataTm** example function. Open testDataTm.m to see the example of using all above mentioned MEX functions.
- **testDataEv** again example function

Some help on functions can be obtained by typing “help FuncName”

Code fragment that shows how to use MEX files

```
function example

vb = 0; %verbosity
ft = 1;  % function type (1 means search by time)
time1=createepoch('2014-10-22_02:55:59');
time2=createepoch('2014-10-22_02:56:40');
% or time2 = time1+41; % time interval is 41s

data = daq_browser(vb,ft,time1,time2, 'RF2Cpl10MWFW','RF2Cpl10MWRE','GUN__COUPLER__PMT_20140905',…);

beginTime = getmatlabtime(data(1,1).time);

figure(104)
plot(data(1,1).spectrum)
title(beginTime)
```

Still to be done

- All root stuff should be made available in SL6 (urgently).
- LILI a little bit modified (in the final demo new and old LILIs will be run, to see performance difference). New LILI should be finally checked by the physicist in order to start the new version.
- Now only for spectrum type functions implemented. Implementation for our all available cases is ongoing.
- JAVA classes for reading data from root files from D-cache will be implemented. In case the JAVA root API is available everything will be implemented purely in JAVA otherwise using JNA interface these C functions will be ported to JAVA.
- It is recommended that all our DAQ related QT applications are built in windows as JDDD is available also in windows. Necessary tools (libraries, headers,…) should be prepared for windows.
- Data with wrong time stamp (10.09.2014-10.10.2014 http://pitzlb.ifh.de:8080/PITZelog/jsp/show.jsp?dir=/2014/41/10.10_a&pos=2014-10-10T17:03:22) can be corrected ???

Acknowledgment

Thanks to Galina for providing example codes, and explaining many details connected to ROOT.

DEMO