EUDAQ for pALPIDEfs - Installation and usage manual

Jan Fiete Grosse-Oetringhaus

rev. 1, August 29, 2014

1 Prerequisites

You need to install and compile the pALPIDEfs software. For more information please refer to the corresponding manual which is located in the git repository [1] and on the sharepoint. Pay attention when you compile the software that you also build the shared library by calling `make lib`.

Further, you need qt, tinyxml and unbuffer installed which are usually available as system packages.

You need ROOT installed and in your search path (i.e. ROOTSYS, PATH and (DY)LD_LIBRARY_PATH environment variables set). For installation instructions, please refer to `http://root.cern.ch`. Note that if you don't have ROOT the compilation will still work. However, the GUI will not be built which is needed for taking data. You can see from the output of the cmake call if your ROOT installation was successfully found.

2 Installation

Check out EUDAQ from our repository which contains a slightly modified version of the one on github. However, we try to stay in sync with github as much as possible. You can clone the repository by calling:

```
git config --global http.sslVerify false
git clone https://pcalice100b.cern.ch/eudaq/eudaq-official.git
```

Currently this machine is only available within CERN. So from outside you would need to set up a tunnel.

Go to the resulting directory and execute

```
cmake -DBUILD_onlinemon:BOOL=ON -DBUILD_palpidefs:BOOL=ON
    -DCMAKE_PALPIDEFS_DRIVER_INCLUDE:FILEPATH=/path/to/software .
```

Please replace `/path/to/software` by the location of the pALPIDEfs software. E.g. if you checked it out to your home directory, you can give as path here `$HOME/pALPIDEfs-software/pALPIDEfs-software`.

Execute `make install` to build the software.

3 Configuration

The data-taking configuration is found in the file `conf/palpidefs.conf`. There are the following relevant general parameters:

- RunSizeLimit (in Bytes): when this limit for the current file is reached, the run is stopped and a new run is automatically started
- QueueSize (in MB): the memory reserved per layer to buffer events (relevant in case the event building is slower than the acquiring of events and one has a spilled beam structure).
- QueueFullDelay (in ms): how long will the system pause in case the queue is full
- StatusInterval (in s): how often the chip temperatures are read out and stored in the data stream
- ReadoutMode: 0 for event-based and 1 for packet-based readout. There must be the correct firmware on the card to support the selected mode.
- Devices: number of DAQ boards which should be used for data-taking. Each of them needs an own section, see below.

Per device the following parameters are available (in each parameter ID is to be replaced by the number of the device, running from 0 to Devices-1):

- BoardAddress_ID: Address of the DAQ board (configured with jumpers on the debug pins 7:0)
- Noisy_Pixel_File_ID (optional): File containing a list of noisy pixels which will be masked in the data-taking. The format of this file is discussed below.
- Config_File_ID (optional): configuration registers which are set after powering the chip. The format of this file is discussed below.

3.1 Noisy Pixel File

Each line contains one pixel to be masked which is identified by

```
REGION DOUBLECOLUMN ADDRESS
```

This format is compatible with the one used in the TDaqboard class.

3.2 Configuration File

The configuration file is an XML file which can also be used with the crystalball tool. It has a tree structure following the composition of the register addresses. As example the configuration of the VCASN and VCASP is given:

```
<?xml version="1.0"?>
<root>
  <address base="6">
    <address rgn="0">
      <address sub="1">
        <title>VCASN / VCASP</title>
        <value begin="0" width="8"><content>50</content></value>
        <value begin="8" width="8"><content>A0</content></value>
      </address>
    </address>
  </address>
</root>
```

Here the VCASN would be configured to 50 and the VCASP to A0. Note that the data values given in the content tag are **HEX** values.

4 Data Taking

To start the EUDAQ software execute

```
./STARTRUN
```

Four windows should open: the run control, the log window, the data collector and the pALPIDEfs producer.

The run control window allows to power and configure the chips. Select the right configuration from the dropdown (palpidefs.conf) and click Config. Messages should appear in the log window and in the producer which report the successful initialization. Subsequently, one can start/stop runs with the corresponding buttons. To power down the cards, close the run control window.

RAW data files are placed in into the `data` directory. Two types of log files are produced which are found in the `logs` directory. A file containing what has appeared in the log collector: this file is found in the logs directory and has the date in the filename. Further, the output of the producer is stored in the `logs/palpidefs` directory. The file name is the epoch time stamp when the producer was opened. These log files are for reference and usually not looked at.

5 Online Monitor

To check the produced events one can use the online monitor. Although the name suggests that it can be used only online it can be used with any RAW data file. It is started by

```
cd bin
./OnlineMon.exe -f FILENAME
```

By default the online monitor does not analyze each event. If this is desired, add the parameter `-sc 0`. The online monitor can also be started on a file which is of the ongoing run.

References

[1] https://git.cern.ch/web/pALPIDEfs-software.git