DB user Guide

L. Mirabito, C. Combaret, G. Baulieu

11 April 2019

1 Pedestal run

1.1 Data taking

One must first declare the alias *daqcontrol* to

```
alias daqcontrol='export DAQLOGIN=cmsLyon:RPC_XXXX;
export DAQURL=https://ilcconfdb.ipnl.in2p3.fr/config-content/febproto_cern_2;
                    /opt/lydaq/qt/daqcmd.py'
```

Then the trigger mode should be disable

```
 daqcontrol --daq-settdcmode --value=0
```

The VTHTIME scans is launch with

```
  daqcontrol --daq-scurve --chan=255 --first=450 --last=500 --step=2
             --spillon=20000 --mask=3
```

where

- *channel* is the PETIROC channel , or 1023 to scan individually each channel, or 255 to scan all channels
- *first* is the lowest VTHTIME value
- *last* is the highest VTHTIME value
- *step* is the VTHTIME step
- *spillon* is teh window width
- *mask* is a ASIC mask (1,2, or 3)

The program shows the value set and wait a Carriage Return to start. Once it's running you can stop it with Ctrl C but you then need to send a *daqcontrol –daq-stop* afterwards.

Once it's completed, the run number is found with :

```
daqcontrol --daq-evbstatus
```

1.2 Analysis

Uses the following commands

```
cd /opt/TdcAnalysis/
export LD_LIBRARY_PATH=.:./lib:$LD_LIBRARY_PATH
export PYTHONPATH=./scripts/:$PYTHONPATH
bin/tdcr -r748676 -d/data/local/FE1PR2/ -g/opt/TdcAnalysis/testshift3.json
```

It creates an histograms file in */opt/TdcAnalysis/Histos/InTime/histo748676_0.root*
The pedestals can be analysed with

```
 python -c ’import fitscurve as fs;fs.fitped(748676,11,455,520,1,48)’
 python -c ’import fitscurve as fs;fs.fitped(748676,11,455,520,2,48)’
 python -c ’import fitscurve as fs;fs.fitped(748676,12,455,520,1,48)’
 python -c ’import fitscurve as fs;fs.fitped(748676,12,455,520,2,48)’
```

2 DB changes

2.1 Environment

This access method will only work on debian jessie installation of LYDAQ.
The CONFDB environment variable should be set to the value store in */etc/ljc.conf*.

2.2 Python program

The python class containing the API to access to the DB is *TdcAccess*

```
NAME
    TdcAccess

FILE
    /opt/lydaq/apps/TdcCms/TdcAccess.py

CLASSES
    TdcAccess

    class TdcAccess
     |  Main class to access the DB
     |
     |  Methods defined here:
     |
     |  Change6BDac(self, idif, iasic, ich, dac)
     |      Change the 6BDAC value to dac  of the asic \#asic on the TDCDIF \#dif
     |
     |  ChangeAllEnabled(self, idif=0, iasic=0)
     |      Change all the ENable signals  of the asic \#asic on the TDCDIF \#dif
     |      If 0 all hardware is changed
     |
     |  ChangeDacDelay(self, delay, idif=0, iasic=0)
     |      Change the DACDELAY  of the asic \#asic on the TDCDIF \#dif
     |      If 0 all hardware is changed
     |
     |  ChangeLatch(self, Latch, idif=0, iasic=0)
     |      Change the Latch mode of the asic \#asic on the TDCDIF \#dif
     |      If 0 all hardware is changed
     |
     |  ChangeMask(self, idif, iasic, ich, mask)
     |      Change PETIROC2 MASKDISCRITIME parameter for one channel
     |      Careful: 1 = inactive, 0=active
     |
     |  ChangeVthTime(self, VthTime, idif=0, iasic=0)
     |      Change the VTHTIME  of the asic \#asic on the TDCDIF \#dif
     |      If 0 all hardware is changed
     |
     |  Correct6BDac(self, idif, iasic, cor)
     |      Change the 6BDAC value   of the asic \#asic on the TDCDIF \#dif
     |      cor is an array of 32 value ,
     |      6BDAC[i]=6BDAC[i]+cor[i]
     |
     |  DumpAsics(self)
     |      Print list of ASicx
     |
     |  DumpDIFs(self)
     |      return XML string of the list of DIFs
     |
     |  SetEnabled(self, idif=0, iasic=0, status=1)
     |      Change ENABLED status of the ASIC tagged by idif and iasic to the value status
     |       if idif is 0 all difs are concerned
     |       if iasic is 0 all asics are concerned
     |       status is 1 by default
     |
     |  SetSlowControlMask(self, idif, iasic=0)
     |      Mask Slow control for one DIF idif and 1 or all asics (iasic=0)
     |      The Masked DIF or Asic are not used for slow control
     |
     |  __init__(self, setupname=None)
     |      Download of the specified setup setupname or just initilise the DB if None
     |
     |  addAsic(self, dif_num, header)
     |      Add a new PETIROC2
     |      dif_num = DIF ID (ipaddr in integer LSB)
     |      header= ASIC number
     |
     |  addDCC(self, lda_address, lda_channel)
     |      (Obsolete) Keep for coherency ,add a DCCC
     |
     |  addDIF(self, ipaddr, nb_asic, lda_address=’ff:ff:ff:ff:ff:ff’, lda_channel=0,
     |           dcc_channel=0, xmlfile=None)
     |      Add a new DIF and load asics conf from file if any
     |      dif_num = DIF ID = LSB of the ipaddr in integer
     |      nb_asic = Number of ASICs
     |      lda_address = LDA MAC address (ff:ff:ff:ff:ff:ff)
     |      lda_channel = LDA Channel (0)
     |      dcc_channel = DCC Channel (0)
     |      xmlfile = XML list of ASICs (None)
     |
     |  addLDA(self, lda_address)
     |      (Obsolete) Keep for coherency ,Add an LDA
     |  dumpStateNames(self)
     |      print the list of states in the DB
     |
     |  end(self)
     |      close the DB
     |
     |  getDIFList(self)
     |      return the list of DIF in the current setup
     |
     |  initDif(self, addr)
     |      Default TDCDIF initialisation
     |
     |  initPR2(self, dif, num)
     |      PETIROC 2  initialisation
     |
     |  initialiseState(self, name)
     |      Create a new state named ’name’
     |
     |  setLDACabling(self, dif_id, ether_address, lda_channel, dcc_channel)
     |      Set the LDA cabling of the DIF dif_id
     |      for a in self.asics:
     |
     |  toXML(self)
     |      Save the current setup to setup_name.xml
     |
     |  uploadChanges(self)
     |      Upload a new version
     |      The state name will be, old_state_name_xx
     |      where xx is the new index (starting from 00)
     |
     |  uploadFromFile(self, setupname, fname)
     |      Upload a new setup from an XML file
     |      setupname = new setup name
     |      fname = XML file
     |
      |  uploadNewState(self, name)
      |      upload newly created state in oracle with state named 'name'
```

2.3 Examples

2.3.1 Create a new state

Create a state with FEB at address 11 with 2 asics

```
import TdcAccess as oa
s=oa.TdcAccess()
s.initialiseState("FEBNEW_11_1")
s.addDIF("192.168.10.11",2)
s.uploadNewState("FEBNEW_11_1")
```

2.3.2 Change the 6BDAC values of an ASIC

Correct the 6BDAC of the 2 asics of FEB 11

```
import TdcAccess as oa
s=oa.TdcAccess("FEBNEW_11_10")
cor11_1= [0, 0, 0, 0, 3, 2, 2, 2, 3, 1, 1, 2, 0, 1, 1, 0,
             -1, 1, 1, -1, 1, 1, 1, 0, 1, 0, 1, 0, -1, 0, 2, 0]
cor11_2= [0, 0, 0, 0, -2, -3, -3, -2, -2, -2, -2, -3, -2, -2, -2, -2, -2, -2, -2,
              -2, -2, -2, -3, -3, -2, 0, -2, 0, -2, 0, -2, 0]
s.Correct6BDac(11*256+10,1,cor11_1)
s.Correct6BDac(11*256+10,2,cor11_2)
s.uploadChanges()
```

The new state name is printed out at *uploadChanges*

2.4 Mask some channels

Mask unused PETIROC Channels of the 2 asics of FEB 11

```
import TdcAccess as oa
s=oa.TdcAccess("FEBNEW_11_1")
chm1=[0,1,2,3,25,27,29,31]

for c in chm1:
  s.Change6BDac(11*256+10,1,c,1)
  s.ChangeMask(11*256+10,1,c,1)
  s.Change6BDac(11*256+10,2,c,1)
  s.ChangeMask(11*256+10,2,c,1)

s.uploadChanges()
```