Mythen v2.0 manual

January 13, 2011

Chapter 1

Installation and upgrades

The new MYTHEN software is intended to control the MCS mythen boards either by using a command line interface (text client) or by using with a graphical user interface (GUI).

Here you can find in brief the main things you need to know in order to start working with your detector.

1.1 The software package

The actual software for the Mythen II system (MCS1 to MCS24) runs on 32 bit Scientific Linux machines (SLC5 tested, gcc 4.1.2 but it should not be critical).

The complete software package is composed of several programs which can be instaleld (or locally compiled) depending on the needs:

- The **slsDetector shared and static libraries** which are necessary for all user interfaces and can be simply used for implementig custom detector drivers;
- The **TSlsDetector shared and static libraries** which require root and Qt3.3 installation and are necessary to install the mytheGUI;
- The **command line interface (mythenClient) mythen_put, mythen_get, mythen_acquire** which is provided to communicate with the detectors;
- The **Graphical User Interface mythenGUI** which provides a user-friendly use of the detector;
- A **virtual server mythenServer** which can be used to simulate the behavior of the detector for what concerns the communication in case the detector is not online or is in use.

1.2 Requirements

For installing the slsDetector shared and static libraries and the mythenClient software, any Linux installation with a working gcc should be fine.

For installing the TSlsDetector shared and static libraries and the mythenGUI, working installations of Qt and Root should be present on the PC.

A Qt version equal or higher to 3.3 (but lower than Qt4) should be installed on the PC. It can be downloaded from `http://www.trolltech.com/developer/downloads/qt/index` and the enviroment variable `QTDIR` should be set to its path. Qt should be compiled with the options `-thread -no-xft -qt-gif -no-exceptions` (check in $QTDIR/config.status). this is normally the default on SLC machines.

Also a root version higher than 5.15 but lower than 5.22 **with Qt support enabled** should be installed (the binaries can be downloaded from `http://root.cern.ch` or compile with the option `--enable-qt`) and the system variable `ROOTSYS` should be set to its path.
Remember to check the in file $ROOTSYS/etc/system.rootrc or in your own custom .rootrc the following options are correctly defined:

```
# GUI specific settings
Gui.Backend: qt
Gui.Factory: qt
```

Further details can be found in the root user's manual at `http://root.cern.ch/root/doc/RootDoc.html` in chapter 1 (Introduction), 26 (ROOT/Qt integration interfaces) and 28 (Install and Build ROOT).

`ROOTSYS`/bin and `QTDIR`/bin should be added to the `PATH`;
`ROOTSYS`/lib and `QTDIR`/lib should be added to the `LD_LIBRARY_PATH` [1].

As an example, we suggest to add the following lines to your `.bashrc` or `.profile` file (changing `QTDIR` and `ROOTSYS` accordingly):

```
export ROOTSYS=/local/root
export QTDIR=/local/qt

export PATH=$QTDIR/bin:$ROOTSYS/bin::$PATH
export LD_LIBRARY_PATH=$QTDIR/lib:$ROOTSYS/lib:$LD_LIBRARY_PATH
export MANPATH=$QTDIR/doc/man/usr/local/man:$MANPATH
```

[1] In some linux installation there might be configuration scripts e.g in the directory `/etc/profile.d/` and `/etc/ld.so.conf.d/` overwriting this variables, so check that the installations correspond. If you have already used other Qt or Root versions, it might be that the old libraries are loaded in cache. To delete the chache, remove the file `/etc/ld.so.cache` and run ..... this is always a problem!!!!

1.3 Compilation

If you simply want to install the software in the working directory you can:

- `make lib` compile slsDetector library
- `make tlib` compile Root/Qt TSlsDetector library
- `make mythenClient` compile mythenClient package
- `make mythenGUI` compile mythenGUI
- `make all` compile slsDetector and TSlsDetector libraries, the mythenClient package and the mythenGUI
- `make clean` remove object files and executables
- `make help` lists possible targets

To be able to run the mythenClient and the mythenGUI as commands, add mythenClient/bin and mythenGUI/bin to your path.

1.4 Building

To install the software you should first configure some enviroment variables by executing:

```
> source configure
```

(NOT `>./configure` otherwise the enviroment variables will not be available for the `make` command). This allows you to configure:

- **QTDIR** i.e. the Qt installation directory. Ignore if you don't want to install the GUI, otherwise it should be defined in your `.bashrc` and added to `PATH` and `LD_LIBRARY_PATH`
- **ROOTSYS** i.e. the Root installation directory. Ignore if you don't want to install the GUI, otherwise it should be defined in your `.bashrc` and added to `PATH` and `LD_LIBRARY_PATH`
- **INSTALLROOT** Directory where you want to install the software. Defaults to /usr/local/
- **BINDIR** Directory where you want to install the binaries. Defaults to bin/
- **INCDIR** Directory where you want to pute the header files. Defaults to include/slsdetector/
- **LIBDIR** Directory where you want to install the libraries. Defaults to lib/

- **DOCDIR** Directory where you want to copy the documentation. Defaults to share/doc/

To build you can:

- `make install_lib` install detector library and include files"
- `make install_tlib` install detector Root/Qt library and include files"
- `make install_client` install mythenClient
- `make install_gui` install mythenGUI
- `make install` install library, include files, mythenClient and mythenGUI"
- `make install_libdoc` install library documentation
- `make install_clientdoc` install mythenClient documentation
- `make install_guidoc` install mythenGUI documentation
- `make install_doc` install all documentation
- `make help` lists possible targets

1.5 Detector upgrade

The upgrade of the detector consists in both the upgrade of the comminication software and of the firmware.

To upgrade the firmware you need either a working version of the Altera Quartus software or of the Quartus programmer, which can easly be downloade from
`https://www.altera.com/download/programming/quartus2/pq2-index.jsp`
Normally installation of the software and of the driver for the USB-Blaster (provided together with the MYTHEN detector) are simpler under Windows.
Under Windows, the first time that you connect the USB-Blasterto one of your USB ports, you will be asked to install new hardware. Set the path to search for the driver to: `C:\altera\80sp1\qprogrammer\drivers\usb-blasterp` (where `C:\altera\80sp1\qprogrammer\` is assumed to be ther path where your Quartus version is installed).

1. After starting the Quartus programmer, click on Hardware Setup and in the "Currently selected hardware" window select USB-Blaster.
2. In the Mode combo box select "Active Serial Programming".
3. Plug the end of your USB-Blaster WITH THE ADAPTER PROVIDED in the connector ASMI on the MCS board taking care that pin1 corresponds to the one indexed and with the rectangualr pad.

4. Click on add file and from select the programming file provided when the upgrade has been reccomended.
5. Check "Program/Configure" and "Verify".
6. Push the start button and wait until the programming process is finished (progress bar top left).
7. In case the programmer gives you error messages, check the polarity of your cable (pin1 corresponds) and that you have selected the correct programming connector.

To upgrade the software on the detector board transfer the provided software by ftp to the MCS:

```
ftp  mymcs.mydomain.com
username: root
password: pass
cd /mnt/flash/root
put mythenDetectorServer
quit
```

If the /mnt/flash/root directory does not exist, create it before the transfer by telnetting to the MCS.
After pressing reset on the board, the board should reboot.
If the program does not correctly start either check by using the http interface that it is started by the inittab (check that the file `/mnt/etc/inittab` ends with the line `myid2:3:once:/mnt/flash/root/mythenDetectorServer`).
Otherwise make the program executable by telnetting to the MCS and executing: `chmod a+xrw  /mnt/flash/root/mythenDetectorServer`
After pressing reset on the board, the board should reboot and the acqusition program correctly start.

1.6 The trimbits and calibration files

In order to be able to properly operate your detector you need a directory where the trimbit files (needed to set the detector settings and eventually equalize the individual channel thresholds) which in the following will be named *trimdir* and a directory where the calibration files (needed to convert the threshold energy in DAC units) are stored which in the following will be named *caldir*. *trimdir* and *caldir* can even be the same directory, and an example of it is given in the software package by the example directory `trimbits`.
Since these directories are customized by producing trimbit files and calibration for each detector, make sure not to overwrite yours every time you upgrade the software.

trimdir should contain three subdirectories `standard`, `fast` and `highgain` containing respectively the trimfiles `standard.trim`, `fast.trim` and `highgain.trim`

which contain the correct voltage settings for the detector although all the individual channel thresholds set to 0. The original files contained in the package should be used, infact in case of error the detector would not recognize the correct settings.
The default trimbit files for each file will be stored in the directory according to the settings with the name `noise.snxxx` where `xxx` is the module serial number.

caldir should contain three subdirectories `standard`, `fast` and `highgain` containing respectively the trimfiles `standard.cal`, `fast.cal` and `highgain.cal` which contain an average calibration of the modules for the diffrent settings. However this can different from the correct one for each individual module even of several kev and therefore it is very important to perform an energy calibration on a module basis (see section 4).
The default calibration files for each file will be stored in the directory according to the settings with the name `calibration.snxxx` where `xxx` is the module serial number.

Chapter 2

mythenClient

2.1 Introduction

This program is intended to control the MYTHEN detectors via command line interface.

To get all the possibilities of usage simply type:

mythen_acquire to readout the detector at full speed

mythen_put to set detector parameters

mythen_get to retrieve detector parameters

You will need to characterize your detector with a unique id (e.g. 0, but take care if you want to operate more than one detector in parallel!). Different detector types (e.g. MYTHEN, PICASSO etc.) must have different ids i.e. if you assign 0 to a MYTHEN detector you can't replace it with a PICASSO unless you reboot your PC.

2.2 Acquisition

mythen_acquire id

the detector is started and the data are acquired, postprocessed and written to file according to the configuration

2.3 Detector setup

mythen_put id:var arg

is used to configure the detector parameter var e.g. mythen_put 0:exptime 1 sets the exposure time to 1 s

The possibilites are:

help i get help

config fname reads the configuration file specified and sets the values

parameters fname sets the detector parameters specified in the file

setup rootname reads the files specfied (and that could be created by get setup) and resets the complete detector configuration including flatfield corrections, badchannels, trimbits etc.

hostname name this is mandatory!!!! sets hostname (or IP adress)

online b b can be 0 or 1 and sets the detector in offline/online state. Must be used to restore communication if some socket called failed because the detector was not connected.

status s either start or stop

caldir path Sets path of the calibration files

trimdir path Sets path of the trim files

outdir path directory to which the files will be written by default

fname name filename to which the files will be written by default (to which file and position indexes will eventually be attached)

index i start index of the files (automatically incremented by the acquisition functions)

nmod n Sets number of detector modules

extsig:i mode Sets usage of the external digital signal i. mode can be: off, gate_in_active_high, gate_in_active_low, trigger_in_rising_edge, trigger_in_falling_edge, ro_trigger_in_rising_edge, ro_trigger_in_falling_edge, gate_out_active_high, gate_out_active_low, trigger_out_rising_edge, trigger_out_falling_edge, ro_trigger_out_rising_edge, ro_trigger_out_falling_edge

settings sett Sets detector settings. Can be: standard fast highgain (depending on trheshold energy and maximum count rate: please refere to manual for limit values!);

threshold ev Sets detector threshold in eV. Should be half of the beam energy. It is precise only if the detector is calibrated

vthreshold dac Sets detector threshold in DAC units. A very rough calibration is dac=800-10*keV

exptime t Sets the exposure time per frame (in s)

period t Sets the frames period (in s)

delay t Sets the delay after trigger (in s)

gates n Sets the number of gates per frame

frames n Sets the number of frames per cycle (e.g. after each trigger)

cycles n Sets the number of cycles (e.g. number of triggers)

probes n Sets the number of probes to accumulate (max 3)

dr n Sets the dynamic range - can be (1,) 4, 8,16 or 24 bits

flags mode Sets the readout flags - can be none or storeinram

flatfield fname Sets the flatfield file name - none disable flat field corrections

ratecorr t Sets the rate corrections with dead time t ns (0 unsets, -1 uses default dead time for chosen settings

badchannels fname Sets the badchannels file name - none disable bad channels corrections

angconv fname Sets the angular conversion file name

globaloff o sets the fixed angular offset of your encoder - should be almost constant!

fineoff o sets a possible angular offset of your setup - should be small but can be senseful to modify

binsize s sets the binning size of the angular conversion (otherwise defaults from the angualr conversion constants)

positions np (pos0 pos1...posnp) Sets the number of positions at which the detector is moved during the acquisition and their values

threaded b Sets whether the postprocessing and file writing of the data is done in a separate thread (0 sequencial, 1 threaded). Please remeber to set the threaded mode if you acquire long real time measurements and/or use the storeinram option otherwise you risk to lose your data

2.4 Retrieving detector parameters (plus trimming and test modalities)

mythen_get id:var arg

is used to retrieve the detector parameter var e.g. mythen_get 0:exptime returns the exposure time in seconds

help This help

config fname writes the configuration file

parameters fname writes the main detector parameters for the measuremen tin the file

setup rootname writes the complete detector setup (including configuration, trimbits, flat field coefficients, badchannels etc.) is a set of files for which the extension is automatically generated

online return whether the detector is in online (1) or offline (0) state.

status gets the detector status - can be: running, error, transmitting, finished, waiting or idle

data gets all data from the detector (if any) processes them and writes them to file according to the preferences already setup

frame gets a single frame from the detector (if any) processes it and writes it to file according to the preferences already setup

hostname Gets the detector hostname (or IP address)

caldir Gets path of the calibration files

trimdir Gets path of the trim files

outdir directory to which the files will be written by default

fname filename to which the files will be written by default (to which file and position indexes will eventually be attached)

index start index of the files (automatically incremented by the acquisition functions)

nmod Gets number of detector modules

maxmod Gets maximum number of detector modules

extsig:i Gets usage of the external digital signal i. The return value can be: off, gate_in_active_high, gate_in_active_low, trigger_in_rising_edge, trigger_in_falling_edge, ro_trigger_in_rising_edge, ro_trigger_in_falling_edge, gate_out_active_high, gate_out_active_low, trigger_out_rising_edge, trigger_out_falling_edge, ro_trigger_out_rising_edge, ro_trigger_out_falling_edge

modulenumber Gets the module serial number

moduleversion Gets the module version

detectornumber Gets the detector number (MAC address)

detectorversion Gets the detector firmware version

softwareversion Gets the detector software version

digitest:i Makes a digital test of the detector module i. Returns 0 if it succeeds

bustest Makes a test of the detector bus. Returns 0 if it succeeds

settings Gets detector settings. Can be: standard fast highgain undefined

threshold Gets detector threshold in eV. It is precise only if the detector is calibrated

vthreshold Gets detector threshold in DAC units. A very rough calibration is dac=800-10*keV

exptime Gets the exposure time per frame (in s)

period Gets the frames period (in s)

delay Gets the delay after trigger (in s)

gates Gets the number of gates per frame

frames Gets the number of frames per cycle (e.g. after each trigger)

cycles Gets the number of cycles (e.g. number of triggers)

probes Gets the number of probes to accumulate (max 3)

dr Gets the dynamic range

trim:mode fname Trims the detector and writes the trimfile fname.snxxx. mode can be: noise beam improve fix offline - Check that the start conditions are OK!!!

flatfield fname returns whether the flat field corrections are enabled and if so writes the coefficients to the specified filename. If fname is none it is not written

ratecorr returns wether the rate corrections are enabled and what is the dead time used in ns

badchannels fname returns wether the bad channels corrections are enabled and if so writes the bad channels to the specified filename. If fname is none it is not written

angconv fname returns wether the angular conversion is enabled and if so writes the angular conversion coefficients to the specified filename. If fname is none, it is not written

globaloff returns the fixed angular offset of your encoder - should be almost constant!

fineoff returns a possible angualr offset of your setup - should be small but can be senseful to modify

binsize returns the binning size of the angular conversion

positions returns the number of positions at which the detector is moved during the acquisition and their values

threaded gets whether the postprocessing and file writing of the data is done in a separate thread (0 sequencial, 1 threaded). Check that it is set to 1 if you acquire long real time measurements and/or use the storeinram option otherwise you risk to lose your data

2.5 Tips

Mandatory setup

First of all you should setup the hostname and the detector size and dynamic range:

```
mythen_put 0:hostname mcs1x00
mythen_get 0:nmod
mythen_get 0:dr
```

You should also tell the program where to find the default trimbits files and calibration files:

```
mythen_put 0:trimdir /scratch/trimbits
mythen_get 0:caldir /scratch/calibration
```

To chose the detector settings (e.g. standard):

```
mythen_put 0:settings standard
```

In case `mythen_get 0:settings` does not answer correctly, it most probably means that there is a problem in the architecture or setting of *trimdir* and *caldir* (see section 1.6).

Acquisition setup

You need to setup where the files will be written to

```
mythen_put 0:outdir /scratch
mythen_put 0:fname run
mythen_put 0:index 0
```

this way your files will al be named /scracth/run_i.dat where is starts from 0 and is automatically incremented.

You will then need to setup the detector threshold and settings, the exposure time, the number of real time frames and eventually how many real time frames should be acquired:

```
mythen_put 0:settings standard
mythen_put 0:threshold 6000
mythen_put 0:exptime 1.
mythen_put 0:frames 10
```

In this case 10 consecutive 1s frames will be acquired. External gating and triggering or more advanced acquisition modes are not explained here.

Acquiring

There are two ways of acquiring data.
The first is fully automatic and freezes the terminal until the acquisition is finished:

```
mythen_acquire 0
```

This is particulary indicated for fast real time acquisitions.

If you want to acquire few long frames you can run:

```
mythen_put 0:status start
```

and the poll the detector status using

```
mythen_get 0:status
```

if the answer is either transmitting or finished, the data are ready to be downloaded from the detector. This can be done using either:

```
mythen_get 0:frame
```

where a single data frame is downloaded or

```
mythen_get 0:data
```

where all data present on the detector are downloaded. This is not indicated when many short real time frames should be acquired since the detector memory would be full before finishing the acquisition since the download time is so limited.

Data processing

Flat field and rate corrections can be applied direcly by simply selecting:

```
mythen_put 0:flatield myflatfield.raw
mythen_put 0:ratecorr -1
```

Chapter 3

mythenGUI

3.1 Introduction

To run the GUI just call:

`bin/mythenGUI`

Possible arguments are:

help This help

-f myconf.txt loads the configuration file to myconf.txt

-id i Sets the detector to id i (the default is i). Useful when more than one detector are operated in parallel.

-offline works in offline mode i.e. not connecting to the detector. Usefule e.g. to perform the energy calibration of the detector and possibly in the future to reprocess and visualize the data (not yet implemented).

-size n sets the size of the text to n (the default is n=10);

-scale s scales the size of the text and the root canvas by the scaling factor s (the default is s=1). It is useful when executing the program on a PC with low screen resolution (e.g. a laptop) and the window would then fall out of the screen.");

The configuration of the detector can either be set when startin the GUI using the configuration file or using the text client or even using the configuration tab of the GUI.

3.2 Acquisition

By pressing the start button in the measurement tab the data will be acquired, saved, corrected and plotted as specified.

The stop button stops the acquisition i.e. if there are data left to be saved processed etc. the program will not really stop until the offline processes are done.

Please don't be too nervous clicking on start and/or stop since this is one of the main causes of crashes (the program has been teste only for quiet users :-)).

3.3 Other functions

The text client and the GUI can be operated in parallel (althoug you should not change parameters or acquire data at the same data from the gui and the text client!) and the values displayed by the GUI should normally be the actual ones. However this kind of parallel operation is at your own risk!

The main parameters are group in tabs according to their meaning. To enable some tabs you should enter the modes menu and select Advanced/configuration/Debug Here is the general subject of the tabs:

Measurement Main acquisition parameters that you may want to change often

Data Output Where to write the data, in which format and what to to with them

Plot What to plot and how (only partially implemented)

Actions Allows to configure scans and/or execute scripts at teh beginning or at the end of the measurement.

Time resolved Parameters for time resolved (real time) measurements

Advanced Must be activated with the modes menu button. Allows to set some advanced configuration which you don't want general users to change (e.g. data size, external signals, advanced acquisition speed)

Trimming Must be activated with the modes menu button. Allows to trim the detector and/or load specific trim files.

Configuration Must be activated with the modes menu button. Allows to configure the detector

Debugging Must be activated with the modes menu button. Allows to test the detectors functionality, acquire serial numbers etc.

Most of the parameters are explained through a tooltip which appers if you leave the mouse on the widget for a few seconds.

The configuration and/or the complete setup of the detector can be loaded and saved using the Utilities menu.

3.3.1 Mandatory configuration

Where to find some important parameters (should be set only once, then it should remain in memory):

Hostname Configuration tab. Press enter to update.

Trim dir Configuration tab. Press enter to update.

Cal dir Configuration tab. Press enter to update.

Number of modules Configuration tab or Advanced tab

Dynamic range Advanced tab

Output directory Data Output tab.

File name Measurement tab.

File index Measurement tab (automatically incremented).

3.3.2 Acquisition setup

Where to find some important parameters (should be set only once, then it should remain in memory):

Settings Measurement tab

Threshold Measurement tab

Exposure time Measurement tab

Number of frames Measurement tab for non time-resolved measurement, Time resolved tab for fast real time measurements. if you need some action between frame see Actions tab.

Chapter 4

Energy calibration

The energy calibration should be performed by illuminating the detector with monochromatic radiation at at least 2 (better 3-4) energies larger than 8 keV. The energy calibration should be performed after trimming and the trim files used should be properly copied in the trimbits directory and used as default.

The data can be acquired either with the mythenGUI (by using the calibration wizard or the threshold scan utility in the Action tab) or with the mythenClient (by scanning the threshold using mythen_put 0:vthreshold), but since the analysis needs the use of root, the GUI must be used to finalize the calibration.

In the mythenGUI menu Utilities/Calibration wizard it is possible to simply and automatically perform the energy calibration of the detector:

1. Check the "Detector online" box in case you want to acquire the data, otherwise simply unclick it and you will be required to provide already acquired data and the details about the detector.
   The first time, chose "Start new calibration" and chose the directory where you want to store the data you want to acquire. The calibration file names have a".root" extension.
   The calibration should be perormed by acquiring always the same settings and with the same number of modules always connected in the same sequence. The clibration files, however, can be used for the modules also on different systems (i.e. different number of modules, readout board, etc.). A new calibration should be performed for different detector settings.

2. If the detector is online, the settings, the number of modules and their serial number will automatically be retrieved. If you selected the offline mode, you must provide the detector settings for the calibration that you want to perform and the serial numbers of the modules in the correct order (to do so, enter the 3 hexadecimal digits in the right sequence and press enter for each module - in case of error the list is editable).

3. Enter the energy of your beam (in keV!);

If you are in online mode, the acquisition time should be chosen such that there are at least 1000 counts per channel at an intemediate threshold; the range of the threshold scan should be between approx 800-15*keV and 800, better with a step of 1 but up to 5 can be fine in order to reduce the acquisition time: it is more important that each step has a sufficient statistics than that the threshold step is low! After pressing “Next”, the detector starts acquiring and showing the histogram of the calibration. When it is finished simply press “Finish” to accept the data, “Cancel” to reject them.
In offline mode, you are required to enter the range and step of the calibration and to select the files (in the same sequence as the threshold values!). After pressing “Next” (enabled only if the number of steps is the same as the number of files), the histogram showing the threshold scan is drawn. Simply press “Finish” to accept the data, “Cancel” to reject them.

4. For the following calibration steps, check the “Detector online” box in case you want to acquire the data, otherwise simply unclick it and you will be required to provide already acquired data and the details about the detector.
   Chose “Add calibration step” and select the file created prevously. The settings, number of modules and serial numbers of the modules and the energies at which the acquisition has been already performed should be displayed.

5. Add a new calibration step like in point 3. and iterate for all the energies at which you want to perform the calibration.

6. To generate the calibration files, chose “Generate calibration files” and select the file created prevously. The settings, number of modules and serial numbers of the modules and the energies at which the acquisition has been already performed should be displayed.

7. Chose the directory and the root of the calibrations files name. An extension corresponding to the serial number of the modules will be generated.

8. The calibration files for each module should be generated. For each energy you can set the start parameters of the fit and the fitting range (press enter after each change) so that the fitted curves nicely fit the data. The linear fit between energies and inflection points can also be checked.