SLS Detectors software installation

Anna Bergamaschi, Dhanya Thattil

August 24, 2018

Contents

1 The Software Package

The SLS detectors software is intended to control the detectors developed by the SLS Detectors group. The detectors currently supported are:

MYTHEN, GOTTHARD, EIGER and JUNGFRAU.

The package provides software for the distributed system that comprises of detectors, data receivers (to process detector data), and the client (to control or monitor the system). The client and data receivers can be embedded in the user's acquisitions system. Furthermore, the package also provides some tools for detector calibration.

1.1 Binaries

The complete software package is composed of several programs which can be installed (or locally compiled) depending on one's requirements:

- libSlsDetector.so, libSlsReceiver.so:
  The *slsDetector shared and static libraries*, which are necessary for all user interfaces. The *C++ API* via the class *slsDetectorUsers* (installed with the default package) or the *Python API* via the class *sls_detector* (installed with the package including Python API), which can be used from the user's acquisition software to control the detectors and the data receivers.

- sls_detector_put, sls_detector_get, sls_detector_acquire, sls_detector_help:
  The *command line interfaces*, which are provided to communicate with the detectors and data receivers using the command line.

- slsReceiver:
  The *data receiver*, which can be run on a different machine than the client, receives the data from the detector and processes it. The receiver can be configured, controlled and monitored by the client.

- slsMultiReceiver:
  It is the same as the *slsReceiver*, but that it is a single process for many multiple slsReceiver child processes. One can configure the start TCP port, number of slsReceiver processes and if call back should be enabled or not.

- slsDetectorGUI:
  The *graphical user interface*, which provides a user friendly way of operating the detectors and data receivers with online data preview.

- energyCalibrationWizard,angularCalibrationWizard:
  The *calibration wizards* to analyze the data and produce the energy or angular calibration files.

- The *virtual Detector servers* to simulate the detectors behavior. However, only control commands work, not the data acquisition itself.

2 Install Binaries via Conda

This section is useful only if one wants to download only the binaries for specific distribution and use the package via command line. Please refer later sections to download source code and compile them.

One can download and install Miniconda via
`https://conda.io/miniconda.html`

The conda package uses Travis CI for continuous integration with automatic deployment to Anaconda Cloud. One can download only the package or the package including the python interface.

After the installation, the binaries will be available in your path.

Please remember to clear shared memory after installation.

```
#displays list of shared memeory segments
ipcs -m
#remove segments that have nattach equal to zero. They key is the first column
ipcrm -M [key]
```

- Only the package

  ```
  #Add conda channels
  conda config --add channels conda-forge
  conda config --add channels slsdetectorgroup

  #Install latest version
  conda install sls_detector_lib
  conda install sls_detector_gui

  #Install specific release
  conda install sls_detector_lib=4.0.0
  conda install sls_detector_gui=4.0.0
  ```

- The package including Python interface

  ```
  #Add conda channels
  conda config --add channels conda-forge
  conda config --add channels sls_detector

  #Install latest version
  conda install sls_detector

  #Install specific release
  conda install sls_detector=4.0.0
  ```

3 Install via Source Code

This section is useful if one wants to use the API and embed it in their acquisition system, or if one wants to download the source code and compile.

3.1 Download Source Code

- Only the package

```
#Clone source code with specific release
git clone https://github.com/slsdetectorgroup/slsDetectorPackage.git --branch
4.0.0
```

- The package including Python interface

```
#Clone source code with specific release
git clone https://github.com/slsdetectorgroup/sls_detector.git --branch
4.0.0
```

3.2 Requirements

These are the basic requirements to install and use the software. Fine Tuning the system will be discussed in other documentation provided.

- *C/C++*:
  The software is written in C/C++. If Python API is used, it is a wrap around to the C++ software. Any Linux installation with working libgcc should be sufficient.

- *Shared Memory*:
  Access to the shared memory of the control PC is required for the client.

- *Network*:
  The control PC communicates to the detectors and data receivers over TCP/IP. Therefore, the detector should receive a proper IP address (either DHCP or static) and no firewall should be present between the control PC and the detector.

- *Compilation*:
  cmake is required to compile. make is also possible, but is harder to find dependencies.

- *GUI*:
  To use the GUI, one requires atleast Qt4.8.2 and Qwt6.0. Installation of these are discussed in the next sections.

- *Calibration Wizards*:
  They are based on the CERN Root data analysis framework. Installation of it is discussed in the next sections.

3.2.1 Qt4 Installation for GUI

It must be installed before Qwt. A Qt version equal or higher than 4.6 is required. One can install it:

- via YUM:

  ```
  yum install qt-devel
  ```

- via download from:
  `https://download.qt.io/archive/qt/4.8/4.8.2/qt-everywhere-opensource-src-4.8.2.tar.gz`
  To install:

  ```
  > gunzip qt-everywhere-opensource-src-4.8.2.tar.gz
  > tar xvf qt-everywhere-opensource-src-4.8.2.tar
  > ./configure
  > make
  > make install
  ```

  By default Qt4 will be installed in /usr/local/Trolltech/Qt-4.8.2/.

Setup Environment

One has to ensure that `PATH` and `LD_LIBRARY_PATH` have been updated to include Qt4 install path, binaries and libraries. Confirm by executing `qmake -v` and ensuring the result points to Qt4 (not Qt3 or Qt5).

If the environment is not set up, one can add the libraries and executables to the .bashrc by adding `LD_LIBRARY_PATH` and `PATH`:

```
export QTDIR=/usr/local/Trolltech/Qt-4.8.2
export PATH=$QTDIR/bin:$PATH
export LD_LIBRARY_PATH=$QTDIR/lib:$LD_LIBRARY_PATH
```

3.2.2 Qwt Installation for GUI

Before installing Qwt, one must install Qt and ensure that `QTDIR`, `LD_LIBRARY_PATH` and `PATH` point to the correct Qt4 version.

A Qwt version equal or higher than 6 is required. One can install it:

- via YUM:

  ```
  yum install qwt-devel
  ```

- via download from:
  `https://sourceforge.net/projects/qwt/files/qwt/6.0.0/qwt-6.0.0.zip/download`
  To install:

```
> cd qwt-6.0.0
> qmake
> make
> make install
```

By default Qwt will be installed int /usr/local/qwt-6.0.0

Setup Environment

One has to ensure that `QWTDIR` and `LD_LIBRARY_PATH` have been updated to include Qwt install path and libraries.

If the environment is not set up, one can add the libraries to the .bashrc by adding `LD_LIBRARY_PATH`:

```
export QWTDIR=/usr/local/qwt-6.0.0/
export LD_LIBRARY_PATH=$QWTDIR/lib:$LD_LIBRARY_PATH
```

3.2.3 Root Installation for Calibration Wizards

The software has been developed and tested with root 5.20, but any version should work. One can download it from:

```
> svn co https://root.cern.ch/svn/root/trunk root
```

To install:

```
> cd root
> ./configure --enable-qt
> make
> make install
```

Edit your .bashrc to define the ROOTSYS enviroment variable and annd the libraries and executables to the `LD_LIBRARY_PATH` and `PATH`:

```
export ROOTSYS=/usr/local/root-5.34
export PATH=$ROOTSYS/bin:$PATH
export LD_LIBRARY_PATH=$ROOTSYS/lib:$LD_LIBRARY_PATH
```

You can also download the binaries, assuming that your linux and gcc versions match as in:

```
http://root.cern.ch/drupal/content/production-version-534
```

3.3 Compilation

One requires `cmake` to compile and can be done in two ways:

3.3.1 Using script cmk.sh

The script uses `cmake`. After compiling, the libraries and executables will be found in 'slsDetectorPackage/build/bin' directory. Usage: [-c] [-b] [-h] [-d HDF5 directory] [-j]

- -[no option]: only make
- -c: Clean
- -b: Builds/Rebuilds CMake files normal mode
- -h: Builds/Rebuilds Cmake files with HDF5 package
- -d: HDF5 Custom Directory
- -t: Build/Rebuilds only text client
- -r: Build/Rebuilds only receiver
- -g: Build/Rebuilds only gui
- -j: Number of threads to compile through

Some example options for compilation:
Most basic option: `./cmk.sh -b`
For only make: `./cmk.sh`
For make clean;make: `./cmk.sh -c`
For using hdf5 without custom dir /blabla: `./cmk.sh -h -d /blabla`
For rebuilding cmake without hdf5: `./cmk.sh -b`
For using multiple cores to compile faster: `./cmk.sh -j9`
For rebuilding only certain parts: `./cmk.sh -tg` (only text client and gui)

3.3.2 Directly using cmake

Use cmake to create out-of-source builds, by creating a build folder parallel to source directory.

```
$ cd ..
$ mkdir slsDetectorPackage-build
$ cd slsDetectorPackage-build
$ cmake ../slsDetectorPackage  -DCMAKE_BUILD_TYPE=Debug -DUSE_HDF5=OFF
$ make
```

Use the following as an example to compile statically and using specific hdf5 folder

```
$ HDF5_ROOT=/opt/hdf5v1.10.0 cmake ../slsDetectorPackage
-DCMAKE_BUILD_TYPE=Debug -DUSE_HDF5=ON
```

After compiling, the libraries and executables will be found at 'bin' directory

```
 $ ls bin/
    gui_client  libSlsDetector.a  libSlsDetector.so  libSlsReceiver.a
libSlsReceiver.so    sls_detector_acquire  sls_detector_get  slsDetectorGui
sls_detector_help sls_detector_put  slsReceiver
```

3.4 Setting environment variables

One can set up the environment variables in the following ways.

3.4.1 Using .bashrc file

1. `emacs ~/.bashrc`
2. Add the following function `setup_slsdet` and replace `path` with absolute path of installed directory

   ```
   function setup_slsdet
   {
   export PKGPATH=[path]
   export LD_LIBRARY_PATH=$PKGPATH/slsDetectorPackage/build/bin:$LD_LIBRARY_PATH
   export PATH=$PKGPATH/slsDetectorPackage/build/bin:$PATH
   cd $PKGPATH/slsDetectorPackage/build/bin
   }
   ```

3. `source ~/.bashrc`
4. Next time, just run `setup_slsdet` to load the environment variables.

One can also add the GUI environment variables if installed locally by adding the following in the function `setup_sldet`

```
export QTDIR=/path-where-it-is/Qt-4.8.2
export QWTDIR=/path-where-it-is/qwt-6.0.1
export QWT3D=/path-where-it-is/qwtplot3d
export QMAKESPEC=$QTDIR/mkspecs/linux-g++
export LD_LIBRARY_PATH=$QTDIR/lib:$QWTDIR/lib:$QWT3D/lib:$LD_LIBRARY _PATH
export PATH=$QTDIR/bin:$PATH
```

3.4.2 Without .bashrc file

Go to binaries folder slsDetectorPackage/build/bin and execute the following:

```
export LD_LIBRARY_PATH=$PWD:$LD_LIBRARY_PATH
export PATH=$PWD:$PATH
```

3.5 Clean Shared Memory

It is very crucial to clean the shared memory, before using a new version of the SLS Detector Package or a different detector type.

One can use the `cleansharedmemory.sh` script available under the slsDetector Package.

One can also just delete the files that are typically located under /dev/shm/ folder and starts with slsDetectorPackage.

One no longer has to delete segments using ipcs.

4 Software Upgrade

The upgrade of the package could require an upgrade of the on-board detector server and/or firmware running on the detector as well.

4.1 MYTHEN

In such cases, the users are not expected to compile the software themselves (which would require dedicated softwares) but only to download on the detector board the programming files and/or software package provided by the SLS Detectors group.

4.1.1 MYTHEN Firmware

To upgrade the firmware you need either a working version of the Altera Quartus software or of the Quartus programmer, which can easily be downloaded 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-Blaster to 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.

4.1.2 MYTHEN On-board Software

1. Connect to the board using telnet:

```
telnet  mymcs.mydomain.com
username: root
password: pass
```

2. Kill currently running servers and ensure `/mnt/flash/root` exists.

```
killall mythenDetectorServer
ls /mnt/flash/root
#if the directory does not exist mkdir /mnt/flash/root
```

3. Transfer the provided software by ftp to the MCS.

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

4. After pressing reset on the board, the board should reboot.

5. If the program does not correctly start

   (a) 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`).

   (b) If program has not started, make the program executable by telnetting to the MCS and executing:
   `chmod a+xrw  /mnt/flash/root/mythenDetectorServer`

   (c) After pressing reset on the board, the board should reboot and the acqusition program correctly start.

4.2 GOTTHARD

In such cases, the users are not expected to compile the software themselves (which would require dedicated softwares) but only to download on the detector board the programming files and/or software package provided by the SLS Detectors group.

4.2.1 GOTTHARD Firmware

For SLS Detector Package v4.0.0

Minimum compatible version:

11.01.2013

Latest version:

26.06.2018 (50um)
08.02.2018 (25 um Master)
09.02.2018 (25 um Slave)

Normally, the firmware will be upgraded by us as it requires programming the FPGA via the USB-Blaster.

To upgrade the firmware you need either a working version of the Altera Quartus software or of the Quartus programmer, which can easily be downloaded 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-Blaster to 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.

4.2.2 GOTTHARD On-board Software

Every SLS Detector package release will have its coresponding matching on-board server under **slsDetectorPackage/serverBin**.

1. Install tftp if the pc does not have it.
2. Copy the server from serverBin folder to /tftpboot (or equivalent tftp folder) of the pc
3. Copy the server to the detector by:
   (a) Connect to the blackfin on the detector
   `telnet bchipxxx`
   (b) Prevent existing on-board server from respawning by:
      i. Edit `/etc/inittab`
      ii. Comment out the line `#ttyS0::respawn:/gotthardDetectorServervxxx`
      iii. Reboot blackfin using `reboot`
      iv. Run `ps` to ensure no gotthardDetectorServers are running
   (c) Copy new on-board server from pc to the blackfin using:
   `tftp pcxxx -r gotthardDetectorServerxxx -g`
   (d) Respawn the new server (server starts at detector statup):
      i. Edit `/etc/inittab`
      ii. Uncomment out the line `ttyS0::respawn:/gotthardDetectorServervxxx`
      iii. Reboot blackfin using `reboot`
      iv. Run `ps` to ensure that both the gotthardDetectorServers are running.
      `gotthardDetectorServerxxx`
      `gotthardDetectorServerxxx 1953`

4.3 EIGER

In such cases, the users are not expected to compile the software themselves (which would require dedicated softwares) but only to download on the detector board the programming files and/or software package provided by the SLS Detectors group.

4.3.1 EIGER Firmware

For SLS Detector Package v4.0.0
Minimum compatible version: 22
Latest version: 22

1. One must get the latest package's corresponding bit files from the SLS Detector Group.
2. If one does not have the bcp script, that should also be obtained from the SLS Detector Group. It is required to program the bit files and requires that tftp be installed on the pc.
3. Bring the detector into programmable mode by either of the following ways. Both ways end up in just the central LED blinking.
   (a) hard reset on the back panel boards resulting in blinking LEDS
   (b) by having the following program running in the background.

   ```
   boot_recovery
   ```

4. Start a terminal for each half module and run the following to see progress.

   ```
   nc -p 3000 -u bebxxx 3000
   ```

5. Run the following to update firmware

   ```
    #update back end fpga
   bcp download.bit bebxxx:/fw0

    #update front left fpga
   bcp download.bit bebxxx:/febl

    #update front right fpga
   bcp download.bit bebxxx:/febr

    #update kernel
   bcp download.bit bebxxx:/kernel
   ```

   Please update bit files with great caution as it could make your board inaccessible, if done incorrectly.

4.3.2 EIGER On-board Software

Every SLS Detector package release will have its coresponding matching on-board server under **slsDetectorPackage/serverBin**.

Update the on-board software without connecting to the detector

```
#password for the boards: root

#Kill existing servers that are running on the detector
ssh root@beb031 killall eigerDetectorServer;

#Copy on-board server to detector inside executables folder
scp ~/path-where-it-is/eigerDetectorServerxxx root@bebxxx:~/executables;

#Overwrite the actual eigerDetectorServer on board
scp ~/path-where-it-is/eigerDetectorServerxxx
root@bebxxx:~/executables/eigerDetectorServer;

#sync
ssh root@bebxxx sync;

#reboot the eiger board
```

One can connect to the detector by:

```
ssh root@bebxxx
password: root
```

The on-board server is in /executables folder and respawned at startup in `/etc/rc5.d/S50board_com.sh`

4.4 JUNGFRAU

In such cases, the users are not expected to compile the software themselves (which would require dedicated softwares) but only to download on the detector board the programming files and/or software package provided by the SLS Detectors group.

4.4.1 JUNGFRAU Firmware

For SLS Detector Package v4.0.0
Minimum compatible version: 15.06.2018
Latest version: 15.06.2018

At times, one has to update the firmware, which then also requires updating the on-board software.

Jungfrau firmware can be upgraded via the SLS Detector Package binaries from the command line.

1. One must get the latest package's corresponding POF file from the SLS Detector Group.
2. Update the latest SLS Detector package installed.

3. Update the on-board software as per the instructions in the next section.

4. Start the on-board server in debug mode:

   (a) Connect to the blackfin on the detector
   `telnet bchipxxx`

   (b) Prevent existing on-board server from respawning by:

      i. Edit `/etc/inittab`

      ii. Comment out the line `#ttyS0::respawn:/jungfrauDetectorServervxxx`

      iii. Reboot blackfin using `reboot`

      iv. Run `ps` to ensure no gotthardDetectorServers are running

   (c) Start the server in debug mode using:
   `./jungfrauDetectorServerxxx -debug`
   Leave this console on to come back to it later.

5. From the command line of the pc, clear shared memory
   `./sls_detector_get free`
   If one gets shmget error, please clean the shared memory properly using the script in `slsDetectorPackage/cleansharedmemory.sh`

6. Add the detector to shared memory using
   `./sls_detector_put hostname bchipxxx`

7. Program the FPGA using
   `./sls_detector_put programfpga xxx.pof`

8. Once the programming is done:

   (a) Switch to the console that has the debug server running and kill it using Ctrl+C and ensure no jungfrauDetectorServers are running

   (b) Restart the new server to see if it runs with the new firmware
   `./jungfrauDetectorServerxxx`
   If the server didn't start properly, please contact us with the error message shown when starting the server up, else continue with the following steps.

   (c) Respawn the new server (server starts at detector statup):

      i. Edit `/etc/inittab`

      ii. Uncomment out the line `ttyS0::respawn:/jungfrauDetectorServervxxx`

      iii. Reboot blackfin using `reboot`

      iv. Run `ps` to ensure that both the gotthardDetectorServers are running.
      `jungfrauDetectorServervxxx`
      `jungfrauDetectorServervxxx 1953`

4.4.2 JUNGFRAU On-board Software

Every SLS Detector package release will have its coresponding matching on-board server under **slsDetectorPackage/serverBin**.

1. Install tftp if the pc does not have it.
2. Copy the server from serverBin folder to /tftpboot (or equivalent tftp folder) of the pc
3. Copy the server to the detector by:
   (a) Connect to the blackfin on the detector
   `telnet bchipxxx`
   (b) Prevent existing on-board server from respawning by:
      i. Edit `/etc/inittab`
      ii. Comment out the line `#ttyS0::respawn:/jungfrauDetectorServervxxx`
      iii. Reboot blackfin using `reboot`
      iv. Run `ps` to ensure no gotthardDetectorServers are running
   (c) Copy new on-board server from pc to the blackfin using:
   `tftp pcxxx -r jungfrauDetectorServervxxx -g`
   (d) Respawn the new server (server starts at detector statup):
      i. Edit `/etc/inittab`
      ii. Uncomment out the line `ttyS0::respawn:/jungfrauDetectorServervxxx`
      iii. Reboot blackfin using `reboot`
      iv. Run `ps` to ensure that both the gotthardDetectorServers are running.
      `jungfrauDetectorServervxxx`
      `jungfrauDetectorServervxxx 1953`