My Project

Generated by Doxygen 1.8.5

Thu Aug 23 2018 15:46:39

Contents

1 File Documentation

1.1 mainClient.cpp File Reference

```
#include "slsDetectorUsers.h"
#include "detectorData.h"
#include <iostream>
#include <cstdlib>
```

Functions

- int dataCallback (detectorData ∗pData, int iframe, int isubframe, void ∗pArg)
- int main (int argc, char ∗∗argv)

1.1.1 Detailed Description

This file is an example of how to implement the slsDetectorUsers class You can compile it linking it to the slsDetector library

g++ mainClient.cpp -L lib -lSlsDetector -L/usr/lib64/ -L lib2 -lzmq -pthread -lrt -lm -lstdc++

where,

lib is the location of libSlsDetector.so

lib2 is the location of the libzmq.a. [libzmq.a is required only when using data call backs and enabling data streaming from receiver to client. It is linked in manual/manual-api from slsReceiverSoftware/include]

Definition in file mainClient.cpp.

1.1.2 Function Documentation

1.1.2.1 int dataCallback (detectorData ∗ *pData,* int *iframe,* int *isubframe,* void ∗ *pArg*)

Data Call back function defined

Parameters

pData	pointer to data structure received from the call back
iframe	frame number of data passed
isubframe	sub frame number of data passed (only valid for EIGER in 32 bit mode)
pArg	pointer to object

Returns

integer that is currently ignored

Definition at line 32 of file mainClient.cpp.

1.1.2.2 int main (int *argc*, char ∗∗ *argv*)

Example of a main program using the slsDetectorUsers class

- Arguments are optional
  - argv[1] : Configuration File
  - argv[2] : Measurement Setup File
  - argv[3] : Detector Id (default is zero)
- if specified, set ID from argv[3]
- slsDetectorUsers Object is instantiated with appropriate ID
- if specified, load configuration file (necessary at least the first time it is called to properly configure advanced settings in the shared memory)
- set detector in shared memory online (in case no config file was used)
- set receiver in shared memory online (in case no config file was used)
- registering data callback
- ensuring detector status is idle before starting acquisition. exiting if not idle
- if provided, load detector settings
- start measurement
- returning when acquisition is finished or data are avilable
- delete slsDetectorUsers object

Definition at line 49 of file mainClient.cpp.

1.2 mainReceiver.cpp File Reference

```
#include "sls_receiver_defs.h"
#include "slsReceiverUsers.h"
#include <iostream>
#include <string.h>
#include <signal.h>
#include <cstdlib>
#include <sys/types.h>
#include <sys/wait.h>
#include <string>
#include <unistd.h>
#include <errno.h>
#include <syscall.h>
```

Macros

- #define PRINT_IN_COLOR(c, f,...) printf ("\033[%dm" f RESET, 30 + c+1, ##__VA_ARGS__)

Functions

- void sigInterruptHandler (int p)
- void printHelp ()
- int StartAcq (char ∗filepath, char ∗filename, uint64_t fileindex, uint32_t datasize, void ∗p)
- void AcquisitionFinished (uint64_t frames, void ∗p)
- void GetData (char ∗metadata, char ∗datapointer, uint32_t datasize, void ∗p)
- void GetData (char ∗metadata, char ∗datapointer, uint32_t &revDatasize, void ∗p)
- int main (int argc, char ∗argv[])

Variables

- bool keeprunning

1.2.1 Detailed Description

This file is an example of how to implement the slsReceiverUsers class You can compile it linking it to the slsReceiver library

g++ mainReceiver.cpp -L lib -lSlsReceiver -L/usr/lib64/ -L lib2 -lzmq -pthread -lrt -lm -lstdc++

where,

lib is the location of lSlsReceiver.so

lib2 is the location of the libzmq.a. [libzmq.a is required only when using data call backs and enabling data streaming from receiver to client. It is linked in manual/manual-api from slsReceiverSoftware/include]

Definition in file mainReceiver.cpp.

1.2.2 Macro Definition Documentation

1.2.2.1 #define PRINT_IN_COLOR(*c*, *f*, *...*) printf ("\033[%dm" f RESET, 30 + c+1, ##__VA_ARGS__)

Define Colors to print data call back in different colors for different recievers

Definition at line 38 of file mainReceiver.cpp.

1.2.3 Function Documentation

1.2.3.1 void AcquisitionFinished (uint64_t *frames*, void ∗ *p*)

Acquisition Finished Call back

Parameters

frames	Number of frames caught
p	pointer to object

Definition at line 85 of file mainReceiver.cpp.

1.2.3.2 void GetData (char ∗ *metadata*, char ∗ *datapointer*, uint32_t *datasize*, void ∗ *p*)

Get Receiver Data Call back Prints in different colors(for each receiver process) the different headers for each image call back.

Parameters

metadata	sls_receiver_header metadata
datapointer	pointer to data
datasize	data size in bytes.
p	pointer to object

Definition at line 98 of file mainReceiver.cpp.

1.2.3.3 void GetData (char ∗ *metadata,* char ∗ *datapointer,* uint32_t & *revDatasize,* void ∗ *p*)

Get Receiver Data Call back (modified) Prints in different colors(for each receiver process) the different headers for each image call back.

Parameters

metadata	sls_receiver_header metadata
datapointer	pointer to data
datasize	data size in bytes.
revDatasize	new data size in bytes after the callback. This will be the size written/streamed. (only smaller value is allowed).
p	pointer to object

Definition at line 132 of file mainReceiver.cpp.

1.2.3.4 int main (int *argc,* char ∗ *argv[]*)

Example of main program using the slsReceiverUsers class

- Defines in file for:
  - Default Number of receivers is 1
  - Default Start TCP port is 1954
- set default values
- get number of receivers and start tcp port from command line arguments
- Catch signal SIGINT to close files and call destructors properly
  - Ignore SIG_PIPE, prevents global signal handler, handle locally, instead of a server crashing due to client crash when writing, it just gives error
- loop over number of receivers
- fork process to create child process
- if fork failed, raise SIGINT and properly destroy all child processes
- if child process
- create slsReceiverUsers object with appropriate arguments

  ```
  - register callbacks. remember to set file write enable to 0 (using the client)
  ```

  if we should not write files and you will write data using the callbacks
- Call back for start acquisition
- Call back for acquisition finished
- start tcp server thread
- as long as keeprunning is true (changes with Ctrl+C)

- interrupt caught, delete slsReceiverUsers object and exit
- Parent process ignores SIGINT (exits only when all child process exits)
- Print Ready and Instructions how to exit
- Parent process waits for all child processes to exit

Definition at line 167 of file mainReceiver.cpp.

1.2.3.5 void printHelp ( )

prints usage of this example program

Definition at line 55 of file mainReceiver.cpp.

1.2.3.6 void sigInterruptHandler (int *p*)

Control+C Interrupt Handler Sets the variable keeprunning to false, to let all the processes know to exit properly

Definition at line 48 of file mainReceiver.cpp.

1.2.3.7 int StartAcq (char ∗ *filepath,* char ∗ *filename,* uint64_t *fileindex,* uint32_t *datasize,* void ∗ *p*)

Start Acquisition Call back slsReceiver writes data if file write enabled. Users get data to write using call back if registerCallBackRawDataReady is registered.

Parameters

filepath	file path
filename	file name
fileindex	file index
datasize	data size in bytes
p	pointer to object

Returns

ignored

Definition at line 72 of file mainReceiver.cpp.

1.2.4 Variable Documentation

1.2.4.1 bool keeprunning

Variable is true to continue running, set to false upon interrupt

Definition at line 42 of file mainReceiver.cpp.

Index