Interface for Sick Laser Measurement Systems

Generated by Doxygen 1.8.4

Wed Apr 2 2014 09:29:52

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

Class Documentation

3.1 pacpus::_scanCfg Struct Reference

Structure containing scan configuration.

`#include <SickLMSData.h>`

Public Attributes

- int scaningFrequency

  Scanning frequency. 1/100 Hz.
- int angleResolution

  Scanning resolution. 1/10000 degree.
- int startAngle

  Start angle. 1/10000 degree.
- int stopAngle

  Stop angle. 1/10000 degree.

3.1.1 Detailed Description

Structure containing scan configuration.

Author

Based on Konrad Banachowicz work.

The documentation for this struct was generated from the following file:

- SickLMSData.h

3.2 pacpus::_scanData Struct Reference

Structure containing single scan message.

`#include <SickLMSData.h>`

Public Attributes

- u_int32_t scanFrequency

  Scanning frequency. 1/100 Hz.
- u_int32_t angleResolution

  Scanning resolution. 1/10000 degree.
- int32_t startAngle

  Start angle. 1/10000 degree.
- int dist_len1

  Number of samples in dist1.
- uint16_t * dist1

  Radial distance for the first reflected pulse.
- int dist_len2

  Number of samples in dist2.
- uint16_t * dist2

  Radial distance for the second reflected pulse.
- int dist_len3

  Number of samples in dist3.
- uint16_t * dist3

  Radial distance for the first reflected pulse.
- int dist_len4

  Number of samples in dist4.
- uint16_t * dist4

  Radial distance for the second reflected pulse.
- int dist_len5

  Number of samples in dist5.
- uint16_t * dist5

  Radial distance for the second reflected pulse.
- int rssi_len1

  Number of samples in rssi1.
- uint16_t * rssi1

  Energy values for the first reflected pulse.
- int rssi_len2

  Number of samples in rssi2.
- uint16_t * rssi2

  Energy values for the second reflected pulse.

3.2.1 Detailed Description

Structure containing single scan message.

Author

Based on Konrad Banachowicz work.

The documentation for this struct was generated from the following file:

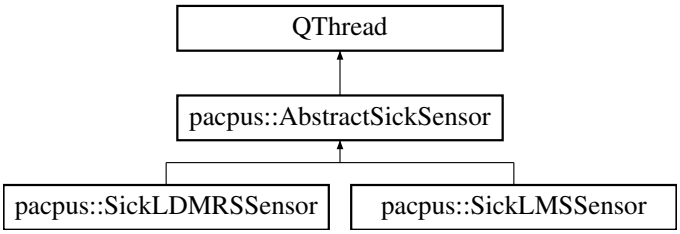
- SickLMSData.h

3.3 pacpus::AbstractSickSensor Class Reference

The AbstractSickSensor class.

`#include <AbstractSickSensor.h>`

Inheritance diagram for pacpus::AbstractSickSensor:



Public Slots

- virtual void customEvent (QEvent ∗e)=0

  customEvent is a slot called when connected to a signal.

Public Member Functions

- void **run** ()
- virtual void stopActivity ()=0
- virtual void startActivity ()=0

Public Attributes

- SickSocket ∗ **S_socket**

Protected Attributes

- QString host_

  The SickLDMRS IP or hostname.
- int port_

  The SickLDMRS port.
- bool recording

  If data need to be recorded, set this member to true.

3.3.1 Detailed Description

The AbstractSickSensor class.

The AbstractSickSensor class provides the abstract model for implementing sensor interfaces using SickComponent, used with the PACPUS Framework.

3.3.2 Member Function Documentation

3.3.2.1 virtual void pacpus::AbstractSickSensor::customEvent (QEvent ∗ *e*) [pure virtual], [slot]

customEvent is a slot called when connected to a signal.

Parameters

e	QEvent that carries information from sensor.

Sick sensor interface implementation uses this slot to get data from the remote sensor, using IP packets. As long as Sick sensors use TCP/IP interface, it is advised to use this slot to get the packets from the device.

See Also

SickFrameEvent

3.3.2.2 virtual void pacpus::AbstractSickSensor::startActivity ( ) [pure virtual]

to start the processing thread

Implemented in pacpus::SickLDMRSSensor, and pacpus::SickLMSSensor.

3.3.2.3 virtual void pacpus::AbstractSickSensor::stopActivity ( ) [pure virtual]

to stop the processing thread

Implemented in pacpus::SickLDMRSSensor, and pacpus::SickLMSSensor.

The documentation for this class was generated from the following file:

- AbstractSickSensor.h

3.4 pacpus::DataHeader Struct Reference

The DataHeader struct.

`#include <SickLDMRSData.h>`

Public Attributes

- u_int32_t magicWord

  0xAFFEC0C2 for the Sick LDMRS sensor (this value must be found in order to decode the message).
- u_int32_t sizePreviousMessage

  Size in bytes of the previous message.
- u_int32_t sizeCurrentMessage

  Size of the message content without the header (DataHeader).
- u_int8_t deviceId

  Unused in data received directly from LD-MRS sensors.
- u_int16_t dataType
- u_int64_t ntpTime

  Time of the sensor when the message is created.

3.4.1 Detailed Description

The DataHeader struct.

The DataHeader struct describes general information about the message used with. On Sick LDMRS, DataHeader corresponds exactly to the very first data carried into the whole message. See Ethernet data protocol LD-MRS, page 4.

Warning : the data from the sensor is coded in Big Endian format.

3.4.2 Member Data Documentation

3.4.2.1 u_int16_t pacpus::DataHeader::dataType

Type of information carried into the message. Types used are :

- Points : 0x2202
- Objects : 0x2221

The documentation for this struct was generated from the following file:

- SickLDMRSData.h

3.5 pacpus::MessageLDMRS Class Reference

The class carrying Sick LDMRS message.

`#include <SickLDMRSSensor.h>`

Public Member Functions

- MessageLDMRS ()

  Constructor.
- ~MessageLDMRS ()

  Destructor.

Public Attributes

- DataHeader hData

  An instance of DataHeader.
- ScanHeader hScan

  An instance of ScanHeader (if data type is scan points).
- char * body

  An array of characters : raw data then array of points or objects, depending on data type.
- road_time_t time

  Time when the message is received.
- road_timerange_t timerange

  Timerange : roughly, time between measurement of a point and the processing step (not implemented).

3.5.1 Detailed Description

The class carrying Sick LDMRS message.

This class is used so that we can store every information sent by a Sick LDMRS sensor. First, the raw data is stored in `body`. These data are then decoded and general information about the message is stored in DataHeader and ScanHeader (Object data decoding is not implemented yet). Then, the body field is replaced by a ScanPoint or ScanObject array in order to be stored in DBT/UTC files.

3.5.2 Member Data Documentation

3.5.2.1 char* pacpus::MessageLDMRS::body

An array of characters : raw data then array of points or objects, depending on data type.

This array pointer points to allocated in memory (basically, in heap (malloc)) and then must be freed (free) when the whole message is decoded and stored.

The documentation for this class was generated from the following file:

- SickLDMRSSensor.h

3.6 pacpus::MessageLMS Class Reference

The class carrying Sick LMS message.

`#include <SickLMSSensor.h>`

Public Member Functions

- MessageLMS ()

  Constructor.
- ~MessageLMS ()

  Destructor.

Public Attributes

- scanData data

  Every needed information about the scan (general info + scan points).
- long msgSize

  Size of the message.
- std::vector< std::string > * splitMessage

  The message is split into an array of string in order to be processed easily.
- char * body

  Raw data.
- road_time_t time

  Time when the first packet of the message is received.
- road_timerange_t timerange

  Timerange : roughly, time between measurement of a point and the processing step (not implemented).

3.6.1 Detailed Description

The class carrying Sick LMS message.

This class is used so that we can store every information sent by a Sick LMS sensor. First, the raw data is stored in `body`. Then, if the message is relevant, `splitMessage` is instanciated in order to parse easily information from the sensor. These data are then decoded and stored in the scanData structure.

The documentation for this class was generated from the following file:

- SickLMSSensor.h

3.7 pacpus::MessagePacket Struct Reference

Structure used to stored Sick data between several decoding processes.

`#include <AbstractSickSensor.h>`

Public Attributes

- road_time_t **time**
- std::string **data**
- bool **previousData**

3.7.1 Detailed Description

Structure used to stored Sick data between several decoding processes.

Note that data coming from sensors are split in IP packets. In order to get the whole message in one block, MessagePacket is used to reconstitue the raw data. As we want to reconstitute the message, it is useful to know if another packet belonging to the message was previously received. Also, the time of the reception of the first packet is carried into the structure in order to trace scans.

The documentation for this struct was generated from the following file:

- AbstractSickSensor.h

3.8 pacpus::ScanHeader Struct Reference

The ScanHeader struct.

`#include <SickLDMRSData.h>`

Public Attributes

- u_int16_t scanNumber

  Number of the scan since the sensor started measuring.
- u_int16_t scannerStatus

  Status of the scanner.
- u_int16_t **phaseOffset**
- u_int64_t startNtpTime

  NTP time first measurement.
- u_int64_t endNtpTime

  NTP time last measurement.
- u_int16_t ticksPerRot

  Angle ticks per rotation (used to compute the real angle of a point)
- int16_t startAngle

  Angle of the first measured value.
- int16_t endAngle

  Angle of the last measured value.
- u_int16_t numPoints

  Number of scanned points during this scan.

3.8.1 Detailed Description

The ScanHeader struct.

General information about points measured. Data type is 0x2202

See Also

DataHeader

see Ethernet data protocol LD-MRS page 5

3.8.2 Member Data Documentation

3.8.2.1 u_int16_t pacpus::ScanHeader::numPoints

Number of scanned points during this scan.

See Also

ScanPoint

3.8.2.2 u_int16_t pacpus::ScanHeader::scannerStatus

Status of the scanner.

- 0x0007: reserved,
- 0x0008: set frequency reached,
- 0x0010: external sync signal detected,
- 0x0020: sync ok,
- 0x0040: sync master (instead of slave),
- 0xFF80: reserved

The documentation for this struct was generated from the following file:

- SickLDMRSData.h

3.9 pacpus::ScanObject Struct Reference

The ScanObject struct (not used)

`#include <SickLDMRSData.h>`

3.9.1 Detailed Description

The ScanObject struct (not used)

Used to describe an object. Data type 0x2221

See Also

DataHeader

The documentation for this struct was generated from the following file:

- SickLDMRSData.h

3.10 pacpus::ScanPoint Struct Reference

The ScanPoint struct.

`#include <SickLDMRSData.h>`

Public Attributes

- u_char layerEcho
- u_char **flags**
- u_int16_t angle
- u_int16_t distance

  Distance of the point from the sensor in centimeters.
- u_int16_t echoPulseWidth

  Width of echo pulse (cm)

3.10.1 Detailed Description

The ScanPoint struct.

Used to describe a point. Data type 0x2202

See Also

DataHeader

3.10.2 Member Data Documentation

3.10.2.1 u_int16_t pacpus::ScanPoint::angle

Angle in number of ticks. You can easily compute the real angle : $angle(degree) = \frac{angle(ticks)}{ScanHeader.ticksPerRot}$

See Also

ScanHeader

3.10.2.2 u_char pacpus::ScanPoint::layerEcho

4 LSB : Layer (scan layer of the point) 4 MSB : Echo

The documentation for this struct was generated from the following file:

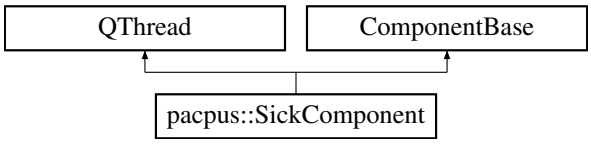
- SickLDMRSData.h

3.11 pacpus::SickComponent Class Reference

The SickComponent class.

`#include <SickComponent.h>`

Inheritance diagram for pacpus::SickComponent:



Public Member Functions

- SickComponent (QString name)

  Constructor.
- ~SickComponent ()

  Destructor.
- void **run** ()
- virtual void stopActivity ()
- virtual void startActivity ()
- virtual ComponentBase::COMPONENT_CONFIGURATION configureComponent (XmlComponentConfig config)

  Configure compenent.

Public Attributes

- SickComponent * **myParent**

3.11.1 Detailed Description

The SickComponent class.

This class defines a PACPUS component used to acquire Sick lidars data.

3.11.2 Member Function Documentation

3.11.2.1 ComponentBase::COMPONENT_CONFIGURATION pacpus::SickComponent::configureComponent (XmlComponentConfig *config*) [virtual]

Configure compenent.

Parameters

config	XML file passed in order to configure the Sick Component.

This function instanciate every sensors configured through the XML file. The XML file must be formated as expected by this function. Depending on used sensors and how many, you should define these three property in a "Sick" node (X must start from '0') :

- sickldmrs_X
- sicklms151_X
- sicklms511_X

For example, let's say we have two Sick LMS151, one LDMRS and one LMS511 :

```
<Sick type="SickComponent" sickldmrs_0="192.168.0.1:2111" sicklms151_0="192.-
168.0.10:2111" sicklms151_1="192.168.0.11:2111" sicklms511_0="192.168.1.50-
:2111">
```

Do not forget `type="SickComponent"`.

3.11.2.2 void pacpus::SickComponent::startActivity ( ) [virtual]

To start the processing thread

3.11.2.3 void pacpus::SickComponent::stopActivity ( ) [virtual]

To stop the processing thread

The documentation for this class was generated from the following files:

- SickComponent.h
- SickComponent.cpp

3.12 pacpus::SickFrame Class Reference

The SickFrame class.

`#include <SickSocket.h>`

Public Member Functions

- SickFrame ()

  SickFrame constructor.
- ~SickFrame ()

  Destructor.

Public Attributes

- qint64 size

  Size of incoming packet.
- road_time_t time

  Time when packet is received.
- char * msg

  Packet (raw data).

3.12.1 Detailed Description

The SickFrame class.

The documentation for this class was generated from the following file:

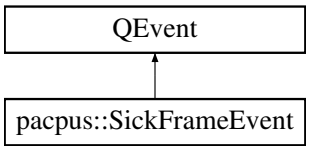
- SickSocket.h

3.13 pacpus::SickFrameEvent Class Reference

The SickFrameEvent class QEvent that encapsulates packets.

`#include <SickSocket.h>`

Inheritance diagram for pacpus::SickFrameEvent:



Public Member Functions

- SickFrameEvent ()

  Constructor.
- ~SickFrameEvent ()

  Destructor.

Public Attributes

- SickFrame * frame

  Packet data.

3.13.1 Detailed Description

The SickFrameEvent class QEvent that encapsulates packets.

The documentation for this class was generated from the following file:

- SickSocket.h

3.14 pacpus::SickLDMRS_dbt Struct Reference

The SickLDMRS_dbt struct.

`#include <SickLDMRSData.h>`

Public Attributes

- u_int64_t timeStartFromSensor

  NTP time (creation of the message on sensor).
- ScanHeader hScan

  General information about points recorded.
- road_time_t time

  DBT timestamp.
- road_timerange_t timerange

  DBT timerange.
- int32_t dataPos

  The position of the data in the binary file associated to the dbt file (utc file).

3.14.1 Detailed Description

The SickLDMRS_dbt struct.

Data recorded in the DBITE file (.dbt).

3.14.2 Member Data Documentation

3.14.2.1 ScanHeader pacpus::SickLDMRS_dbt::hScan

General information about points recorded.

See Also

ScanHeader

The documentation for this struct was generated from the following file:

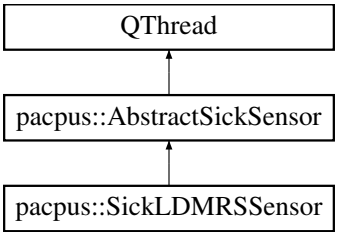
- SickLDMRSData.h

3.15 pacpus::SickLDMRSSensor Class Reference

The class implenting receiving, decoding and storing process of Sick LD-MRS data.

`#include <SickLDMRSSensor.h>`

Inheritance diagram for pacpus::SickLDMRSSensor:



Public Slots

- void customEvent (QEvent ∗e)

  customEvent allows to receive the incoming data and store them into known structures.
- void configure ()

  Configure the object, not used for the moment.

Public Member Functions

- SickLDMRSSensor (QObject ∗parent)

  Constructor.
- SickLDMRSSensor (QObject ∗parent, QString name, QString ip, int port, int recording)

  SickLDMRSSensor constructor.
- ∼SickLDMRSSensor ()

  Destructor.
- void **run** ()
- void stopActivity ()
- void startActivity ()
- void splitPacket (const char ∗packet, const int length, road_time_t time)

  splitPacket reconstitute incoming data and find messages.
- unsigned long processMessage (MessageLDMRS &msg)

  Process/decode a message.
- u_int32_t findMagicWord (const char ∗message, const unsigned length)

Find the position of the magic word into the array and returns this index.

- u_int32_t getMessageSize (const char *message, const unsigned length, const long magicWordIndex)

  getMessageSize get the message size of the entire message.

- bool isMessageComplete (const unsigned length, const long size)

  isMessageComplete compare the size of the message read into the message and the length of the received data.

Public Attributes

- SickSocket * S_socket

  S_socket, used to receive and send data to the remote sensor.

Additional Inherited Members

3.15.1 Detailed Description

The class implenting receiving, decoding and storing process of Sick LD-MRS data.

This class can be used as a particular thread to acquire data from Sick LDMRS sensors. The Ethernet interface is used to get data from the sensor. Thus, the goal of this class is to get packets and decode them. Also, it offers the possibility to store all relevant information in two files (.dbt and .utc). It can be managed by SickComponent objects.

3.15.2 Constructor & Destructor Documentation

3.15.2.1 pacpus::SickLDMRSSensor::SickLDMRSSensor (QObject * *parent*, QString *name*, QString *ip*, int *port*, int *recording*)

SickLDMRSSensor constructor.

Parameters

parent	Basically, a SickComponent object.
name	Name of the sensor in order to write on .dbt and .utc files and to recognize every sensors used.
ip	The IP address of the remote Sick LDMRS sensor.
port	The port of the remote Sick LDMRS sensor.
recording	If `true`, data is recorded into dbt + utc files. Data is not recorded otherwise.

3.15.3 Member Function Documentation

3.15.3.1 void pacpus::SickLDMRSSensor::customEvent (QEvent * *e*) [slot]

customEvent allows to receive the incoming data and store them into known structures.

Parameters

e	Event that carries the Ethernet packets and receiving time.

3.15.3.2 u_int32_t pacpus::SickLDMRSSensor::findMagicWord (const char * *message*, const unsigned *length*)

Find the position of the magic word into the array and returns this index.

Parameters

message	Array of characters, raw data received from sensor.
length	Length of the array.

Returns

- **-1** if no magic word is found
- **position** of the magic word otherwise

3.15.3.3 u_int32_t pacpus::SickLDMRSSensor::getMessageSize (const char * *message*, const unsigned *length*, const long *magicWordIndex*)

getMessageSize get the message size of the entire message.

Parameters

message	Raw data of the message.
length	Length of the raw data received.
magicWordIndex	First element of the message, used to get the size of the message.

Returns

The **size** of the whole message.

The size of the message is found inside the message thanks to an offset after the index of the Magic Word.

- The first header of the message that contains the size of the message is in **Big Endian** format !

3.15.3.4 bool pacpus::SickLDMRSSensor::isMessageComplete (const unsigned *length*, const long *size*)

isMessageComplete compare the size of the message read into the message and the length of the received data.

Parameters

length	Length of the received data.
size	Size of the message read. See getMessageSize.

Returns

true if the message is complete, **false** otherwise

3.15.3.5 unsigned long pacpus::SickLDMRSSensor::processMessage (MessageLDMRS & *msg*)

Process/decode a message.

Parameters

msg	The message is encapsulated into a MessageLDMRS

Returns

Type of the message

Process the raw data of the message and update the MessageLDMRS object passed : it fills the 2 headers (message and scan) and replace the body field of the MessageLDRMS object by an array of ScanPoint.

- **Warning** : the process of object data type is not implemented yet !

3.15.3.6 void pacpus::SickLDMRSSensor::splitPacket (const char * *packet,* const int *length,* road_time_t *time*)

splitPacket reconstitute incoming data and find messages.

Parameters

packet	Raw data coming from the sensor.
length	Length of the data.
time	Time of the last received data.

Analyse the ethernet packet received from the Sick sensor and try to find a complete message (scan data message or object message) If a message has been found it is added at the end of the message list else the pending bytes are stored to be analyzed by further incoming data.

3.15.3.7 void pacpus::SickLDMRSSensor::startActivity ( ) [virtual]

To start the processing thread

Implements pacpus::AbstractSickSensor.

3.15.3.8 void pacpus::SickLDMRSSensor::stopActivity ( ) [virtual]

To stop the processing thread

Implements pacpus::AbstractSickSensor.

The documentation for this class was generated from the following files:

- SickLDMRSSensor.h
- SickLDMRSSensor.cpp

3.16 pacpus::SickLMS_dbt Struct Reference

Public Attributes

- u_int16_t scanNumber

  number of the scan
- u_int16_t scannerStatus
- road_time_t time

  DBT timestamp.
- road_timerange_t timerange

  DBT timerange.
- u_int32_t scanFrequency

  Frequency of the scan [1/100 Hz].
- u_int32_t angleResolution

  Angle resolution (default is 5000 <=> 0.5 degree) [1/10000 degree].
- int32_t startAngle

  Angle of the first scanned point.
- int dist_len1

  Number of points (1st echo).
- uint32_t dataPos_dist1

  Distance between the sensor and the remote point (1st echo).
- int dist_len2

  Number of points (2nd echo).
- uint32_t dataPos_dist2

  Distance between the sensor and the remote point (2nd echo).
- int dist_len3

  Number of points (3rd echo).

- uint32_t dataPos_dist3

  Distance between the sensor and the remote point (3rd echo). ***LMS5xx only****.*
- int dist_len4

  Number of points (4th echo).
- uint32_t dataPos_dist4

  Distance between the sensor and the remote point (4th echo). ***LMS5xx only****.*
- int dist_len5

  Number of points (5th echo).
- uint32_t dataPos_dist5

  Distance between the sensor and the remote point (5th echo). ***LMS5xx only****.*
- int rssi_len1

  Number of energy values (1st echo).
- uint32_t dataPos_rssi1

  Energy of the returned pulse (1st echo). ***LMS1xx only****.*
- int rssi_len2

  Number of energy values (2nd echo).
- uint32_t dataPos_rssi2

  Energy of the returned pulse (2nd echo). ***LMS1xx only****.*

3.16.1 Member Data Documentation

3.16.1.1 u_int16_t pacpus::SickLMS_dbt::scannerStatus

- 00 00 OK
- 00 01 Error
- 00 02 Pollution Warning
- 00 04 Pollution Error

The documentation for this struct was generated from the following file:

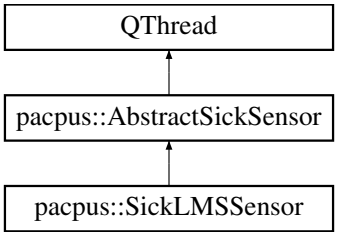
- SickLMSData.h

3.17 pacpus::SickLMSSensor Class Reference

The class implenting receiving, decoding and storing process of Sick LMS data.

`#include <SickLMSSensor.h>`

Inheritance diagram for pacpus::SickLMSSensor:



Public Slots

- void customEvent (QEvent ∗e)

  customEvent allows to receive the incoming data and store them into known structures.
- void configure ()

  Configure the object, not used for the moment.

Public Member Functions

- SickLMSSensor (QObject ∗parent)

  Constructor.
- SickLMSSensor (QObject ∗parent, QString name, QString ip, int port, int recording)

  SickLMSSensor constructor.
- ∼SickLMSSensor ()

  Destructor.
- void **run** ()
- void stopActivity ()
- void startActivity ()
- void reconstituteMessage (const char ∗packet, const int length, road_time_t time)

  reconstituteMessage reconstitute a complete message from received packets
- int processScanData (MessageLMS ∗msg)

  processScanData Parse information and process every needed values.
- int isMessageComplete (const char ∗packets, unsigned int size)

  isMessageComplete find the <ETX> character (corresponding to the end of a message).

Public Attributes

- SickSocket ∗ S_socket

  S_socket, used to receive and send data to the remote sensor.

Additional Inherited Members

3.17.1 Detailed Description

The class implenting receiving, decoding and storing process of Sick LMS data.

This class can be used as a particular thread to acquire data from Sick LDMRS sensors. The Ethernet interface is used to get data from the sensor. Thus, the goal of this class is to get packets and decode them. Also, it offers the possibility to store all relevant information in two files (.dbt and .utc). It can be managed by SickComponent objects.

3.17.2 Constructor & Destructor Documentation

3.17.2.1 pacpus::SickLMSSensor::SickLMSSensor (QObject ∗ *parent*, QString *name*, QString *ip*, int *port*, int *recording*)

SickLMSSensor constructor.

Parameters

parent	Basically, a SickComponent object.

name	Name of the sensor in order to write on .dbt and .utc files and to recognize every sensors used.
ip	The IP address of the remote Sick LMS sensor.
port	The port of the remote Sick LMS sensor.
recording	If `true`, data is recorded into dbt + utc files. Data is not recorded otherwise.

3.17.3 Member Function Documentation

3.17.3.1 void pacpus::SickLMSSensor::customEvent (QEvent * *e*) [slot]

customEvent allows to receive the incoming data and store them into known structures.

Parameters

e	Event that carries the Ethernet packets and receiving time.

3.17.3.2 int pacpus::SickLMSSensor::isMessageComplete (const char * *packets*, unsigned int *size*)

isMessageComplete find the <ETX> character (corresponding to the end of a message).

Parameters

packets	Raw data.
size	Size of raw data.

Returns

The index of the <ETX> character.

3.17.3.3 int pacpus::SickLMSSensor::processScanData (MessageLMS * *msg*)

processScanData Parse information and process every needed values.

Parameters

msg	Carries a message. splitMessage field of MessageLMS must be filled.

Returns

Not used for the moment.

3.17.3.4 void pacpus::SickLMSSensor::reconstituteMessage (const char * *packet*, const int *length*, road_time_t *time*)

reconstituteMessage reconstitute a complete message from received packets

Parameters

packet	Raw data coming from the sensor.
length	Length of the raw data received.
time	Time of the last received data.

A message starts with a <STX> (0x02 in ASCII) char and ends with <ETX> (0x03 in ASCII). This function stores packets until a complete message is received. In this case, the message is added to the list of MessageLMS, msgList.

3.17.3.5 void pacpus::SickLMSSensor::startActivity ( ) [virtual]

To start the processing thread. TODO get response from sensor and analyse it to know if measuring has started See p23 telegram

Implements pacpus::AbstractSickSensor.

3.17.3.6 void pacpus::SickLMSSensor::stopActivity ( ) [virtual]

To stop the processing thread.

Implements pacpus::AbstractSickSensor.

The documentation for this class was generated from the following files:

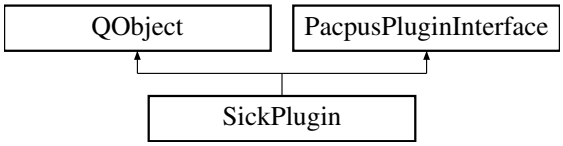
- SickLMSSensor.h
- SickLMSSensor.cpp

3.18 SickPlugin Class Reference

Auto-generated plugin class.

`#include <SickPlugin.h>`

Inheritance diagram for SickPlugin:



Protected Member Functions

- QString **name** ()

3.18.1 Detailed Description

Auto-generated plugin class.

The documentation for this class was generated from the following files:

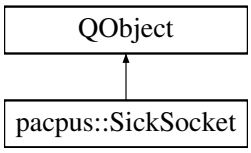
- SickPlugin.h
- SickPlugin.cpp

3.19 pacpus::SickSocket Class Reference

The SickSocket class Handles the ethernet connection with the remote sensor.

`#include <SickSocket.h>`

Inheritance diagram for pacpus::SickSocket:



Public Slots

- void connectToServer (QString host, int port)

  Enable the connection to the server.
- int socketConnected ()

  Warns about connection of the socket and launch configuration of the socket.
- void socketReadyRead ()
- void closeSocket ()

  Close the connection with the server.
- void sendToServer (QString data)

  sendToServer Sends data to the remote lidar.

Signals

- void configuration ()

  Asked for configuring sensor.

Public Member Functions

- SickSocket (AbstractSickSensor *parent)

  Constructor.
- ~SickSocket ()

  Destructor.

Protected Slots

- void socketConnectionClosed ()

  Says to the user the connection is closed.
- void socketError (QAbstractSocket::SocketError e)

  Warns the user an error occured.

3.19.1 Detailed Description

The SickSocket class Handles the ethernet connection with the remote sensor.

3.19.2 Member Function Documentation

3.19.2.1 void pacpus::SickSocket::sendToServer (QString *data*) [slot]

sendToServer Sends data to the remote lidar.

Parameters

data	Data to be sent, translated in ASCII.

3.19.2.2 void pacpus::SickSocket::socketConnectionClosed ( ) [protected],[slot]

Says to the user the connection is closed.

protected slot

3.19.2.3 void pacpus::SickSocket::socketReadyRead ( ) [slot]

Called when incoming data is received. Create an event and send it to the sensor's handler.

See Also

AbstractSickComponent

The documentation for this class was generated from the following files:

- SickSocket.h
- SickSocket.cpp

Index